

# A New Method of Using Sensor Network for Solving Pursuit-Evasion Problem

Peng Zhuang, Yi Shang, and Hongchi Shi

Department of Computer Science

University of Missouri-Columbia

Columbia, MO 65211

pz797@mizzou.edu, {shangy,shih}@missouri.edu

**Abstract**—Wireless sensor networks offer the potential to significantly improve the performance of pursuers in pursuit-evasion games. In this paper, we study several sensor network systems, their interaction with the pursuers, and the effect on pursuer performance. We propose a general framework to solve the pursuit-evasion problem and present new centralized as well as distributed methods. Specifically, we address three issues in the design of pursuers based on data provided by the sensor network : a) how to identify evader moving patterns, (b) how to predict the evader locations using different evader moving models, and c) how to choose the most efficient pursuit strategies. We propose efficient algorithms to solve these problems and show that they are effective in reducing the capturing time in our simulations. We also compare the distributed and centralized methods. Experimental results show that the distributed method is efficient and produces solutions close to the centralized method.

**Index Terms**—Pursuit-evasion game, pursuer collaboration, distributed planning, probabilistic map, sensor networks

## I. INTRODUCTION

The pursuit-evasion problem (PEP) has been an important research topic for many years [2], [3], [8], [7]. The basic problem is to study how to guide one or a group of pursuers to catch a number of mobile evaders. Many real-world applications can be formulated into PEPs, such as search and rescue operations, surveillance, localization and tracking of moving parts in a warehouse, and search and capture missions. In some cases, the evaders are actively avoiding the pursuers as in capture missions, whereas in other cases their motion is approximately random as in rescue operations.

Recently, with the development of wireless sensor networks, it is becoming feasible to deploy a large-scale network with thousands of wireless sensor nodes. Initial work on applying wireless sensor networks to PEPs has shown great potential in improving the performance of pursuers [12]. A challenge in applying wireless sensor networks to PEPs is the energy, communication, and computation constraints of the sensor nodes. Without careful system design, a sensor node using normal batteries (as in MICA/MICA2) will deplete its batteries very quickly. It is well known that communication between sensors is the most energy consuming action, and thus effective wireless sensor network design and operation involve

much careful consideration of communication overhead. Furthermore, sensor nodes have very limit computation resources, which requires simple and efficient algorithms. Finally, sensor nodes are subject to various problems such as transmission failure, radio delay [16], and false and missing detection.

### A. Related Works

To the pursuers, finding the location of an evader is one of the most important yet difficult tasks. Without the assistance of sensor networks, the task is usually performed by the on-board sensors of the pursuers, and the detection information is usually reported to and stored at a central controller. Vidal et al. [14] presented a centralized planning system to control a small number of pursuers. The system maintains a two-dimensional map at the central controller. Slower grounded pursuers search the field and report their detection to the central controller who continuously guides the faster aerial pursuer to capture the evader. A probability method is used in this work to predict the evader location to overcome the uncertainty caused by the delayed message and local information.

Using the information provided by the sensor network, pursuers are given extra sight that allows them to plan more efficient pursuit trajectories. One underlying technology is target tracking in sensor network. There are many works in this area over the years, which can be categorized into two classes. One class of problems focus on tracking a single mobile target. While the problem itself is relative simple, most of the works concentrate on scheduling and energy saving of the sensor nodes [6], [15]. Another class of problems study how to track multiple targets. The main challenge is how to associate the detections with the correct targets. Some methods are based on the principle of machine learning [4], [13]. Another method using Markov chain Monte Carlo data association [9] has also been proposed [10]. A framework for the PEP using [9] is presented [11].

Using wireless sensor networks, various centralized and distributed algorithms have been proposed to keep track of multiple targets and assign targets to pursuers. A representative centralized method for sensor network aided pursuit-evasion games is proposed in [11], where each sensor node connects to a *supernode* with long range

of communication and is assumed to be fault-free. The supernodes are connected with the base station. Every single sensor detection is sent to the base station via a supernode. After retrieving the detection information, the base station predicts the evader locations and designs the pursuit strategies. The commands are sent to the pursuers afterwards via a direct connection (e.g., satellite communication).

Various distributed methods have been proposed that store the trace information of the evaders inside the sensor networks and allow the pursuers to communicate with the nearby sensor nodes to get the information. In [5], the sensor network maintains a tracking tree rooted at the evader. When a pursuer comes across a branch of the tracking tree, it follows the branch to its root in order to catch the evader. Every time a sensor detects a target, the detection is propagated from the sensor node to the whole sensor network. When the single pursuer retrieves the detection, it pursues the target in a greedy manner. Because the sensor network is assumed to be fully connected and fault-free, the pursuer always captures the evader as long as the pursuer moves faster. In addition, because there are only one evader and one pursuer, no specific concern is given to the pursuer strategy or coordination. Other distributed methods for tracking targets include the hierarchical multiple-target tracking (MTT) algorithm [10] and Markov chain Monte Carlo data association [9].

