A Survey of Sensor Selection Schemes in Wireless Sensor Networks

Hosam Rowaihy*, Sharanya Eswaran*, Matthew Johnson†, Dinesh Verma‡, Amotz Bar-Noy†, Theodore Brown† and Thomas La Porta*
*Department of Computer Science and Engineering
Pennsylvania State University
†Deparment of Computer Sience
City University of New York
‡IBM T. J. Watson Research Center

***Abstract*— One of the main goals of many sensor networks is to provide accurate information about a sensing field for an extended period of time. This requires collecting measurement from as many sensors as possible to have a better view of the sensor surroundings. However, due to energy limitations and to prolong the network lifetime the number of active sensors should be kept to a minimum. To resolve this conflict of interest, sensor selection schemes are used. In this paper, we survey different schemes that are used to select sensors. Based on the purpose of selection, we classify the schemes into (1) coverage schemes, (2) target tracking and localization schemes, (3) single task assignment schemes and (4) multiple task assignment schemes. We also look at solutions to relevant problems from other areas and consider their applicability to sensor networks. Finally, we take a look at the open research problems in this field.**

I. Introduction

Sensor networks consist of a large number of small sensor devices that have the capability to take various measurements of their environment. These measurements can include seismic, acoustic, magnetic, IR and video information. Each of these devices is equipped with a small processor and wireless communication antenna and is powered by a battery making it very resource constrained. To be used, sensors are scattered around a sensing field to collect information about their surroundings. For example, sensors can be used in a battlefield to gather information about enemy troops, detect events such as explosions, and track and localize targets. Upon deployment in a field, they form an ad hoc network and communicate with each other and with data processing centers.

Sensor networks are usually intended to last for long periods of time, such as months or even years. However, due to the limited energy available on board, if a sensor remains active continuously, its energy will be depleted quickly leading to its death. To prolong the network lifetime, sensors alternate between being active and sleeping. There are several sensor selection algorithms to achieve this.

The decision as to which sensor should be activated depends on a variety of factors depending on the algorithms: residual energy, required coverage, or the type of information required. Sensors are selected to do one or multiple missions. These missions can can be general and related to the function of the network, such as monitoring the whole field by ensuring complete coverage, or more specific and application-oriented, such as tracking a target's movement. At a given time, the system might be required to do multiple missions such as monitoring an event and, at the same time, track a single or multiple moving targets.

In this paper, we survey the recently proposed schemes used for sensor selection. These schemes can be categorized in many different ways. For example, they can be centralized (all processing is done by a single node) or distributed (processing cost is divided upon many nodes). They can also be classified according to the criteria or parameters that decide the selection process, e.g., physical attributes-based, task utility-based, information gain-based, etc. In this paper we categorize the schemes based on the purpose of selection, into the following classes:

1) *Coverage schemes*: includes selection schemes that are used to ensure coverage.
2) *Target tracking and localization schemes*: includes schemes that select sensors for target tracking and localization purposes.
3) *Single mission assignment schemes*: includes schemes that select sensors for a single specific mission.
4) *Multiple mission assignment schemes*: includes schemes that select sensors so that multiple specific missions are collectively accomplished.

The rest of this paper is organized as follows. In Section II we define the sensor selection problem. In Sections III-VI we discuss the different classes of schemes. We present some theoretical approaches as they apply to sensor selection and assignment of sensors to missions in Section VII. In Section VIII, we define open research problems in this area. Finally, Section IX concludes the paper.

II. Sensor Selection Problem

The sensor selection problem can be defined as follows: Given a set of sensors $\mathcal{S} = \{S_1, ..., S_n\}$, determine the "*best subset*" S' of k sensors to satisfy the requirements of one or multiple missions. The "best set" is one which achieves a tradeoff of energy constraints and quality of information with respect to its task.

So, we have two conflicting goals: (1) to collect information of high quality and (2) to conserve energy. This trade-off is usually modeled using the notions of *utility* and *cost*:

1) **Utility:** accuracy of the gathered information and its usefulness to a mission.
2) **Cost:** the energy expended activating and operating the sensors and any other cost associated with selection. This is directly proportional to number of selected sensors k.

The goal of a sensor selection scheme is to select k sensors such that the total utility is maximized while the overall cost is less than a certain budget. In most cases, this problem becomes equivalent to the Knapsack problem which is known to be NP-complete [13]. This means that there is no solution that can run in polynomial time (in number of sensors). This is clearly not desirable, especially if we consider a network with large number of sensors. Hence, approximation and heuristics are mostly used to solve this problem. For example, in [14] the authors assume that for a scheme that selects subset S' of k sensors, the cost associated with sensors is zero if $|S'| \leq k$ and infinity otherwise. In the following sections we examine different sensor selection schemes.

III. Coverage Schemes

In this section, we discuss schemes in which sensors are selected in order to ensure coverage. We look at schemes that assume static sensors then we look at selection schemes for mobile sensors.

A. Selection for Static Nodes

If the sensors are deployed densely, such that there is no coverage hole and there is redundancy in coverage, then only a subset of sensors must be active, while the rest enter sleep mode. This conserves energy and hence prolongs the network lifetime. Selection schemes are used to decide which sensors are to be turned on and for how long. If, during the course of operation, a node fails, resulting in a coverage hole, the selection protocols can be rerun to activate one or more of the redundant nodes to restore the coverage. In [29], [7], [8], [31], [35], [22] sensor selection schemes to ensure coverage using only a few static sensors are discussed.

In [29], the sensor nodes are divided into sets, such that each set is capable of providing complete coverage of the field and only one set is active at a time. The objective is to obtain an optimal scheduling of these sets (i.e., to optimally select which set is active when and for how long) such that the lifetime of the application is maximized and a minimum level of quality of service constraints (such as bandwidth, coverage and energy constraints) are met. This problem is formulated as a generalized maximum flow graph and an optimal solution is found through linear programming (LP).

