# Probabilistic Strategies for Pursuit in Cluttered Environments with Multiple Robots

Geoffrey Hollinger, Athanasios Kehagias, and Sanjiv Singh

***Abstract*— In this paper, we describe a method for coordinating multiple robots in a pursuit-evasion domain. We examine the problem of multiple robotic pursuers attempting to locate a non-adversarial mobile evader in an indoor environment. Unlike many other approaches to this problem, our method seeks to minimize expected time of capture rather than guaranteeing capture. This allows us to examine the performance of our algorithm in complex and cluttered environments where guaranteed capture is difficult or impossible with limited pursuers. We present a probabilistic formulation of the problem, discretize the environment, and define cost heuristics for use in planning. We then propose a scalable algorithm using an entropy cost heuristic that searches possible movement paths to determine coordination strategies for the robotic pursuers. We present simulated results describing the performance of our algorithm against state of the art alternatives in a complex office environment. Our algorithm successfully reduces capture time with limited pursuers in an environment beyond the scope of many other approaches.**

## I. INTRODUCTION

The applications of pursuit-evasion are as diverse as they are numerous. In urban search and rescue scenarios, it is often necessary to search urban environments for survivors or first responders, some of whom may be moving. In military applications, human or mechanized infantry often need to locate friendly or hostile targets in theaters of battle or peace keeping situations. Furthermore, pursuit-evasion is rooted in game theory and economics. World markets and the stock exchange operate under similar principles to those of pursuit-evasion. The goal of pursuit-evasion is to develop strategies for coordinating agents when the future actions of potentially adversarial agents are unknown or partially known.

The major application that has inspired our work is that of autonomous robotic assistance for human first responders in disaster scenarios using range-only sensors [14]. In such potentially confusing situations, robots can help to track the location of first responders and relay that information back to them. To track first responders with range-only sonar sensors, robots must maintain line-of-sight sensor contact with them. In noisy, dynamic environments, loss of contact with first responders is an inevitable possibility. Once robots have lost contact, pursuit-evasion strategies are necessary to relocate the first responder. This paper provides principled methods for coordinating robots in this scenario. Combining the methods we describe in this paper with range-only tracking methods yields a complete solution to the problem of tracking a non-adversarial first responder in a disaster scenario.

We make the assumption in this paper that the evader is *non-adversarial* and moves randomly in the environment. This assumption is consistent with the task of locating a first responder or moving survivor in a disaster scenario. The methods in this paper are also easily applicable to a stationary "evader" (see Section IV). For the task of finding a moving survivor, it may also be advantageous to utilize environmental clues such as smoke, voices, and heat gradients. If such information is known, it can easily be incorporated into our framework by modifying the dispersion model.

In this paper, we develop coordination algorithms for multiple robots pursuing a non-adversarial mobile evader in indoor environments. Our algorithms are designed to minimize the expected time of capture. This approach is particularly relevant in complex environments where the guaranteed capture of an evader would be lengthy or impossible. In many of the applications described above, it is desirable to capture the evader in as little time as possible to increase chances of survival, mission success, or profit. We formulate this problem using a probabilistic framework on a coarsely discretized map. We improve on methods previously used for locating stationary targets by developing a principled method for incorporating the evader's movement model, and we define several one-step cost heuristics for use in path planning. We propose a decoupled method for searching the space of paths in the environment, and we present simulated results showing the performance of our algorithm in a complex office environment. The novelties of our method include: the integration of motion modeling into a probabilistic representation of the evader's state, the formulation a scalable planning algorithm for multiple pursuers, and the use of entropy and target probability cost heuristics for pursuit-evasion planning. Our methods reduce expected time of capture with limited pursuers in an environment beyond the scope of many current methods.

This paper is organized as follows. Section II describes related work in the area of pursuit-evasion, surveillance, and search. Section III provides a mathematical definition of the pursuit-evasion problem for multiple pursuers searching for a mobile, non-adversarial evader. Section IV describes our coordination algorithm including our discretization technique, dispersion modeling, cost functions, and path plan-

This material is based upon work supported by the National Science Foundation under Grant No. IIS-0426945.

G. Hollinger and S. Singh are with The Robotics Institute, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213, USA gholling@andrew.cmu.edu, ssingh@ri.cmu.edu

A. Kehagias is with the Div. of Mathematics, Dep. of Math., Phys., and Comp. Sciences, Faculty of Engineering, Aristotle University of Thessaloniki, Thessaloniki GR54124, Greece kehagiat@mail.auth.gr

ning. Section V presents simulated results for a complex indoor environment. Finally, Section VI draws conclusions and discusses future directions for this work.