### B. Our Contribution

In this paper, we study several issues in applying sensor networks to the PEP and present a new method with both centralized and distributed implementations. We show that while the centralized method provides better performance, the distributed method imposes fewer requirements on the network, is more responsive, and is able to achieve good results compared with the centralized method. In sensor network design, we propose a new communication protocol. We analyze different evader moving patterns and present a probabilistic inference method based on Markov models to predict the evader locations. We also propose an efficient multivariate Gaussian approximation model and a machine learning method to classify the evaders based on their moving history. Finally, we present a method that can choose a proper pursuit strategy according to evader moving patterns.

Different from the existing PEP solutions that assume a particular evader moving dynamic and design a pursuit strategy accordingly, our approach is adaptive to different evader moving patterns. In addition, our approach requires less computation resources on the sensor nodes and can be implemented in either a centralized or distributed manner.

The rest of the paper is organized as follows. In the next section, we present the formulation of the problem. In Section III, we give an overview of our approach. In Section IV, We present the centralized and distributed methods and discuss their tradeoffs. In Section V, we

present the probabilistic inference method in predicting evader locations and the classification method in identifying evader classes. In Section VI, we address the problem of choosing the most efficient pursuit strategy. In Section VII, we present experimental results to compare the performance of various algorithms, and finally in Section VIII we conclude the paper.

## II. PROBLEM FORMULATION

Our environment for the PEP is a grid of  $N \times N$  cells. The coordinates of a cell  $C_i$  are  $(x_i, y_i)$ , and  $C_i$  is the neighbor of  $C_j$  if and only if

$$|x_i - x_j| \leq 1 \quad \text{and} \quad |y_i - y_j| \leq 1. \quad (1)$$

Sensor nodes are deployed in some cells. In the fully-covered case, every cell has a sensor node, whereas in partially-covered cases, a cell will have a sensor node with certain probability. The detection range of a sensor node is limited to the cell. A sensor node can communicate with its neighboring nodes as defined in Eq. (1). In a fully-covered case, the sensor nodes are guaranteed to be connected to form a single network. In partially-covered cases, multiple disconnected networks may be formed.

The centralized system has a base station that is at a random location. We assume the base station has direct communication to the pursuers. Task allocation is done at the base station, and tasks are assigned to the pursuers. In contrast, there is no base station in the distributed system.

The pursuers and evaders move in discrete time steps. During one time step, an evader moves to a neighboring cell from its current location, and a pursuer moves to a cell at most  $v_p$  units away.

The detection of an evader is assumed to be binary, either detected or not detected. A sensor node detects the evader(s) within the same cell and associates the evader id with the detection.

Capturing is defined in such a way that an evader in cell  $c_i$  is caught by a pursuer in cell  $c_j$  if and only if

$$|x_i - x_j| \leq r_p \quad \text{and} \quad |y_i - y_j| \leq r_p, \quad (2)$$

where  $r_p$  is the capturing radius of a pursuer.

The objective of the pursuit-evasion problem is to minimize the time used to capture all evaders. In the discrete world, the time-to-capture or capturing time  $T$  is defined as the number of time steps elapsed until all evaders are captured.

A sample environment is shown in Fig. 1. A triangle mark denotes a sensor node, and a square mark denotes an evader (with the evader id shown at the top right). A pursuer is denoted by a circle mark with its id at the top right, and the capturing range is denoted by a surrounding square.

## III. OVERVIEW OF OUR APPROACH

In this paper, we solve the problem of multiple pursuers and multiple evaders with different moving patterns in both distributed and centralized manners. We extend the

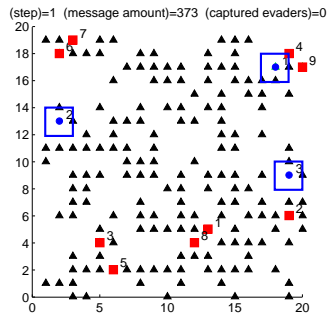


Fig. 1. An example with 3 pursuers, 9 evaders, and a sensor network with 40% coverage.

centralized method in [11] with pursuer collaboration and evader class identification. Our distributed method works with either a single connected sensor network corresponding to the fully-covered case or multiple disconnected networks corresponding to the partially-covered cases. The detection is broadcasted from a sensor node to other connected sensor nodes. The pursuers communicate with the nearby sensor nodes to retrieve the information and leave messages about their own behaviors on the sensor node (e.g., "I am about to pursuit evader A toward the coordinates (23.4, 56.4)"), which are broadcasted to other sensor nodes and pursuers. Upon receiving the information about evaders and other pursuers, a pursuer predicts the evaders' locations and movement and makes pursuit decision individually. The details of the communication protocols for the distributed and centralized system are discussed in Section IV.