A similar approach is used in [7] and [8]. In [7] the sensors in the field are divided into disjoint sets, such that every set completely covers all the targets and only one set is active at a time. Unlike [29], here the sets are scheduled in a round-robin order and the focus is on the problem of finding the maximum number of disjoint sets. As the number of sets increase, the efficiency becomes greater because the time interval between two activations for any sensor is longer. Cardei, et al [7] prove that finding this maximum number of disjoint sets ("Disjoint Set Cover problem" or DSC) is NP-complete and provide a heuristic-based approximate solution. The DSC problem is transformed into a max-flow problem, which is formulated as a *Mixed Integer Programming* (MIP) instance. The output of the MIP is used to compute the disjoint set covers in polynomial time.

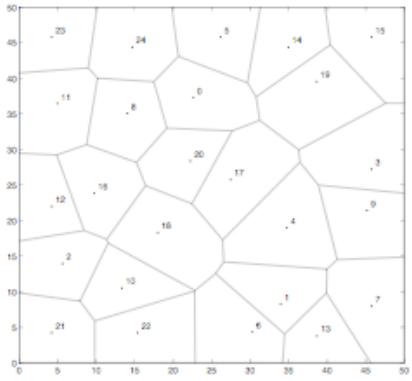

Fig. 1. An example of a Voronoi diagram [33].

Both [7] and [8] illustrate the tradeoff between coverage and bandwidth requirements. Each sensor node in the set has to send its observation to a data sink (base station). So the maximum number of sensors sending simultaneously is restricted by bandwidth and the minimum of sensors is determined by the "complete coverage" requirement. While [7] compromises on the former, [8] compromises on the latter. The objective of [8] is to minimize the breach of coverage and the set selection is restricted by bandwidth constraints. This is formulated as a 0-1 Integer Programming problem and two heuristics are provided based on this formulation.

The main drawback in [29], [7], [8] is that it is difficult to implement this approach in a distributed manner, which is more desirable in a sensor network environment.

In [31], full coverage with minimal sensors is obtained by identifying the redundant sensors and turning them off. Identification of redundant sensors is done using Voronoi diagrams [3].

Voronoi diagrams are a common and useful way of representing the the physical location of the sensors and the coverage thereby provided. A Voronoi diagram divides a plane that includes a number of sensors into polygons such that each polygon contains exactly one sensor and every point inside the polygon is closer to the contained sensor than to any other. Figure 1 shows an example of a Voronoi Diagram.

The algorithms presented in [31] make use of the properties of Voronoi diagrams to find appropriate sensors to activate. These algorithms account for the residual energy of sensors to balance the power consumption and prolong the network lifetime. Initially, all sensor nodes are inactive. An initiator node finds its Voronoi diagrams assuming different neighboring nodes are activated. It then chooses the configuration (i.e. the active node set) that makes the area of its Voronoi cell less than or equal to its sensing area. Finding such an optimal set of neighbors is NP-complete, and hence the authors propose an approximation algorithm which requires $O(n^2)$ complexity.

In [22], the authors aim to provide k-coverage, which means that every point in the field is covered by at least k sensors. This improves the fault-tolerance of the network. Initially, all sensors are inactive. Before turning on, each sensor waits for a back-off period determined by the amount of contribution they can provide for the coverage. Thus, the sensors are turned on one by one in a greedy fashion, with the sensor with most "contribution" turning on first, then the next one, and so on. The "contribution" or the "coverage merit" is computed based on the probability of detection of an event by that sensor within its sensing area.

[35] provides a "self-scheduling" scheme, in which time of operation is the only parameter in the selection process. Here, the nodes dynamically schedule themselves while guaranteeing a certain degree of coverage. The sensors are time-synchronized, and each sensor generates a random reference time which is exchanged with its neighbors. Each sensor then establishes its sleep-awake cycle by observing the reference time of its neighbors. Differential coverage is possible by adjusting the node's schedule parameters. If an event requires more coverage, just this information is disseminated and the nodes adjust their schedules locally such that there is more overlap, i.e., more sensors stay awake at a time.

B. Selection for Mobile Nodes

If nodes are mobile (e.g. placed on robot), more options arise, because a node can now be moved to a new location in order to cover a hole. This makes deployment easier, because even if the random deployment results in incomplete coverage, the nodes can then relocate to ensure coverage. Similarly, if a node fails, then the network can be dynamically reconfigured. There are also new challenges posed by mobile nodes for selection schemes because the decision of which node to move has additional energy cost due to movement.

In [20], the objective is to optimally utilize the limited-range sensor capability and limited-travel capability of a group of mobile sensors (robots) that have to cover a region completely, after an initial random deployment. The field is modeled as a grid and the problem is formulated as an assignment problem (assigning sensor to grid point). Two variations of this problem exist: (i) to minimize the maximum distance traveled by a sensor and (ii) to minimize the total distance traveled subject to energy (fuel) constraints. The first is an instance of the maximum cardinality (unweighted) matching problem on a bipartite graph (which connects the robots to grid points). The second is solved by computing a minimum-weight matching on the graph. We discuss the details of solving the sensor assignment problem using techniques such as matching in Section VII.

[30], [32] and [33] discuss selection schemes, where the problem is to decide which sensor to move in order to compensate for incomplete coverage.

In [30] four different schemes, based on different heuristics, are evaluated. The first scheme selects the node with the maximum residual energy. This avoids repeated failures at the same point. The second scheme tries to minimize the migration distance of a node. For each neighbor of the dead node, the maximum distance that it can move (which can vary from 0 to completely reaching the dead node's position itself) is calculated. The node which has to move the minimum of these maximum distances is selected. This ensures that the maximum area can be recovered, while the migration distance is minimum. The third scheme combines the first two heuristics and selects the node with the minimum value of the ratio (maximum distance to be moved) / (residual energy). While these three schemes move nodes the maximum possible distance towards the dead node (greedy approach), in the fourth scheme, the minimum distance each node has to move is calculated and the node with the minimum of these min-distances is selected, just so that the uncovered area is likely to be covered.

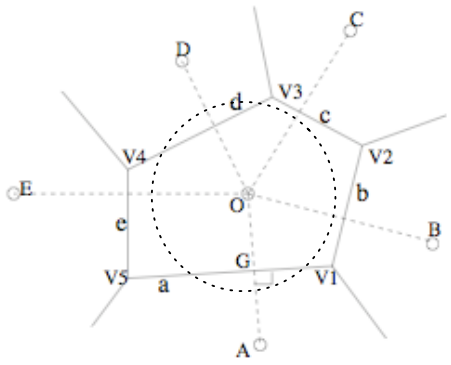

Fig. 2. Example of a coverage hole. Node O's Voronoi polygon is greater in size than its coverage area (denoted by dotted circle) [33].