## II. RELATED WORK

Pursuit-evasion has been heavily researched in the fields of computer science, robotics, and mathematics. Cheng gives a short survey of previous work in pursuit-evasion [3]. This survey includes both recent and historical developments in the area. Parsons did some of the earliest work on examining pursuit-evasion in graphs [15]. He considered the graph to be a system of tunnels in which an evader was hiding, and he defined the *search number* of a graph: the minimum number of guards necessary to catch an adversarial evader with arbitrary speed. Adler et al. extended pursuit-evasion on graphs to randomized environments by examining the hunter and rabbit problem [1]. They define the *escape length* of a strategy as the worst case expected number of rounds for the hunter to catch the rabbit, and they derive error bounds for various hunter strategies. Isler et al. advanced the concept of probabilistic pursuit-evasion to polygonal environments [10]. They develop coordination strategies for one or two pursuers in simple polygonal environments based on the assumption that an adversarial evader does not have knowledge of some actions made by the pursuer.

More recently, LaValle and Guibas developed pursuit-evasion strategies for multiple pursuers in polygonal environments [7], [12], [13]. Their algorithm discretizes polygonal environments into conservative visibility regions and then uses an information space approach to develop complete algorithms that guarantee capture in simple environments. Gerkey extended these methods to cases where the pursuer has a limited field-of-view [6]. For a single pursuer, these algorithms are guaranteed to find a solution if one exists. When scaled to multiple pursuers, however, they lose this property. Additionally, these algorithms are difficult to extend to complex environments because of the sheer number of (often very small) cells necessary in a conservative visibility discretization. These algorithms are also not applicable to complex environments in which capture cannot be guaranteed. This drawback becomes particularly prominent with limited pursuers and in any environment with a loop.

Other similar research has been conducted in the areas of urban surveillance and urban search and rescue. For example, Hegazy presents methods for coordinating multiple robotic pursuers in urban environments [9], and Liao et al. develop a framework for tracking mobile targets in complex environments with limited sensor information [14]. Ferris, Hahnel, and Fox use a particle filter with Gaussian processes to track a human using wireless signal strength [5]. They discretize the environment into a mixed graph of 1D hallways and 2D rooms, but they do not extend their work into the pursuit-evasion domain. With some modifications, a particle filter method like theirs could be used to provide a similar role in pursuit-evasion planning as our dispersion and capture matrices. Our method provides a computationally efficient alternative to this approach.

Our algorithm falls most closely with those designed to solve the "search" variation of pursuit-evasion. Sarmiento et al. present a framework for searching polygonal environments with multiple robotic pursuers using a one-step cost heuristic [16]. They do not extend this work to mobile evaders, and they do not present results in large-scale environments. In Section V, we compare our cost-heuristics to those developed by Sarmineto et al.

The pursuit-evasion problem we discuss here has strong similarities to a problem which has been studied extensively in the *Markov Decision processess* (MDP) and *partially observable MDP* (POMDP) literature. The first formulation and solution of this problem is given by Eaton and Zadeh [4]. They deal with the problem of formulating an optimal control sequence for a pursuer chasing an evader through a finite number of positions. The pursuer attempts to place itself in the same position as the evader in the *minimum expected time*. Extensions for the case of multiple pursuers are immediate. The important difference between Eaton and Zadeh's (the MDP) problem and the one we discuss here is that they assume that the evader's position is always known. In our problem, the position of the evader is unknown, which implies that the joint pursuer/evader state is *partially observable*. The MDP problem has been extended to cover partially observable Markov chains as well, but a solution which is both exact and computationally practical has not yet been found [2], [8]. Our algorithm in this paper does not directly solve the underlying POMDP because of poor scalability when adding additional pursuers.

## III. PROBLEM DEFINITION

To describe the state of the world in a pursuit-evasion scenario, we must develop representations for the environment and the locations of the pursuers and evaders. Assume that the state of a pursuer at time $t$ is known to be $X^P(t)$ and the state of an evader at any time $t$ is known with a certain probability to be $X^E(t) = p$. We can adapt this formulation to multiple pursuers $i$ at any time $t$ by expanding $X^P(t)$ to $X_i^P(t)$. In a polygonal environment, this problem is continuous. To collapse the problem into a discrete domain, discretize the environment (see Section IV) into a number of cells, $1 \dots N$. The state of the $i^{th}$ pursuer is now fully defined by $X_i^P(t) = n$ where $n$ is the cell in which the pursuer is currently located. Now, define a capture event at time $t$ as the occurrence of $X_i^P(t) = X^E(t)$ for any pursuer $i$. To define the state of the evader, let $p$ be a row vector such that $p = [p_0, \dots, p_N]$ where values $p_1 \dots p_N$ represent the probability that the evader is in the corresponding cell. Let the value $p_0$ represent the probability that the evader has already been captured by the pursuers. Refer to this as the "capture state." The vector $p$ now defines a probability distribution function over the evader's position in the environment (with the addition of a capture state).