Usually pursuers do not have the exact locations of the evaders due to either transmission delay or inaccurate sensing data. In addition, if the sensor network is not a single connected network, some pursuers may be in a subset disconnected from the subset that detects the evader. When the pursuer moves to the other subset and picks up the data, the data are delayed. In Section V, we present a new probabilistic inference method inspired by [14] to predict the current evader locations based on the delayed data and different evader moving patterns. This method is designed based on Markov network and is shown to perform well in PEP. The problem of the inference method in [14] is that it requires large computational resources, which is usually not feasible on sensor nodes. We propose a more efficient Gaussian approximation model, which is learned prior to the field operation and is downloaded to the sensor nodes during installation. Then during operation, the computational requirement on sensor nodes is very low. Furthermore, we analyze different evader moving patterns and propose an efficient algorithm to identify the evader classes.

Most existing works on PEP assume the evaders having one moving pattern and design specific pursuit strategies. In our work, we design an adaptive method that can work with different evader moving dynamics. In Section VI, we present a new heuristic method with several coordination

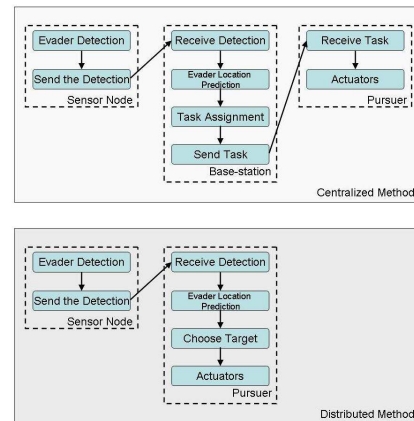


Fig. 2. Architecture of Centralized and Distributed Systems

and collaboration strategies that can adaptively select the most effective ones during operation. Fig.[2] shows the system architecture of the distributed and centralized method.

#### IV. NEW TARGET TRACKING METHODS

In the centralized method of [11], the sensor nodes send messages to the supernodes in a multi-hop manner who forward the messages to the basestation via a 1-hop connection. The connection between the supernodes and the base station is assumed to be reliable. Therefore, although the field is not necessarily fully covered, the sensor network is fully connected. In addition, planning and predicting are all performed in the base station, and the decisions are sent to the pursuers via another long range and reliable communication line. Although this method relies on some simplified assumptions, it usually performs better than the distributed method in general. We extend this method with algorithms for pursuer collaboration and evader moving pattern recognition. The solution in the centralized manner can be adopted if the assumption is satisfied in the field to achieve better performance.

A more realistic setup assumes there is no reliable connection between the supernode and the base station, nor is there a directed communication channel between the pursuers and the base station. Under such a condition, the sensor network is usually both partially covered and partially connected. The centralized algorithm performs badly because the base station does not have the global knowledge and is not able to send out the pursuit assignments instantly. We propose a distributed framework where the prediction and the decision are made locally by the pursuers. The information is shared within a subnet of the sensor network, and the mobile pursuers pick up the messages when they visit the subnet. Table I shows a comparison between the distributed system and the centralized system.

##### A. Distributed Protocol

Each sensor node keeps a data set for every evader denoted by  $S(i, k) = (t_k, x_k, y_k)$ , where  $i$  is the sensor

TABLE I  
DISTRIBUTED METHOD V.S. CENTRALIZED METHOD

	Distributed method	Centralized method
Evader detection	Sensor nodes detect the evader in the same cell	Same as the distributed method
Sensor network communication	Messages are propagated to the connected sensor nodes via multi-hop	Messages are sent to the supernode via multi-hop who forwards them to the base station via 1-hop
Pursuer communication	Pursuers communicate with the sensor node in the same cell	Pursuers communicate with the base station via long range and reliable connection
Predicting & planning	Performed by pursuers	Performed in the base station
Pursuer coordination & collaboration	Pursuers leave messages at sensor nodes, which are picked up by other pursuers later	By the base station

id,  $k$  is the evader id,  $t_k$  is the detection time, and  $(x_k, y_k)$  is the detected position of evader  $k$ .

While a sensor  $i$  has a new detection for evader  $k$  in time step  $t_k$ , it updates its own data and also sends out the message to other sensor nodes.

While a sensor node receives a message, it compares the message with its own data and keeps the most recent message.

When a sensor node receives a message older than its original data, it sends back a updating message to the source with the most recent message.

Every pursuer  $j$  has a similar data set  $S'(j, k) = (t_k, x_k, y_k)$  for every evader  $k$ . A pursuer talks to the sensor node in the current cell, then compares its own data with the data obtained from the sensor, and keeps the most recent data.

The pursuers also communicate with each other via the sensor network. The method is similar and is further discussed in Section VI.

### B. Centralized Protocol

The base station keeps a data set for every evader denoted by  $S(k) = (t_k, x_k, y_k)$ , where  $k$  is the evader id,  $t_k$  is the detection time, and  $(x_k, y_k)$  is the detected position of evader  $k$ .