In [33] a bidding protocol is used for deciding which sensors to move to cover the holes. A mixed network of static and mobile sensors is deployed with mobile sensors being redundant initially. Static sensors then discover coverage holes in the field. Coverage holes are discovered using Voronoi diagrams (Figure 2 illustrates how this can be done). Each mobile sensor has a base price for serving one hole in the sensing field. The price is related to the size of any new hole generated by its movement. The static sensors in the network estimate the size of the coverage holes, and place bids for the mobile sensors accordingly. The mobile sensors choose the highest bids and move to heal the largest coverage holes.

IV. Target Tracking and Localization Schemes

In this section we discuss the methods in which sensors are selected for the purposes of target tracking and target localization. These schemes can be further classified based on the approach used in the selection algorithm. For example, we consider three categories, (i) Entropy-based solutions, where the selection schemes aim to minimize the entropy of measurement, (ii) Mean Squared Error-based solutions, where the aim is to minimize the mean squared error of measurement and (iii) Dynamic information-driven solution, where the aim is to maximize the information gain based on the dynamic information gathered.

A. Entropy-based Solutions

Entropy refers to a measure of uncertainty. The lesser the entropy of some measurement, the more we can be certain of its accuracy. There are several solutions that use this concept. In both [10] and [21], the authors use the mutual information about the future state and the current node measurement to determine the information gain of the different sensors. A greedy approach is used to solve the sensor selection problem for target localization and tracking. At each round, an unused sensor with an observation that is expected to yield the maximum entropy reduction of the target location distribution is selected. This newly added observation is then used to determine the target location distribution using recursive Bayesian estimation techniques [4]. Sensor selection continues in this manner until the level of entropy of the target location distribution is less than or equal to a predefined value. This value reflects the desired confidence that an application needs regarding the target's location. The goal here is to reach the required entropy level without using more sensors than necessary.

The major problem of this scheme is that it requires a way to effectively evaluate the expected entropy reduction for the different candidate sensors. This is difficult to determine without actually retrieving measurement data from the different sensors. Also, this scheme is centralized in the sense that sensor selection decisions are made by a single node. This is not scalable and incurs a high communication overhead which makes it impractical in many scenarios.

In [34], the authors consider sensor selection for target localization and develop a solution based on entropy. However, instead of trying to find the optimal solution using mutual information, which is computationally intensive, it proposes a heuristic that provides a sub-optimal solution but is more efficient. Given a prior probability distribution of the target location and the locations and sensing models of a set of sensors, the method selects an informative sensor such that the collection of the selected sensor observation with the prior target location distribution results in the greatest reduction in uncertainty. The proposed heuristic adds one sensor at a time to reduce the entropy of the target location distribution. Although, this solution is more efficient than [21], [10], it is still centralized which limits its scalability.

B. Dynamic Information-driven Solutions

In [36], the authors propose a sensor selection scheme that is used for target tracking. They consider the problem of selecting a sensor j, such that j provides the greatest improvement in the estimate of a target location at the lowest cost. This is solved as an optimization problem defined in terms of information gain and cost. The goal is to improve: (1) detection quality, (2) track quality, (3) scalability, (4) survivability and (5) resource usage.

The proposed scheme selects a single sensor (*leader*), that becomes active. The initial selection is made by considering the predicted position of the target. From that point on, after collecting the required measurements about the target, the leader selects the next node that it believes to be the most informative and passes its measurements to it. That node becomes the new leader. This continues as long as needed to track a target.

When deciding on the next leader, the current leader considers the information utility value of candidate sensors. To be practical, this value must be based only on available information such as a sensor's location, its modality and the current belief state. The authors consider two possible definitions of information utility; one based on entropy and another based on distance measure. Although, an entropy based definition is mathematically more precise, it is very difficult to compute in a practical setting. This is because it requires knowing a sensor's measurement before making any decision, which is clearly very difficult. With distance based measure, the leader node measures the utility of other sensors based on how far are they located from the estimated target position. This provides a good approximation of the sensor's utility.

One of the drawbacks of this scheme is that its accuracy depends on the quality of the choice of the first leader. If the first leader is not close to the target location, due to error in prediction, the overall tracking quality will degrade and the whole process might even fail. Also, this scheme selects a single sensor (leader) at a time, so although it may be energy efficient, it might not provide information that is as good as if more sensors are used.

A similar scheme is proposed in [26]. The authors look at the problem of tracking in video-based sensor networks. The main difference from [36] is that this paper considers the initialization cost (measured in time) that sensors encounter when performing motion segmentation for target tracking. Instead of activating a single sensor at each time step, a set of sensors is selected such that their total information utility is maximized subject to an average energy constraint.

At each time step, the sensor network will have a processing node active (similar to the leader in [36]). This node is the closest node to the predicted target location. The processing node then determines the state of relevant sensors (i.e. active or not) depending on the total information-gain of the set and the energy constraint. After collecting data about a target and having a better estimate of its location, the processing node selects the next processing node from a set of candidates. The new processing node must be closer to the target than the current one. This scheme places a high overhead on the processing nodes.

C. Mean Squared Error-based Solutions

Minimizing mean squared error (MSE) is another approach that is commonly used. Kaplan [17], [18] discusses several schemes that achieve this in sensor networks for target localization. The system that is considered in these two papers consists of sensors that can estimate the direction of arrival (DOA) of a target using acoustic properties.

[17] present a global node selection scheme in which the location of all sensors is used to determine the set of active sensors. Initially, two sensors are selected as the active set. These two sensors must not be collinear with the target. Selecting the first active set is done by trying all the combinations of sensors and hence is $O(n^2)$. After selecting

the initial active set, sensors are added one at a time using a greedy approach. The goal is to minimize the MSE of the target's position. The paper also discusses methods that can be used to improve on the active set by continuously checking if replacing an active node with an inactive one will result in an improvement in localization quality. Because this solution requires global knowledge of the sensors location, this information must be delivered to all sensors. This can be done either by broadcasting this information to all nodes or by using multi-hop routing. Both of these schemes are inefficient in terms of energy and hence this solution can only be used in small-sized networks.