The pursuers' goal is to minimize the expected time of reaching a capture event. Thus, the pursuers seek to maximize the probability that the evader is in the capture state at any given time $t$. The coordination problem is then defined as the determination of paths for the pursuers such that the probability of capture is maximized at any given time.

## IV. ALGORITHM DESCRIPTION

### A. Map Discretization

Our method for discretization takes advantage of the inherent characteristics of indoor environments. To discretize an indoor map by hand, simply label convex hallways and rooms as cells and arbitrarily collapse overlapping sections. This method is simple enough that it can be performed by hand even for large maps. Fig. 1 shows an example with a small map, and Fig. 3 gives an example discretization for a large map. Taking into account the cell adjacency in a discretized map yields an undirected graph that can be searched by the pursuers. This ties our research into that of probabilistic graph search. Fig. 2 shows the undirected graph derived from the house map.

This method for discretization also has the advantage of guaranteeing that a pursuer in a given convex cell will have line-of-sight to an evader in the same cell. This allows the capture event to be reduced to the attainment of line-of-sight to the evader. This makes intuitive sense because gaining line-of-sight effectively collapses the unknown state of the evader to a known state. Gaining line-of-sight is relevant to nearly all sensors that a robotic pursuer would possess.

In comparison with the visibility-based discretization proposed by LaValle and Guibas [7], our discretization technique yields far fewer cells making it more applicable to large, complex environments. The tradeoff is that our discretization does not provide a discretization suitable for use with LaValle and Guibas's visibility-based pursuit algorithms.

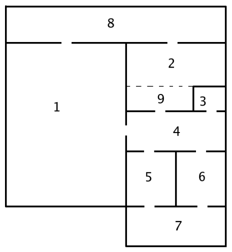

Fig. 1. Small house map used for pursuit-evasion simulation

### B. Dispersion and Capture Modeling

To integrate a motion model of the evader into our pursuit-evasion framework and better define capture events, we develop "capture" and "dispersion" matrices for application to the evader's state vector. As presented in Section III, the location of the evader is represented by a vector $p = [p_0, \ldots, p_N]$ where $p_0$ represents the probability the evader has already been captured, and $p_1 \ldots p_N$ represent the probability the evader is in the corresponding discretized cell. We can mathematically represent a capture event on that state

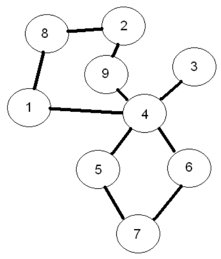

Fig. 2. Undirected graph built from house discretization

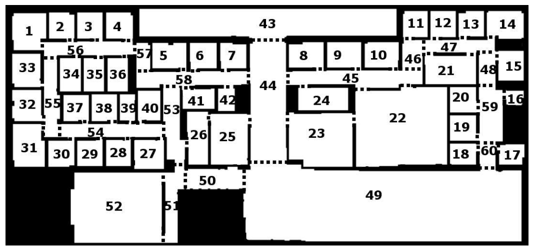

Fig. 3. Office building map used for pursuit-evasion simulation (dotted lines show discretization boundaries)

vector by defining a matrix that moves all probability[1] from all cells visible from pursuer $i$'s current cell $X_i^P(t)$ to the capture state. The appropriate capture matrix $C_{X_i^P(t)}$ for cell $X_i^P(t)$ is applied at time $t$ to yield $p(t+1) = p(t)C_{X_i^P(t)}$. For example, if we assume that the pursuer cannot see through doorways, the capture matrix for a pursuer in cell 1 of the environment in Fig. 1 would be the 10x10 identity with the second row unity value shifted to the first column.

Under the current assumption that capture is guaranteed in the pursuer's current cell and not possible in neighboring cells, the capture matrices contain diagonal unity values except for the row corresponding to the current cell. Relaxing either of these assumptions would allow for more complex capture matrices.