While a sensor  $i$  has a new detection for evader  $k$  in time step  $t_k$ , it sends the message without storing it to the base station via its nearby supernode.

While the base station receives a new message, it compares the message with its own data and keep the most recent one.

The base station updates its decision as soon as there is an update on its data set, and the new assignments are sent to and received by the pursuers instantly.

## V. EVADER MOVING PATTERNS AND EVADER LOCALIZATION

Different evader moving patterns require various location prediction algorithms for the purpose of maximal

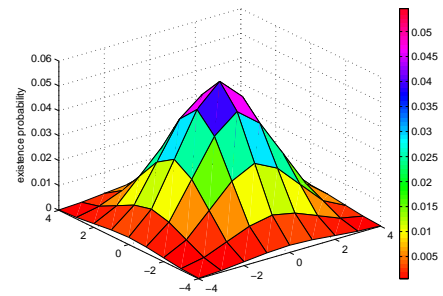


Fig. 3. Existence probability of an evader after 5 time steps in a 2-D plane. The center of the figure is the location of the last detection.

accuracy. In cases such as capture missions, an omniscient evader is capable of detecting the pursuer positions and actively escaping from the nearest pursuer. In other cases like rescue operations, evaders perform more random motions. In this section, we introduce two typical evader moving patterns, analyze their expected capturing time, and propose inference algorithms. We also propose an evader moving pattern recognition algorithm based on the principle of machine learning.

### A. Random Moving Evader (RME)

A random moving evader makes random moves regardless the position of the pursuers. In [14], Vidal et al. present a Markov model to reason about the existence probability of a evader from the probability in the previous time unit. It assumes that an evader has the equal probability to move to each of the neighboring cell or stay within the same cell. Let  $\Lambda(C_i)$  denote all the neighboring cells (up to 8) of cell  $C_i$ . Let  $\bar{C}_i$  denote one of the neighboring cells of  $C_i$  with  $\bar{C}_i \in \Lambda(C_i)$ . Let  $P(k, C_i, t)$  denote the existence probability of evader  $k$  in cell  $C_i$  at time step  $t$ . We have

$$P(k, C_i, t) = \frac{P(k, C_i, t-1) + \sum_{\bar{C}_i \in \Lambda(C_i)} P(k, \bar{C}_i, t-1)}{9} \quad (3)$$

The model assumes that the status of time  $t+1$  only depends on the status of time  $t$ . To obtain the probability of time  $t$ , we have to use a chain function that computes all probabilities from time 0 to time  $t$ . This is quite expensive if the message delay is large.

We approximate the probability by using a Gaussian model. For example, the existence probability of an evader after 5 time steps is shown in Fig. 3, where the center of the figure is the location of the last detection.

The probability follows a multivariate Gaussian distribution on the random variable as the values of x-coordinate and y-coordinate with mean vector  $\mu$  and the covariance matrix  $\Sigma$ . The two entries of  $\mu$  are the coordinates of the last detection, and each entry in  $\Sigma$  is a polynomial function of the number of time steps after the last observation. The method to learn the model is based on the principle of machine learning, which is well studied [1].

Let  $C_j$  denote the last detection,  $F(t) = \Sigma$  denote the function to compute the covariance matrix, and  $G(X, \mu, \Sigma)$  denote the probability in data point  $X$  for a Gaussian distribution. Eq. (3) is approximated as

$$P(k, c_i, t) = G(C_i, C_j, F(t)). \quad (4)$$

Based on Eq. (3), the distribution of an RME's position has the mean location at the place of the first detection. Thus the expected capturing time  $E(T)$  only depends on the pursuer's speed  $v_p$  and the distance between the assigned pursuer  $j$  in cell  $C_j$  and the position of the last detection  $C_d$ . The expected capturing time is

$$E(T) = \frac{d(C_j, C_d)}{v_p}. \quad (5)$$

In the centralized method, the detections are sent to the base station which performs the prediction. Eq. (3) is adopted and yields a more accurate prediction. To the contrary, in the distributed system, the detections are sent to the pursuers to process. We use Eq. (4) since it requires less computational resources.

### B. Active Escaping Evader (AEE)

An AEE always tries to maximize the distance between itself and the nearest pursuer. Assume the next move (cell id) is  $C_x$  and the position of the closest pursuer  $j$  is  $C_j$ . The distance between  $C_x$  and  $C_j$  is denoted by  $d(C_x, C_j)$ . The evader chooses the next move  $C_x$  as

$$C_x = \arg \max_{C_x} d(C_x, C_j). \quad (6)$$

The inference about the evader's position  $C_t$  at time  $t$  is straightforward given its position  $C_{t-1}$  at time  $t-1$  and the exact position  $C_{j,t-1}$  of the closest pursuers at time  $t-1$ . We have

$$C_t = f(C_{t-1}, C_{j,t-1}) = \arg \max_{C_x \in \Lambda(C_{t-1})} d(C_x, C_{j,t-1}). \quad (7)$$