In [18], the author overcomes the limitation of the previous scheme by presenting a local node selection scheme called *autonomous node selection* (ANS). In this scheme, the decision on which a node is selected is based only on local information about the usefulness of that node relative to the current active set. The usefulness is measured by considering the MSE. The scheme starts by performing an exhaustive search to determine the initial set of active sensors. This is similar to the scheme used in [17]. After selecting the initial active set, ANS is used in each node that is within the communication range of the active set. At each iteration, active nodes from previous iteration select k sensors (where k is the required number of sensors) from a list of candidates. The candidate list begin with the initial set but then grow to include previously inactive nodes. The growth of the candidate set is done by calculating the differential utility of all nodes in the active set then determining a threshold value for an inactive set to join the of candidate nodes.

Both of the schemes discussed above incur a very high computational overhead especially when performing the exhaustive search to determine the initial set of active sensors which limit their applicability to small networks. They also require the exchange of a large number of messages to find a solution which again affect their scalability.

V. Single Mission Assignment Schemes

In a sensor network which must perform a specific mission repeatedly over time, sensors need to be selected such that the mission is accomplished in the most efficient manner. The objective of such selection schemes is to select the sensor nodes that are most useful for the mission. This notion of "usefulness" is quantified using a "utility" value. Th papers we discuss here differ from those discussed in sections III and IV in that they use sensor selection for the "application layer", while those in the previous two sections use selection to perform the basic functionalities of a sensor network (namely, coverage and tracking) more efficiently.

In [6], the authors develop a model for such applications in which the global objectives are defined based on utility functions and a cost model for energy consumption due to sensing and data delivery. In their algorithm, a set of sensors has a total utility function that depend on the number of individual sensors and their placement.

The authors of [5] provide a generic framework in which the application can specify the utility values of the sensors when the goal is to select a sequence of sets of sensors so that the total utility is maximized, while not exceeding the available energy. Alternatively, the framework can be used to look for the most cost-effective sensor set, maximizing the product of utility and sensor lifetime.

VI. Multiple Mission Assignment Schemes

The methods discussed in this section deal with a multiple-mission scenario. It must be noted that, as in Section V, by mission, we refer to a specific (application-level) mission and not a network level function, such as coverage or data dissemination.

[2] provides a greedy heuristic to cover the maximum number of targets with the minimum number of active sensors. The sensors here are directional and covering each target can be viewed as a different mission. After the initial random deployment, not all the targets are covered. Hence, the objective is to change the initial orientations of the directional sensors in order to cover as many targets as possible, while activating as few sensors as possible. This can be formulated as a *Integer Linear Programming* (ILP) problem with the number of sensors (n), number of targets (m) and number of orientations available for a sensor (p) as parameters. [2] prove this problem to be NP-complete. A greedy heuristic-based solution is provided. The algorithm is briefly described here.

Initially, all the sensors are inactive. In each subsequent iteration, for each sensor that has not yet been activated, the number of additional targets that would change from uncovered to covered for each possible configuration is calculated. Then, the inactive sensor that maximizes the number of newly covered targets in a particular orientation is activated in that orientation. Thus, the number of targets covered is maximized greedily. A distributed implementation of this approach is also provided in [2].

[24] models the system as a market and explores the advantages of incorporating e-commerce concepts to sensor management. In e-commerce, the customer wants to essentially drive the process, while the conventional sensor management follows a much less capitalistic approach, producing information "goods" based on pre-defined system goals and priorities [24]. The two main components in this model are the mission manager and sensor manager, which are implemented using genetic algorithms (GA). The mission manager allocates budgets to the application (consumer), based on the different missions involved in the application and their requirements. Using these budget values, the consumer places bids to the sensor manager. Based on these bids, the sensor manager allocates sensors to the missions. This is a centralized system and it is very difficult to implement it in a distributed way because of the complexity of the model and the GA computation involved.

In [25], multi-modal sensors are used and multiple missions may arrive simultaneously. The possible sensor configurations (which sensor operates in which mode) and the mission utility value, for each mission, are translated into a bid. The winner is determined using a modified combinatorial auction algorithm and thus a sensor is allocated to the mission for which it is responsible, and the mode in which it has to operate.

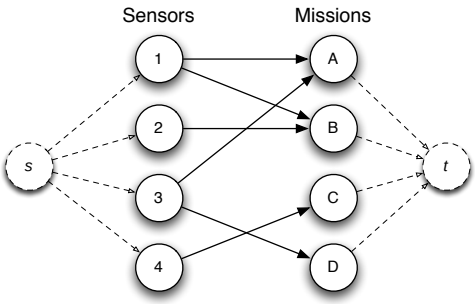

Fig. 3. Example of a bipartite graph. Dotted portion shows conversion to flow.

We can also consider the bidding scheme presented in [33] as a multiple-mission assignment scheme. In this case covering each hole is a single mission. In the same manner, the scheme that is proposed in [20] can be looked at as a multiple-mission assignment. However, the missions in this case are the grid points to which the robots are assigned.

VII. Theoretical Approaches to Sensor Assignment

We observe that when there are multiple missions required of a sensor network need to do, sensors must not just be selected but assigned to particular missions. A subproblem of the general Sensor Selection Problem is the *Sensor Assignment Problem*. This problem may be informally defined as assigning sensors to missions in the "best" way, which may depend on the cost of using individual sensors and the requirements of individual missions.

Somewhat more formally, we are given a set of sensors $\mathcal{S} = \{S_1, ..., S_m\}$ and a set of missions $\mathcal{M} = \{M_1, ..., M_n\}$. We may draw this as a bipartite graph, where an edge (S_i, M_j) will correspond to a pairing of sensor S_i with mission M_j. This will in general not be a complete graph, since each sensor may be applicable to only some missions. Each edge (S_i, M_j) may be associated with a utility or cost value. See Figure 3 for an example of this setup. Our goal is to assign sensors to missions optimally (maximizing utility and/or minimizing cost), subject to the constraints we impose. This is not yet a well defined problem, but rather an informal motivation for the specific optimization problems we look at below.