Similarly, we can define dispersion matrices to represent the expected motion of the evader in the environment. As discussed previously, discretization of the environment yields an undirected graph of possible evader movements between cells. Based on a motion model, we can assign probabilities to each of these movements and define a matrix that properly disperses the evader's probable location. We can then apply this matrix $P$ at time $t$ to yield a new evader state vector at time $t+1$ as in Equation 1.

$$p(t+1) = p(t)P \tag{1}$$

For instance, if we assume an equal probability that the evader will remain still or move to any adjacent cell at the next time step, the dispersion matrix for the environment in Fig. 1 would be:

[1]The capture matrix can also contain non-unity values if the probability of seeing an evader when it is in a pursuer's line-of-site is less than one. This would be the case with noisy sensors.

$$P = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{3} & 0 & 0 & \frac{1}{3} & 0 & 0 & 0 & \frac{1}{3} & 0 \\ 0 & 0 & \frac{1}{3} & 0 & 0 & 0 & 0 & 0 & \frac{1}{3} & \frac{1}{3} \\ 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 \\ \mathbf{0} & \mathbf{\frac{1}{6}} & \mathbf{0} & \mathbf{\frac{1}{6}} & \mathbf{\frac{1}{6}} & \mathbf{\frac{1}{6}} & \mathbf{\frac{1}{6}} & \mathbf{0} & \mathbf{0} & \mathbf{\frac{1}{6}} \\ 0 & 0 & 0 & 0 & \frac{1}{3} & \frac{1}{3} & 0 & \frac{1}{3} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{3} & 0 & \frac{1}{3} & \frac{1}{3} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{3} & \frac{1}{3} & \frac{1}{3} & 0 & 0 \\ 0 & \frac{1}{3} & \frac{1}{3} & 0 & 0 & 0 & 0 & 0 & \frac{1}{3} & 0 \\ 0 & 0 & \frac{1}{3} & 0 & \frac{1}{3} & 0 & 0 & 0 & 0 & \frac{1}{3} \end{pmatrix}$$

The top row of the capture matrix corresponds to the capture state. For reference, row 5 (emboldened) corresponds to the probabilities associated with cell 4. It is important to note that while there are as many capture matrices as there are cells in an environment, there is a single dispersion matrix for the entire environment.

The capture matrix of an evader $i$ and the dispersion matrix can be concatenated yielding a new evader state vector as in Equation 2.

$$p(t+1) = p(t)PC_{X_i^P(t)} \tag{2}$$

In larger environments, it is desirable to use multiple pursuers to search for a single evader. To expand to multiple pursuers, the capture matrices for all pursuers can also be concatenated to yield the new state as in Equation 3.

$$p(t+1) = p(t)P[\prod_{i=1}^{R} C_{X_i^P(t)}], \tag{3}$$

where R is the number of robotic pursuers.

To perform this step, it is necessary for the pursuers to communicate their states at each time step. Together, the dispersion and capture matrices provide a method for properly modifying the evader's probable position vector when advancing time steps.

The computational complexity of the dispersion and capture matrix application is determined by the number of cells in the environment (the size of the matrices) and the number of pursuers: $O(nc^3)$ where $n$ is the number of pursuers and $c$ is the number of cells in the environment. Since the matrices are often sparse, this complexity can sometimes be further reduced.

The dispersion and capture matrix formulation can also be easily modified to account for measurements that provide a prior on the distribution of the evader. If a pursuer receives information about the evader's position, e.g. range to the evader, and this information is insufficient to warrant a capture event, a matrix $M$ can be formed to encode this information and applied during the forward time-step. Another advantage of this problem formulation is the ease of which prior knowledge of the evader's position can be incorporated into the framework. If the evader's position is known with some certainty to be in a subset of the graph, the initial $p$ vector can be initialized with the appropriate probabilities. This is particularly relevant if the pursuers had line-of-sight and then lost it. In this case, they would know that the evader is near its previous known location. The incorporation of measurements and the use of prior knowledge further integrate our methods into a full tracking solution as described in Section I.

### *C. Cost Functions*

Since the expected time of capture of the evader cannot be known exactly during planning, we develop heuristic cost functions to guide coordination of robotic pursuers.

The simplest cost function shown in Equation 4 is the probability of failing to capture the evader when moving into a cell.

$$C(x, p(t)) = 1 - p_x(t)], \tag{4}$$

where $x$ is the cell that the pursuer is about to enter.