Performing a chain function on Eq. (7) from the time of the last detection (time 0) to the current time step  $t$ , we have the current position  $C_t$  as

$$\begin{aligned} C_t &= f(C_{t-1}, C_{j,t-1}) \\ &= f(f(C_{t-2}, C_{j,t-2}), C_{j,t-1}) \\ &= \dots \\ &= f(\dots f(f(C_0, C_{j,0}), C_{j,1}) \dots), C_{j,t-1}). \end{aligned}$$

Computing the expected capturing time for an AEE is more complicated than that for an RME. Because an AEE only moves one step to a neighboring cell, it only increases the distance to the closest pursuer by 1 (or 0 if it is trapped in a corner). A pursuer reduces its distance to the assigned evader by  $v_p$  in each time step. Assume the assigned pursuer to an AEE is actually the closest pursuer (usually it is the case in a greedy assignment algorithm), the position of AEE  $i$  is  $C_i$ , the position of pursuer  $j$  is  $C_j$ , and the corner the evader will be trapped in is  $C_c$ .

The expected time the evader reaches the corner (if it is not caught sooner) is given by

$$E(t) = \frac{d(C_i, C_c)}{1}. \quad (8)$$

At this point, the distance between the pursuer and the evader  $d(C'_i, C'_j)$  is

$$d(C'_i, C'_j) = d(C_i, C_j) - E(t) \times (v_p - 1). \quad (9)$$

The remaining capturing time only depends on this distance and the pursuer's speed  $v_p$ . Therefore, the overall expected capturing time is:

$$E(T) = E(t) + \frac{d(C'_i, C'_j)}{v_p}. \quad (10)$$

When  $v_p = 1$ , both evaders and pursuers move at the same speed, and the expected capturing time is

$$E(T) = d(C_i, C_c) + d(C_i, C_j). \quad (11)$$

Predicting the position of an AEE requires the knowledge of the pursuer moving history, which is easily accessible at the base station in the centralized system. In the distributed method, however, the prediction is performed on pursuers with incomplete knowledge of other pursuers. The prediction in the distributed system is thus less accurate than that in the centralized system. A possible improvement is to introduce a probabilistic inference about other pursuers' positions. We leave this for the future work.

### C. Evader Moving Pattern Recognition

We use a classification method based on the nearest neighbor algorithm to predict the evader moving pattern. We choose to perform the algorithm always in the distributed manner as follows, because performing it in the central base station does not present better accuracy and the distributed implementation is more efficient.

- If a sensor node detects an evader, it records the evader's moving direction and sends it in a package  $p$  to the sensor node to which the evader is heading.
- The new sensor node adds the new direction to the package  $p$  and sends it to the next sensor node. By relaying the data package, the sensor nodes build a moving history of the evader.
- After a pre-defined period  $t'$  or the evader is moving out of the connected subset, the sensor node currently holding the package  $p$  performs the classification algorithm.
- The algorithm uses two types of information from the package  $p$ . One is the longest  $k_1$  durations in which the evader maintains the same direction. The other is the largest  $k_2$  turning angle.
- After the evader class is identified, it is sent out with the evader id to the connected sensor nodes (and supernodes in the centralized system).

To raise the accuracy of the classification, we either take a longer observation period  $t'$  or use more dimensions (larger  $k_1$  and  $k_2$  values). However, we also want

to get the result quickly and reduce the complexity of the classification calculation. We further discuss the tradeoffs in Section VII.

## VI. PURSUER ALGORITHMS

### A. Target Selection

A pursuer at time  $t$  in cell  $C_m$  chooses its target by finding the maximal value of the heuristic function as

$$U_i = \alpha \cdot \frac{1}{d(C_i, C_m)} + \beta \cdot P(k, C_i, t), \quad (12)$$

where  $P(k, C_i, t)$  is the probability of every evader  $k$  in every cell  $C_i$ , and  $\alpha$  and  $\beta$  are the weights for two separate heuristic functions.

We set  $\alpha = \beta = 1$ , which makes the pursuers choose the cell with higher probability and takes into account the distance factor. Therefore, if there is a nearby cell with a slight lower probability, the pursuer will go to the nearby cell first.

A more precise heuristic function is the existence probability when the pursuer arrives at the target. For the same problem as Eq. (12), we have

$$U'_i = P(k, C_i, t'), \quad (13)$$

where

$$t' = t + \frac{\max(|x_i - x_m|, |y_i - y_m|)}{v_p}. \quad (14)$$

In the centralized system, the decision is made in the base station, whereas in the distributed system, it is made by individual pursuers based on their local knowledge.

### B. Coordination Among Pursuers

In the centralized system, there is no need for pursuer coordination because the base station assigns the tasks. To the contrary, in the distributed system, the coordination is necessary to eliminate the cases where two pursuers are pursuing the same target or a pursuer is pursuing an evader already captured by another pursuer.