In this section, we will define several particular problems with this motivation, using tools from network flow theory. We survey several classical problems and algorithms, as well as recently studied problems from other domains, and adapt them to the setting of sensor assignment. These problems are different versions of network flow, semi-matching, online matching, or combinations of these.

A. Flow-based Problems

Maximum flow is used to model many kinds of optimization problems that involve the movement of resources from a source location to a target location. We now consider the graph representation of the sensor assignment problem presented above and reduce it to a max-flow problem, since by finding the max-flow of such a graph, we can determine the best possible sensor assignment.

1) Max-Flow: Max-flow models the problem of determining how much of a resource can pass from a source to a destination, given the capacities of each segment of transport along the way. More precisely, we are given a directed graph $G = (V, E)$, where V is a set of vertices and E is a set of edges. The vertex set contains two special nodes, the source node *s* and the sink node *t*. Each edge is associated with a non-negative integral capacity which is the maximum amount of resource that may pass through it (e.g., the size of the pipe). A flow can consist of a collection of paths. Because edges may in general have different capacities, the amount of flow along any path is limited by the minimum capacity of the edges along that path. Moreover, it may not be possible to use all edges' capacities even if all edges have the same capacity, since a vertex may have higher bounds on its in-flow than on its out-flow. Given these constraints, we seek to find the setup and amount of the maximum flow.

The classical Ford-Fulkerson algorithm [12] solves max-flow in polynomial time. The algorithm starts with no paths that belong to the flow and then proceeds as follows: as long as there exists in the graph a path from s to t that can be added to the flow (i.e. still has available capacity), increase the usage of the entire path as high as possible. When there are no longer any paths, stop. Crucially, the max-flow returned by Ford-Fulkerson is guaranteed to be integral, i.e., the flow through every edge is an integer.

In the standard max-flow problem, each edge's usage has an upper bound (its capacity) but no lower bound (apart from 0). For some applications, lower bounds can be incorporated. The strategy of Ford-Fulkerson is to begin with a *feasible* solution (all usages set to 0), and then iteratively improve this until we reach an *optimal* solution. We can't simply apply Ford-Fulkerson to an instance with lower bounds, since the all-0 assignment may not be feasible. In [11], the author presents a classical solution: Given a graph with upper and lower bounds on the edges, we transform it into a conventional max-flow graph. This graph has only upper bounds, but they will differ from the upper bounds of the original. The all-0 assignment is feasible for the new graph, which is then converted back to a feasible assignment for the original graph. So, we can simply run Ford-Fulkerson with this assignment as our starting point.

Max-flow models can by useful in solving the sensor assignment problem. Consider the following formulation. We are given the sets of sensors and missions. Each sensor is applicable to some, but not all, of the missions. Each sensor is associated with a total utility U_i it has to contribute. Associated with each mission M_j is a demand D_j, indicating the total utility the mission requires. A sensor may be assigned to multiple missions, distributing its utility fractionally, as long as its upper bound U_i is observed; a mission may have multiple sensors assigned to it, as long as its lower bound M_j is met. What we seek in the feasibility problem is an assignment of each sensor to 0, 1, or more missions, while respecting both sets of constraints.

This problem can be solved efficiently by reduction to max-flow in at least two ways. In both methods, we create a flow-

graph by adding two nodes, s and t, as the source and sink (see the dotted part of Figure 3); the capacity of each edge (s, S_i) is set to U_i. For any pairing (S_i, M_j), the capacity of the corresponding edge is infinity if this pairing is allowed and 0 if not. The way the flow graphs differ is in edges directed towards t. One option is to set the lower bound of (M_j, t) to D_j, and then solve max-flow with lower bounds. A second option is to set the capacity of (M_j, t) to D_j, and then solve conventional Max-Flow. If a feasible solution exists, then Max-Flow will necessarily find it, since it will be impossible to send superfluous flow to any mission.

2) Min-Cost Max-Flow: Given a flow graph where each edge has a capacity and a per-unit cost, there will in general be many different maximum flows. The Min-Cost Max-Flow problem is to find a maximum flow of minimum total cost. There are classical algorithms to find integral solutions to this problem in polynomial time [1].

The *transportation problem* (a.k.a. the Hitchcock Problem) is a problem that models the min-cost transportation of goods [28]. We are given a weighted bipartite graph of sources and terminals. Edge weight w_{ij} is the unit cost for transportation of goods from S_i to M_j. Each source has a supply P_i and each terminal has a demand D_j. We seek a min-cost way to satisfy all demands using the available supplies. This problem easily reduces to Min-Cost Max-Flow: add s and t to the graph; the capacity of each edge (s, S_i) is set to P_i, and the capacity of each edge (M_i, t) is set to its (integral) demand D_j.

We now describe a more sophisticated model for sensor assignment. Here, we require unique sensor assignments (i.e. a sensor can only be assigned to a single mission), but relax the nature of the mission demands to become simply the number of sensors that the mission needs. Each possible sensor-mission pair has a utility value. The feasibility problem is to assign sensors uniquely to missions so that each mission receives the number of sensors it demands. The optimization problem is to find a feasible assignment that maximizes the sum utility. This formulation is simply the Transportation Problem, with all source supplies equal to 1. Notice that the uniqueness aspect does not decrease nor increase the difficulty of the problem.

B. Matching and Semi-Matching

Now we discuss different ways of representing sensor assignment as a bipartite matching problem, whose two node sets will be interpreted as sensors and missions. We seek a matching of the two node sets of largest cardinality, or of largest weight. In the semi-matching problem, we allow nodes on one side to be used multiple times. This is appropriate here because usually multiple sensors may be assigned to the same mission, but not vice versa.

The *bipartite cardinality matching problem* can be defined as follows: given a bipartite graph, with node sets A and B, we seek a maximum matching in the graph, i.e., a subset of edges of maximum cardinality, under the restriction that no two chosen edges share an end-point. Although there are more sophisticated algorithms [1], one simple way of solving this problem in polynomial time is by reduction to Max-Flow.

To transform a bipartite matching instance to max-flow, simply add a new node s pointing to each member of A, and a new node t pointed to by each member of B. Set the capacity of every present edge to 1. Now, run a max-flow algorithm which will produce an integral solution. Since all edge capacities are 1, we can read off the matching from the resulting edge flows.