Another simple cost heuristic shown in Equation 5 is to take the distance it takes to reach a given cell and divide it by the probability of capture in that cell. This is the cost heuristic employed by Sarmiento et al. for searching polygonal environments [16].

$$C(x, p(t)) = \frac{D(x, X_i^P(t))}{p_x(t)}, \tag{5}$$

where $x$ is the cell that the pursuer is about to enter, and $D(x, y)$ is the distance between cells $x$ and $y$.

The above cost heuristics have a disadvantage in that they do not take into account probabilities across the entire graph. They only examine the probability of capture in the cell to which the pursuer is about to move. When planning more than one step ahead, expected time of capture is minimized if the pursuer maximizes capture probability in as little time as possible. Aggregating the previous two cost heuristics over several steps does not take this into account.

The following cost functions utilize the probability of non-capture over all cells. This punishes the robotic pursuer for the remaining probability in all cells and relates more directly to expected time of capture.

One such cost heuristic shown in Equation 6 is the vectorial distance between the evader's current state probability vector and an ideal capture vector.

$$C(p(t)) = \sum_{n=0}^{N} |p_n(t) - \overline{p}|, \tag{6}$$

where the target probability $\overline{p} = [1, 0, \ldots, 0]$.

Entropy is another one-step cost function that takes into account the state of every cell as shown in Equation 7.

$$C(p(t)) = -\sum_{n=0}^{N} p_n(t) \log p_n(t), \tag{7}$$

where $N$ is the number of cells (including capture state).

Adding a summation to any of the above cost functions allows the robotic pursuers to search several cells ahead to determine the optimal pursuit path. The resulting cost function for a path is shown in Equation 8.

$$C(path) = \sum_{x=x_0}^{x_d} C(x, p(t)), \quad (8)$$

where $x_0$ is the starting cell and $d$ is the cell depth.

The task of searching all paths is simplified because the discretization method only allows a small number of options in each cell (the pursuer must move to one of the adjacent cells). Furthermore, all possible paths to a given depth can be searched with minimal computation by storing the costs of previous paths and branching into a breadth first tree at replanning points.

When planning using cost functions, locations in the environment must be assigned as replanning points. These locations, designated as checkpoints, represent the points in the environment between which the robots move. The most obvious checkpoints to use are the centroids of cells because they provide quick access to adjacent cells. Another possible location for checkpoints would be cell boundaries. In our experiments, we found that centroids and cell boundaries performed similarly, and we use centroids as checkpoints in our results in Section V.

### D. Coupled Coordination

Once a cost function has been defined, coordination strategies for the robotic pursuers can be found by searching possible paths in the discretized floor plan. To develop a coupled coordination strategy, a centralized planner can search all possible paths on the pursuit-evasion graph to a given depth for every robotic pursuer. This will output paths for all pursuers that minimize the cost function. This algorithm can be decentralized by assigning identification numbers to the pursuers and forcing each pursuer to plan for all pursuers.

The advantage of using a coupled coordination strategy is that the pursuers take the future positions of the other pursuers into account during planning. However, coupled coordination scales exponentially with the number of pursuers. The number of cells that must be searched is $O(b^{dn})$ where $n$ is the number of pursuers, $d$ is the lookahead depth, and $b$ is the maximum branching factor of the cell graph. This does not lead to a tractable solution for more than a small number of robotic pursuers with a short lookahead.

### E. Decoupled Coordination

To decouple the planning algorithm, we allow each pursuer to plan for itself while assuming that the states of the other pursuers are fixed. This prevents the search space from growing in complexity with the number of pursuers. With this assumption, each pursuer must simply search for its optimal path given the current state information of the other pursuers. The complexity of this planning algorithm is not affected by the number of pursuers: $O(b^d)$. Even though planning is decoupled, the initial positions of other pursuers provide information that the evader is not in that cell. For this purpose, our algorithm always communicates pursuer locations at each time step.

Algorithm 1 gives a summary of the entire pursuer coordination algorithm. We show results from both coupled and decoupled planning in Section V.