A pursuer sends the coordination messages to the sensor node in the same cell, and the message is propagated to the connected sensor nodes. Every sensor node stores the message and sends the message to the pursuers who pass it.

*Locking a Target:* If two pursuers choose the same target, one of them wastes the trip and thus increases the overall time-to-capture. This usually happens when two pursuers are close to each other. In such a situation, they have similar information which results in similar probabilistic maps. Also, their distances to other cells are similar. According to Eq. (12), they will usually choose the same target.

A local coordination approach solves the problem effectively. A pursuer sends a message to lock the target it chooses. The message is propagated by the sensor network to other pursuers. If there is a conflict that two pursuers try to lock the same target, the one with higher id will get the priority.

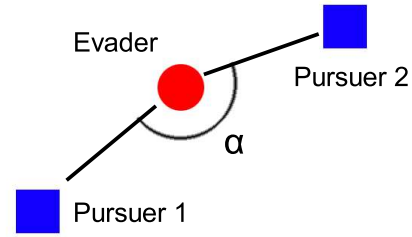


Fig. 4. Illustration of the pursuers' collaboration to capture an AEE evader. The angle  $\alpha$  is formed by the two pursuers and the evader. When  $\alpha > \frac{\pi}{2}$ , the collaboration is effective.

*Clearing off a Target:* When a pursuer captures an evader, it clears off the probability of the evader in its own memory and does not hunt for it any more. However, other unaware pursuers still look for the evader. To avoid this, a pursuer sends out a message to clear the captured evader off from the memories of other pursuers via the sensor nodes.

### C. Collaboration Among Pursuers

For different evader moving patterns, the collaboration strategies exhibit different performances. An RME's expected capturing time only depends on the distance to the assigned pursuer (Eq. (5)). An addition of pursuers will not give further improvement. However, for an AEE, an extra pursuer may reduce the overall capturing time effectively. If the extra pursuer is on the escaping direction of the AEE, every time an evader increases the distance to the nearest pursuer by 1, it also reduces the distance to the collaborating pursuer by 1. This is also the case when the angle formed by two pursuers and the evader  $\alpha > \frac{\pi}{2}$  (Fig. 4). In our experiments, we examined different cases and concluded that the collaboration in the case where two pursuers and one AEE form an angle larger than  $\frac{\pi}{2}$  give the most significant improvement (Table II).

TABLE II

IMPROVEMENT OF THE PURSUER COLLABORATION IN DIFFERENT CONDITIONS. THIS IS A 2-PURSUER-1-EVADER PROBLEM, IN A  $50 \times 50$  FIELD WITH  $v_p = 1$ , AND 30 RUNS FOR EACH CASE.

Evader Moving Pattern	$\alpha$	% of improvement
AEE	$> \frac{\pi}{2}$	39.3
AEE	$\leq \frac{\pi}{2}$	19.2
RME	$> \frac{\pi}{2}$	2.4
RME	$\leq \frac{\pi}{2}$	1.0

In the cases where there are more evaders than pursuers, the extra pursuers perform as helpers. However, if the number of the evaders is large, the collaboration wastes one pursuer, thus reducing the system performance. This is because the average improvement is less than 50% (in Table II). Therefore, the precondition that  $N_p > N_e$  has to be satisfied before the pursuers switch to the collaboration stage.

In the centralized system, pursuer collaboration is coordinated by the base station. In the distributed system,

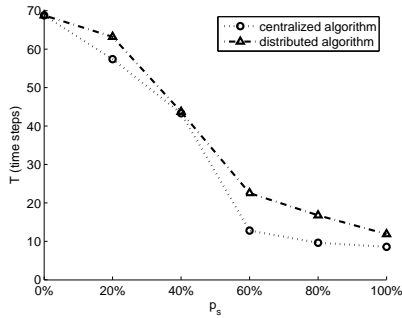


Fig. 5. Comparison of the centralized and distributed algorithms in different network conditions. The performance of pursuers, Time-to-capture  $T$  is plotted against the space coverage of the sensor network,  $p_s$

collaboration is coordinated using the same manner as the distributed coordination system we described above. The collaboration is local optimal because the coordination range of a pursuer is only within a connected subnet.

VII. EXPERIMENTAL RESULTS

A. Distributed Method v.s. Centralized Method

We first tested the performance of the distributed system and the centralized system in the same setups. We draw a  $20 \times 20$  field with different degree of sensor node coverage  $p_s$  ranging from 0 to 100%, which is defined as the ratio of the number of the cells with sensor nodes to the total number of cells. There were 3 pursuers and 9 RME evaders. The pursuer speed  $v_p = 2$ , and the capturing radius  $r_p = 1$ . We ran simulation 50 times for both systems in each of the 6 network coverages. The average time-to-capture is shown in Fig. 5.