In the *Weighted Bipartite Matching*, we are given a bipartite graph of two node sets A and B with non-negative weights on its edges. We seek a max-weight matching i.e., a subset of edges of maximum combined weight, again under the restriction that no two chosen edges share an end-point. Finding a *maximum cardinality matching of minimum weight* is essentially the same problem; we can introduce extra nodes so that A and B are equal in size, and we can set the weights of all missing edges to 0. Since weights are non-negative, every max-weight matching will be a perfect matching. By replacing edge weight w_{ij} with edge cost $c_{ij} = W - w_{ij}$, where W is the highest value of all edge weights, we get a minimization problem.

Bipartite *Semi-Matching* is similar to ordinary bipartite matching, except the matching constraint is relaxed on one side: we now seek a subset of edges such that no two chosen edges share an endpoint in A. The problem can be formulated for both cardinality and weighted settings, and can be solved through similar reductions to max-flow, as above.

Both the weighted and unweighted matching models can be applied to sensor networks. Given a bipartite graph of sensors and missions, the presence of an edge (S_i, M_j) indicates that S_i is capable of serving M_j. In a feasible solution, each sensor is assigned to at most one mission. We seek a feasible solution of maximum cardinality. This formulation reduces immediately to cardinality semi-matching, but it is a very coarse model of our problem.

A weighted semi-matching model can also be used to solve the sensor assignment problem. In this case, given a weighted bipartite graph of sensors and missions, each edge (S_i, m_j) is associated with a corresponding a utility value which can be a measure of how useful a sensor is to a mission. The motivation here is that each mission wants to be covered, and different sensors may be capable of covering it, but some might get better quality of information for a mission than others. Each sensor can be assigned to at most one mission. We seek a feasible solution of maximum weight. This formulation reduces immediately to weighted semi-matching, but again it is a very coarse model of our problem. Both the weighted and unweighted versions matching problems were studied in [20] in the context of sensor assignment.

C. Online Matching and Semi-Matching

In the two previous sections, we took for granted that the entire set of missions is given at once, and solve this problem in an offline fashion. However, in real-life deployments, missions will arrive at different points of time. In this section, we study a sequential, online model. Each time a mission arrives, we must assign sensors to it, without knowledge of future missions. These decisions are irrevocable, meaning that once a sensor

is assigned, it cannot be reassigned to any future mission. In alternative models, we can allow for sensor reassignments or impose a cost on them, but these are not considered here.

1) Online versions of matching problems: Karp, Vazirani, and Vazirani [19] study the online version of unweighted bipartite matching and give a simple randomized algorithm with competitive ratio $1 - 1/e$, or about 0.63. The input is a bipartite graph whose nodes are partitioned into a set of servers and a set of requests. The algorithm initially selects a random ordering of the k server nodes, out of all possible $k!$ orderings. Each request is simply assigned to its highest-ranked available server.

Kalyanasundaram and Pruhs [16] study the online unweighted version of the *b-Matching Problem*, and present an optimal deterministic algorithm. In the b-Matching Problem, we seek a matching on a bipartite graph containing servers and requests. An edge indicates that a server is capable of serving a request. Each request should be assigned to a server; each server can at most serve b requests. In the online version, we receive requests (along with all their edges) one at a time; each request must immediately be assigned to a server or declined service.

[16] gives a simple algorithm, BALANCE, whose competitive ratio is $1 - 1/(1 + 1/b)^b$, or close to $1 - 1/e$ for large b. For each request r, BALANCE selects a server, among those adjacent to r, which is the least used so far. The usage levels of the servers is kept in balance, and hence the name.

One way to adapt this problem to the context of sensors and missions is to assume that each sensor can serve at most b missions at once, corresponding perhaps to b different sensor modalities. Given this restriction, we then seek a maximal matching in the setting where missions arrive online.

Kalyanasundaram and Pruhs [15] study weighted versions of the online matching problem, which they call Online Min-Matching (minimizing edge weight) and Online Max-Matching (maximizing edge weight). Because both of these are hard in the general case, they restrict themselves to metric spaces which satisfy the triangle equality. An edge weight is then construed as the distance between a server (site) and a request. We wish to match k servers with r requests so as to minimize the total distance traveled. One metaphor they give is a matching between (simultaneous) fires and fire stations. Min-Matching turns out to be quite hard. [15] gives an optimal competitive algorithm, called PERMUTATION, whose competitive ratio is $\frac{1}{2k-1}$, for an instance with *2k* nodes.

For the Max-Matching problem, they give an optimal greedy algorithm with competitive ratio of $1/3$ or 0.33. Since the triangle inequality will be violated by sensor-target bipartite graphs which include some zero-utility edges, it's not clear how adaptable Max-Matching is to our setting. Although sensor networks exist in the physical (and metric) world, and the utility of a sensor to a target is partly based on distance, there presumably are other factors that influence this as well, such as the modality of information sought.

2) AdWords Problem: Mehta, Aberi, Vazirani and Vazirani [23] define the *Adwords Problem*, which models Google's AdWords market and seeks to maximize total profit earned from potential advertisers. When a user enters a query, search results are displayed in response to it, as well as targeted ads. Potential advertisers place bids for different adwords, and the search engine must decide which bid(s) to accept in each case. Advertisers are charged their bid amount only if (and each time) the user actually clicks on the displayed advertisement. A natural algorithm for this problem, therefore, is to display the advertisement for which the $bid \times click-through-rate$ is maximized.

An additional feature of the Adwords problem is that each advertiser specifies a daily budget, which is the maximum amount the advertiser will spend per day. Therefore a problem instance comprises a bid value for some or all bidder-keyword combinations and a budget value for each advertiser. Note that restricting to unit bids and unit budgets yields the online matching problem.

In [23], the authors solve this problem by using the tradeoff function $\phi(x) = 1 - \exp(x - 1)$, which they derive using a tradeoff-revealing family of LPs. Their algorithm is simply: allocate the next query to the bidder i that maximizes the product of its bid and $\phi(B_i)$, where B_i is the fraction of i's budget already spent. The algorithm is $(1 - 1/e)$-competitive, on the assumption that budgets are much greater than bids.