**Algorithm 1** Pursuer coordination algorithm

```
while evader not captured: $X_i^P(t) \neq X^E(t)$ for any $i$ do
  for all pursuers $i$ do
    Transmit current pursuer's location: $X_i^P(t)$
    Update $X^P(t)$ with info from other pursuers
    Apply dispersion and capture matrices:
       $p(t) \leftarrow p(t-1)P[\prod_{i=1}^{R} C_{X_i^P(t)}]$
    if at checkpoint then
      $x_0 \leftarrow X_i^P(t)$
      if coupled planning then
        for all coupled pursuers' paths to depth $d$ do
          Calculate: $C(path) = \sum_{x=x_0}^{x_d} C(x, p(t))$
        end for
        Set lowest cost goal for current robot ID
      else if decoupled planning then
        for all current pursuer's paths to depth $d$ do
          Calculate: $C(path) = \sum_{x=x_0}^{x_d} C(x, p(t))$
        end for
        Set lowest cost goal from planner
      end if
    else
      Move current pursuer towards next checkpoint
    end if
  end for
  $t \leftarrow t + 1$
end while
```

## V. SIMULATED RESULTS

To test the above coordination algorithms, we developed a multi-agent pursuit-evasion simulation in C++ on a 3.2 GHz Pentium 4 processor. Our simulation environment allows for simulation of multiple pursuers and evaders. Fig. 4 gives screenshots of the simulation at different time steps during a pursuit-evasion trial. The screenshots show a test run using the office building map with two pursuers (P1 and P2) and one evader (E).

To formulate the specific dispersion and capture matrices for the simulation, we needed to make several assumptions. We simplified the formulation of the capture matrices by assuming that the robotic pursuers will always see an evader in the same cell and never see an evader in another cell. This approximation is reasonable because of the coarse and convex discretization of the environment. We derived the dispersion matrices from the possible paths of the evader in each cell. The probability of remaining in the current cell is proportional to the area of the cell, the speed of the evader, and the number of adjacent cells. Similarly, the probability of moving to an adjacent cell is proportional to the size of the cell and the number of adjacent cells. We assumed that the speed of the evader and pursuers were known to be 1 m/s,

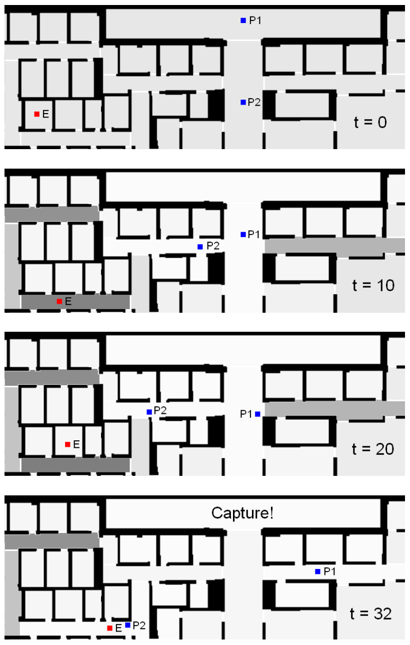

Fig. 4. Snapshots of pursuit-evasion simulation at different time steps until capture event. The pursuers automatically branch into the two major cycles on the map. Darker cells denote more probable evader locations.

and that they move holonomically between cell boundaries and centroids.

## A. *Complex Environment*

We performed experiments in a complex office environment discretized as shown in Fig. 3. Two pursuers were placed at the top of the map in cells 43 and 44, and an evader was randomly placed in one of the other cells. The evader moves randomly between the centroid and boundaries of the cells, and the pursuers moves in accordance with our coordination algorithms described above. Two other pursuer coordination methods were added to the results for comparison. In the *random* method, the pursuer randomly moves between the boundaries and centroids of adjacent cells. In the *stationary* pursuer method, the pursuer remains in its starting cell. Fig. 5 gives the results of pursuit-evasion trials on the office building floor plan.[2]

The results in Fig. 5 compare planning using four different cost heuristics. The probability and distance cost heuristic was introduced by Sarmiento et al. [16]. Since this method was developed for stationary targets, we augment their planning method with our dispersion matrix to model a mobile evader. Visibility-based methods, such as those developed by LaValle and Guibas [7], are not possible with two pursuers on this map. This is obvious by inspecting the number of cycles on the map. A visibility-based discretization of this map would also yield nearly one-thousand cells, and the

[2]The average capture times on this map were 749.319 +/- 24.6114 seconds for the stationary strategy and 344.385 +/- 14.6694 seconds for the random strategy. They have been left out of the figure to better show trends in the other results.