We have the following observation from Fig. 5. Firstly, the centralized method always outperforms the distributed method because it achieves more accurate prediction on evader locations and performs more effectively in the pursuer collaborations. Secondly, in sensor networks with coverage less than 60%, the distributed method achieves performance close to the centralized method because the subnets are usually well separated and the local decisions by the pursuers are close to the decisions made by the central controller. Finally, in well connected sensor networks with coverage at least 60%, centralized method outperforms the distributed method significantly. Under such conditions, a large portion of the sensor nodes are usually connected, and the centralized decision are near optimal, because it uses the near global information. To the contrary, the local predictions by pursuers who are in disconnected regions have large uncertainties and the coordination is also not effective, because it does not communicate with other pursuers.

B. Accuracy on Evader Moving Pattern Recognition

Next we show that accuracy of the classification algorithm in identifying evader moving patterns. We set up a  $50 \times 50$  field with 100% degree of coverage. There were

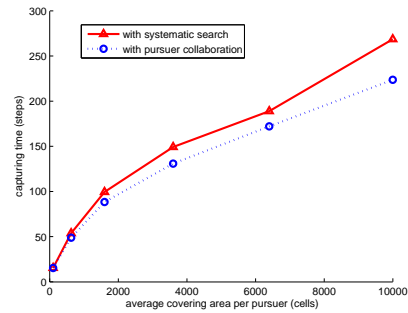


Fig. 7. The improvement of collaboration strategy over the systematic search for different network sizes.

3 pursuers and 8 evaders. The pursuers' speed  $v_p = 2$ . The capturing radius  $r_p = 1$ . We set the observation period  $t'$  to 10, 30, and 50. We tested different numbers of dimensions. We started with  $k_1 = 1$  and  $k_2 = 1$ . In each new try we increased both numbers by 1 and finally reached 20 dimensions ( $k_1 = 10, k_2 = 10$ ). We ran the simulation 30 times (240 single evader data points). We used the first 160 evader data points as the training data and the last 80 as the test data. Fig. 6 shows the accuracy on both training data and test data:

- The result on the training data is very accurate, which suggests that the difference between the two moving patterns is obvious. The result on the test data is worse but is still correct in about 75% of the cases.
- The accuracy increases with the number of dimensions. However in both training and test data, the accuracy does not show a significant improvement after 12 dimensions ( $k_1 = 6, k_2 = 6$ ), which is a good tradeoff between computation complexity and accuracy.
- As we have predicted, longer observation period yields better result, and the performance when  $t' = 30$  is obviously better than when  $t' = 10$ . However,  $t' = 50$  does not lead to a huge improvement compared with  $t' = 30$ . This suggests a good observation period of 30.

C. Effect of Collaboration

Finally we show the performance of the collaboration algorithm in the centralized system as an example. We compared our algorithm with the algorithm in [17] where extra pursuers systematically scan through the field based on a preset routine. We also tested the generality of the new collaboration strategy. In the same setup as the last set of experiments, we varied the density of the pursuers (normalized to the number of cells covered per pursuer) from 100 to 10,000. We change the pursuers' speed between 1 and 2 so that pursuers move either faster than or at the same speed as the evaders. In addition, we randomly set the evader moving patterns to AEE or RME so that the simulation reflects the condition of capture and rescue missions and the combined operations. The performance of the collaboration algorithm and the systematic search algorithm (SSA) is shown in Fig. 7.

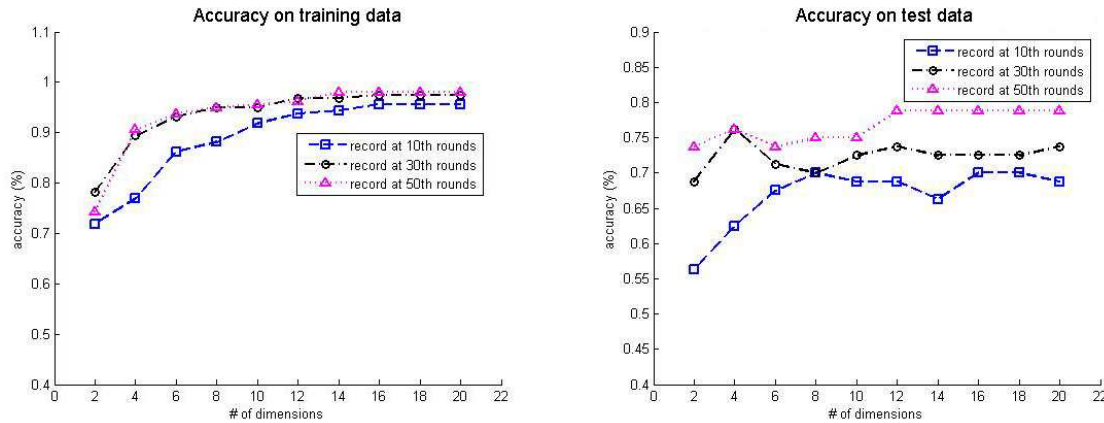


Fig. 6. Classification accuracy on training data (left) and test data (right). The x-axis is the number of dimensions, and the y-axis is the accuracy in percentage. The data is collected at 3 different observation period  $t' = 10, 30, 50$ .