The AdWords problem can be reinterpreted as a certain online version of the sensor-assignment problem:

- bidders $\longrightarrow$ missions
- adwords $\longrightarrow$ sensors
- each adword gets one winning bid $\longrightarrow$ each sensor is applied to (at most) one mission
- b_i's bid for w_i $\longrightarrow$ amount that s_i can contribute to t_i
- b_i's budget $\longrightarrow$ s_i's total possible contribution
- maximizing search engine profit $\longrightarrow$ maximizing total sensor utility

This online version of sensor assignment is a generalization of semi-matching. One assumption used in the analysis of the $(1-1/e)$-competitive AdWords algorithm is that daily budgets are substantially greater than individual bids. This assumption may carry over to our setting if daily budgets are construed as initial battery life or total lifetime contribution of a sensor.

D. Combining Semi-Matching with Max-Flow

Let us return to the flow-based problem we considered in section VII-A.1, where each mission has demand D_j, but with the addition of weighted edges and the semi-matching restriction from section VII-B: each sensor can serve just one mission at a time. Note that this problem generalizes both Semi-Matching and Max-Flow with Lower Bounds. Although problem formulations based on flow and semi-matching are both easy to solve by reduction to classical problems, this combined formulation appears to be both novel and hard. We will briefly discuss how SMD relates to some other important graph problems, one a special case and one a generalization.

Generalized Flow (a.k.a. Flows with Losses and Gains) [1] is a generalization of Max-Flow, in which each edge has both a capacity and a loss/gain factor (think of leaky pipes). By a simple reduction, we can translate SMD into a bipartite instance of Integer Generalized Flow. Unfortunately, Integer Generalized Flow is NP-Hard. It is interesting to note that

the authors of [29] use a Generalized Flow model to solve a related but distinct sensor selection problem. They partition the sensors into multiple feasible coverage sets to maximize network lifetime.

The *Bin Covering* problem [9], which is strongly NP-Hard, is sometimes described as a dual of the well known Bin Packing problem [13]. In Bin Covering, we are given a list of positive real numbers and a set of bins of unit size. A bin is considered *covered* if the numbers placed in it sum to at least 1, and the optimization problem is to cover as many bins as possible. It is easy to see that SMD is a generalization, since if impose on it the restriction that all mission demands equal 1 and that a sensor offers the same potential contribution to each mission then we obtain the Bin Covering problem.

We can prove either through a reduction from 3-Partition [13] or from the observation that Bin Covering is a special case that this problem is strongly NP-Hard. We can also show in the same way that the problem is APX-Hard [27] if we require missions to be satisfied exactly.

Given the intractability of SMD, there are two natural approaches to consider: optimization for easy cases, and approximation for hard cases. That is, if there are many available sensors, then look for a min-cost feasible assignment; if not, then approximate the goal of achieving all missions.

VIII. Open Research Problems

Although the problem of sensor selection has received a sizable attention lately, there are important open research problems that need to be addressed.

To the best of our knowledge, the issue of handling multiple missions that might have different priorities has not been discussed before. A mission can be divided into a set of small tasks. The tasks of multiple missions may not be disjoint, i.e. one or more tasks can be shared between missions and hence there is no need to have duplicate nodes doing the same task. Also, missions may change during the lifetime of the network but some tasks may not, e.g. monitoring an area of interest. Because each node may perform multiple tasks the issue of task-scheduling also arises. Moreover, missions may have different priorities, so handling the assignments of nodes to missions needs to take this into account.

Another issue here is node reassignment. A node that is assigned to one mission may later be reassigned to another mission because it is more useful there. That node then needs to find a replacement. Effects of such node reassignment on missions and its associated cost need to be studied. Figure 4 illustrates an example of handling two missions with different priorities and shows a case in which nodes are reassigned.

The high cost of sensor deployments make multi-modal sensors more interesting. These sensors are able to collect different types of data which means that their contribution to missions will depend on the activated modality. Reconfiguration of the sensors in such a situation must take into account both the task utility and the information gain, along with other factors to choose which modality should be activated and thus poses interesting challenges.

The issue of sensor utility which determines how useful a sensor is to a task is studied in different papers (see section V and VI). However, there are no realistic models for utility that take the quality of information into account. Currently, most schemes determine the utility of a sensor by its physical attributes such as location. We need to develop quality of information models that take not only physical attributes, but also some sort of semantics into account. For example, a video sensor that is the closest to an event might not be the best candidate for selection because its view of the event is obscured by smoke. Also, although it is embedded in some of the information-gain schemes, the issue of conditional utilities need to be studied in more detail. By conditional utilities we mean how would the selection of one sensor affect the utility of another sensor. For example, in Figure 5, video sensor A may have a high value if selected alone. However, if F is also selected then A's value may be lower since the two cameras have a high overlap in coverage (the area between the two dotted line represent a camera's coverage).

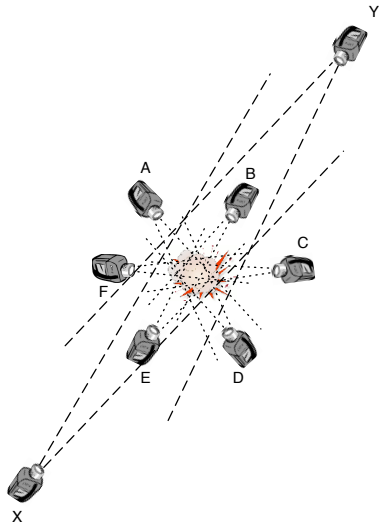

Fig. 5. Difference between optimal vs. actual sensor assignment. The optimal set of cameras that should cover the event are the six closest ones (A, B, ... F). However, due to bandwidth constraints, two further cameras (X, Y) are chosen to cover a larger area of the field.

Many of the papers that propose these sensor selection schemes look at the problem from a theoretical stand point. But another issue that needs to be studied is the dynamics of these different schemes and how would they perform in realistic settings in which messages can be dropped and nodes can fail. Also, the different performance aspects of these schemes such as convergence times, communication overhead and how they affect sensor network lifetime need to be studied.

In most schemes, the only cost that is considered when selecting sensors is energy. Although energy might be the most valuable resource in sensor nodes, other resource constraints should be considered. Bandwidth, which is also limited, can be a deciding factor when selecting sensors. For example, an optimal solution to a sensor selection problem might suggest the use of 6 sensors, however, due to bandwidth constraints only 2 can be activated. In this case, the 2 sensors that are selected might not even be from the optimal set of 6 sensors. This example is illustrated in Figure 5.