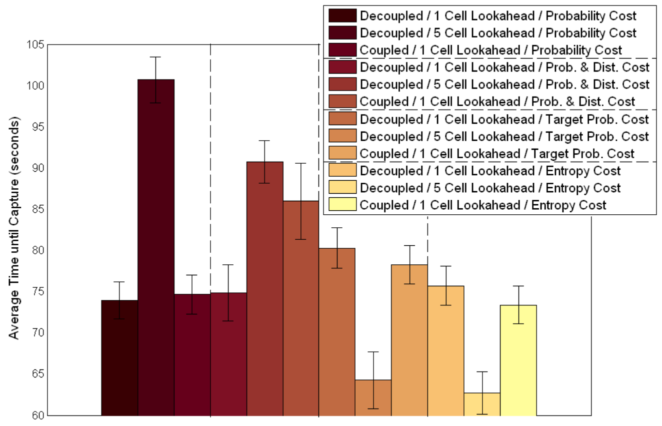

Fig. 5. Average simulated pursuit-evasion capture times for two pursuers over 1000 trials for coupled and decoupled coordination strategies with different cost heuristics on a complex office map. All methods used dispersion matrices, and lookahead trials with coupled planning were not feasible on this map. Error bars represent one standard error of the mean (SEM). See footnote for results of random and stationary strategies. Starting location of evader randomized, and starting location of pursuers fixed.

number of edges in any given visibility cone would make computation of a solution in this environment infeasible.

The office building results show that the decoupled planning methods with lookahead, capture matrix dispersion, and the entropy or target probability cost heuristics yield the lowest expected times of capture. These methods reduce the time of capture by a factor of five over the random method and by 17% over the methods with the cost heuristic introduced by Sarmiento et al. [16].

The paths of the pursuers in the better performing methods diverge to explore the two major hallways on the map. The pursuers move around these cycles until they block the evader's path. This coordination strategy stems from the use of the dispersion matrices to model the evader's movement. Since the evader is moving, searching the small rooms in the environment does not yield low expected times of capture. Instead, it is advantageous for the pursuers to move through the long hallways and encounter the evader as it moves out of the rooms. These results coincide with intuition, and they further confirm that our methods lead to effective multi-robot coordination strategies.

The decision to use two pursuers on the office map was made because of the size of the environment and the two major hallway cycles. Experiments with coordinating up to five pursuers on this map were successful at reducing time of capture over simpler methods for most evader paths. In some cases, adding additional pursuers does not reduce capture time because one of the pursuers gets *lucky* and catches the evader in a low probability cell. Additional pursuers decrease the luckiness of captures by reducing probabilities in other areas of the map. This further demonstrates that our algorithms can often perform well with few pursuers.

In the office environment, adding dispersion to the evader's motion model greatly helps in catching a mobile evader.

Adding a five cell lookahead further reduces capture time for the entropy and target probability cost heuristics. For the probability and probability & distance cost heuristics, adding a lookahead actually hurts planning. This is because these cost heuristics are inherently shortsighted in that they do not take into account probability across the entire graph. Thus, these cost heuristics do not differentiate between *when* the cost is reduced. To minimize expected capture time, it is advantageous for the pursuers to reduce cost early in the planning cycle. The entropy and target probability cost heuristics account for this.

We also ran several trials with a stationary evader using the identity as the dispersion matrix. This forces the pursuers to inspect each room in the map rather than cutting off the cycles. In this case, our coordination methods with the entropy or target probability cost heuristic were effective at reducing the time of capture over simple methods, and were not significantly different from those with the probability and distance heuristic introduced by Sarmiento et al. [16]. This shows that our methods are also effective at coordinating pursuers to search environments for immobile targets.

## VI. CONCLUSIONS AND FUTURE WORK

Our coordination methods for robotic pursuers effectively reduce the expected time of capture in a complex pursuit-evasion scenario. We have shown through simulated results that searching for lowest cost paths generates plans that greatly reduce expected capture time versus random or stationary search methods. Our algorithms also reduce expected capture time by 17% over methods introduced by Sarmiento et al. [16], and they perform in environments beyond the scope of methods introduced by LaValle and Guibas [7]. We have also demonstrated that the decoupled version of our planning methods works effectively, allowing our methods to scale to both a large number of robotic pursuers and a significant lookahead. Furthermore, we have shown that entropy and target probability cost heuristics improve on simpler heuristics that rely solely on distance and probability of capture. Our dispersion and capture matrix formulation integrates target motion modeling into the planning framework, and we successfully demonstrate a method for generating a coarse discretization of indoor environments that is scalable to large and complex maps. Our research provides a probabilistic solution to solving the problem of searching for non-adversarial mobile evaders in indoor environments with multiple robotic pursuers.