From the figure, we observe that less dense pursuer deployment makes the mission more difficult. In those cases, the collaboration strategy yields more improvement over the systematic search (16.7% in the most sparse deployment). In denser deployments, the collaboration algorithm does not outperform SSA much because the systematically scanning pursuers have more chance to run into an evader.

This result shows that the collaboration algorithm is useful in the field applications where a small number of expensive pursuers (usually unmanned vehicles) are deployed to monitor a large field. With the assistance of a large number of inexpensive sensor nodes, the long-distance monitoring is feasible. In addition, the general collaboration algorithm assures that the capturing mission is efficient and adaptive to different field conditions.

### VIII. CONCLUSION AND FUTURE WORK

In this paper, we have presented new centralized and distributed methods for solving the pursuit-evasion problem (PEP) using sensor networks. We extend an existing centralized method with predicting evader locations based on different evader moving patterns, evader pattern recognitions, and pursuer coordinations and collaborations. We have proposed a new distribution method that are effective and efficient. We have shown with experimental results that our proposed algorithms are effective and have analyzed the tradeoffs between the distributed and centralized methods. Our future work includes studying the case with false detection and radio delay in sensor networks and the corresponding pursuer strategies.

### ACKNOWLEDGMENT

The work presented in this paper was partially supported by the National Science Foundation under grant CNS-0423386.

### REFERENCES

- [1] E. Alpaydin. *Introduction to Machine Learning*. MIT press, 2004.

- [2] B. Awerbuch and D. Peleg. Online tracking of mobile user. *Journal of the Association for Computing Machinery*, 42:1021–1058, 1995.
- [3] A. Bar-Noy and I. Kessler. Tracking mobile users in wireless communication networks. *INFOCOM*, pages 1232–1239, 1993.
- [4] R. R. Brooks, P. Ramanathan, and A. M. Sayeed. Distributed target classification and tracking in sensor networks. In *Proceedings Of The IEEE*, volume 91, Aug 2003.
- [5] M. Demirbas, A. Arora, and M. Gouda. A pursuer-evader game for sensor networks. The Sixth Symposium on Self-Stabilizing Systems, June 2003.
- [6] Y. He and E. K. P. Chong. Sensor scheduling for target tracking in sensor networks. In *43rd IEEE Conference on Decision and Control*, Dec 2004.
- [7] J. Hespanha, H. Kim, and S. Sastry. Multiple-agent probabilistic pursuit-evasion games. *IEEE International Conference on Decision and Control*, 1999.
- [8] H. Kim, R. Vidal, D. Shim, O. Shakernia, and S. Sastry. A hierarchical approach to probabilistic pursuit evasion games with unmanned ground and aerial vehicles. *IEEE International Conference on Decision and Control*, 2001.
- [9] S. Oh, S. Russell, and S. Sastry. Markov chain monte carlo data association for general multiple-target tracking problems. In *Proc. of the IEEE International Conference on Decision and Control*, December 2004.
- [10] S. Oh, L. Schenato, and S. Sastry. A hierarchical multiple-target tracking algorithm for sensor networks. In *Proc. of the International Conference on Robotics and Automation*, April 2005.
- [11] L. Schenato, S. Oh, and S. Sastry. Swarm coordination for pursuit evasion games using sensor networks. In *Proc. of the International Conference on Robotics and Automation*, April 2005.
- [12] B. Sinopoli, C. Sharp, S. Schaffert, L. Schenato, and S. Sastry. Distributed control applications within sensor networks. In *IEEE Proceedings Special Issue on Distributed Sensor Networks*, November 2003.
- [13] T. Vercauteren, D. Guo, and X. Wang. Joint multiple target tracking and classification in collaborative sensor networks. In *IEEE International Symposium on Information Theory (ISIT)*, Jun 2004.
- [14] R. Vidal, O. Shakernia, H. J. Kim, D. H. Shim, and S. Sastry. Probabilistic pursuit-evasion games: Theory, implementation, and experimental evaluation. *IEEE Transactions On Robotics And Automation*, 18(5):662–669, 2002.
- [15] J.-P. Wagner and R. Cristescu. Power control for target tracking in sensor networks. In *2005 Conference on Information Sciences and Systems*, Mar 2005.
- [16] G. Zhou, T. He, S. Krishnamurthy, and J. A. Stankovic. Impact of radio irregularity on wireless sensor networks. *MobiSYS04*, June 2004.
- [17] P. Zhuang, Y. Shang, and H. Shi. A new distributed planning method for pursuit-evasion game using sensor networks. the 2nd IEEE International Workshop on Adaptive Wireless Networks (AWiN), November 2005.