For future work, we plan to examine the performance of our algorithm on maps with more cycles. Maps of museums with cyclical galleries are common examples of this type of environment. These maps add complexity to the pursuit-evasion problem because they prevent the pursuers from cutting off all cycles and limiting evader movement. Additionally, we plan to extend this algorithm to a POMDP formulation. The state representation already exhibits the form of a POMDP with the state of the evader as the partially observed state. A coordination algorithm could be developed by solving this POMDP as in previous POMDP research [11]. The major drawback to this course of action is the poor scalability of POMDPs when adding additional pursuers.

Furthermore, our method should be extended to multiple evaders. This extension would require an increase in the dimensionality of the evader's state estimation vector and a modification of the cost functions. We also plan to incorporate constraints (such as minimizing distance between pursuers) and the use of noisy measurements into the current framework. Our method for discretization could be automated by arbitrarily collapsing convex regions found using the Quine-McClusky method [17]. We have not implemented this automatic discretization in this paper, but its implementation is straightforward. Finally, we plan to test our algorithms on physical robots and verify their performance in some of the real-world applications discussed in Section I.

## VII. ACKNOWLEDGMENTS

We gratefully acknowledge Christopher Geyer for his insightful comments.

## REFERENCES

[1] M. Adler, H. Racke, N. Sivadasan, and C. Sohler, "Randomized Pursuit-Evasion in Graphs," *Combinatorics, Probability, and Computing*, 12:225–244, 2003.

[2] D. Bertsekas and J. Tsitsiklis. *Parallel and distributed computation*, Prentice-Hall, 1989.

[3] P. Cheng, "A Short Survey on Pursuit-Evasion Games," Department of Computer Science, University of Illinois at Urbana-Champaign, 2003.

[4] J. Eaton and L. Zadeh, "Optimal pursuit strategies in discrete-state probabilistic systems," In *Trans. ASME Ser. D, J. Basic Eng*, vol. 62, pp.23-28, 1962.

[5] B. Ferris, D. Hahnel, and D. Fox, "Gaussian Processes for Signal Strength-Based Location Estimation," In *Proc. of Robotics Science and Systems*, 2006.

[6] B. Gerkey, S. Thrun, and G. Gordon, "Visibility-based pursuit-evasion with limited field of view," *Int'l Journal of Robotics Research*, 25(4):299–315, 2006.

[7] L. Guibas, J. Latombe, S. LaValle, D. Lin, and R. Motwani, "Visibility-based pursuit-evasion in a polygonal environment," *Int'l Journal of Computational Geometry and Applications*, 9(5):471–494, 1999.

[8] E. Hansen and Z. Feng, "Dynamic programming for POMDPs using a factored state representation," In *Proc. 5th Int'l Conf. on AI, Planning, and Scheduling*, 2000.

[9] T. Hegazy, "A Distributed Approach to Dynamic Autonomous Agent Placement for Tracking Moving Targets with Application to Monitoring Urban Environments," Ph.D. Dissertation, Georgia Tech, 2004.

[10] V. Isler, S. Kannan, and S. Khanna, "Randomized Pursuit-Evasion in a Polygonal Environment," *IEEE Trans. Robotics*, 5(21):864–875, 2005.

[11] L. P. Kaelbling, M. L. Littman, A. R. Cassand, "Planning and acting in partially observable stochastic domains," *Artificial Intelligence*, 101:99–134, 1998.

[12] S. M. LaValle, D. Lin, L. J. Guibas, J.-C. Latombe, and R. Motwani, "Finding an unpredictable target in a workspace with obstacles," In *Proc. Int'l Conf. on Robotics and Automation*, pages 737–742, 1997.

[13] S. M. LaValle, *Planning Algorithms*, Cambridge Univ. Press, 2006.

[14] E. Liao, G. Hollinger, J. Djugash, and S. Singh, "Preliminary Results in Tracking Mobile Targets Using Range Sensors from Multiple Robots," in *Distributed Autonomous Robotic Systems*, Vol. 7, M. Gini and R. Voyles, eds., Tokyo: Springer-Verlag, pp. 125–134, 2006.

[15] T. D. Parsons, "Pursuit-evasion in a graph," In *Theory and Applications of Graphs*, Y. Alavi and D. Lick, eds., Springer, pp. 426–441, 1976.

[16] A. Sarmiento, R. Murrieta-Cid, and S. Hutchinson, "A Multi-robot Strategy for Rapidly Searching a Polygonal Environment," in *Proc. 9th Ibero-American Conf. on AI*, Puebla, Mexico, 2004.

[17] J. S. Singh, M. D. Wagh, "Robot Path Planning using Intersecting Convex Shapes: Analysis and Simulation," *IEEE Journal of Robotics and Automation*, vol. RA-3, no. 2, April 1987.