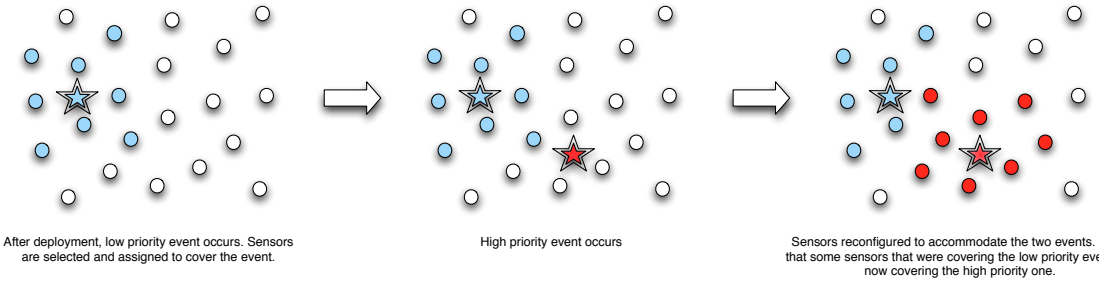

Fig. 4. Multi-mission support.

Another interesting topic is to study whether and how the purpose of selection (for monitoring, answering a query, disseminating data, fault tolerance, etc.) may affect the selection scheme. For instance, the selection schemes for choosing sensors for monitoring purposes may face different requirements and challenges when compared to choosing sensors for answering a query.

IX. Conclusion

In this paper, we examined the problem of sensor selection in wireless sensor networks. We discussed four different classes of schemes, namely (1) coverage schemes, (2) target tracking and localization schemes (3) single mission assignment schemes and (4) multiple mission assignment schemes. We also looked at solutions to similar selection and matching problems in other fields and discussed their applicability to sensor networks. We believe that there are some important open research problems in this area and we have discussed some of them here.

Acknowledgments

This research was sponsored by US Army Research laboratory and the UK Ministry of Defence and was accomplished under Agreement Number W911NF-06-3-0001. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the US Army Research Laboratory, the U.S. Government, the UK Ministry of Defence, or the UK Government. The US and UK Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

References

[1] R. Ahuja, T. Magnanti, and J. Orlin. *Network Flows*. Prentice Hall, 1993.

[2] J. Ai and A. Abouzeid. Coverage by directional sensors in randomly deployed wireless sensor networks. *Journal of Combinatorial Optimization*, Feb. 2006.

[3] F. Aurenhammer. Voronoi diagrams - a survey of a fundamental geometric data structure. *ACM Computing Surveys*, 1991.

[4] J. Bernardo and A. Smith. *Bayesian theory*. Wiley, New York, 1996.

[5] F. Bian, D. Kempe, and R. Govindan. Utility-based sensor selection. In *Proceedings of the IEEE Conference on Information Processing in Sensor Network (IPSN 2006)*, April 2006.

[6] J. Byers and G. Nasser. Utility-based decision-making in wireless sensor networks. In *Proceedings of the IEEE Workshop on Mobile and Ad Hoc Networking and Computing*, 2000.

[7] M. Cardei and D. Du. Improving wireless sensor network lifetime through power-aware organization. *ACM Wireless Networks*, May 2005.

[8] M. Cheng, L. Ruan, and W. Wu. Achieving minimum coverage breach under bandwidth constraints in wireless sensor networks. In *Proceedings of 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, 2005.

[9] J. Csirik, D. Johnson, and C. Kenyon. Better approximation algorithms for bin covering. In *Symposium on Discrete Algorithms*, pages 557–566, 2001.

[10] E. Ertin, J. Fisher, and L. Potte. Maximum mutual information principle for dynamic sensor query problems. In *Proccedings of the 2nd International Workshop on Information Processing in Sensor Networks (IPSN '03)*, 2003.

[11] S. Even. *Graph Algorithms*. Computer Science Press, 1979.

[12] L. Ford and D. Fulkerson. *Flows in networks*. Princeton University Press, 1962.

[13] M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Colmpleteness*. Freeman, 1979.

[14] V. Isler and R. Bajcsy. The sensor selection problem for bounded uncertainty sensing models. In *Proceedings of the 4th international symposium on Information processing in sensor networks*, 2005.

[15] B. Kalyanasundaram and K. Pruhs. Online weighted matching. *Journal of Algorithms*, 14:478–488, 1993.

[16] B. Kalyanasundaram and K. Pruhs. An optimal deterministic algorithm for online b-matching. *Theoretical Computer Science*, 233:319–325, 2000.

[17] L. Kaplan. Global node selection for localization in a distributed sensor network. *IEEE Transactions on Aerospace and Electronic Systems*, 42(1):113–135, January 2006.

[18] L. Kaplan. Local node selection for localization in a distributed sensor network. *IEEE Transactions on Aerospace and Electronic Systems*, 42(1):136–146, January 2006.

[19] R. Karp, U. Vazirani, and V. Vazirani. An optimal algorithm for online bipartite matching. In *Proceedings of STOC*, 1990.

[20] K. Kwok, B. Driessen, C. Phillips, and C. Tovey. Analyzing the multiple-target-multiple-agent scenario using optimal assignment algorithms. *Journal of Intelligent and Robotic Systems,*, Sept. 2002.

[21] J. Liu, J. Reich, and F. Zhao. Collaborative in-network processing for target tracking. *Journal on Applied Signal Processing*, 2002.

[22] J. Lu, L. Bao, and T. Suda. Coverage-aware sensor engagement in dense sensor networks. In *Proceedings of the International Confernece on Embedded and Ubiquitous Computing - EUC 2005*, Dec. 2005.

[23] A. Mehta, A. Saberi, U. Vazirani, and V. Vazirani. Adwords and generalized on-line matching. In *Proceedings of FOCS*, 2005.

[24] T. Mullen, V. Avasarala, and D. L. Hall. Customer-driven sensor management. *IEEE Intelligent Systems*, 21(2):41–49, Mar/April 2006.

[25] J. Ostwald, V. Lesser, and S. Abdallah. Combinatorial auctions for resource allocation in a distributed sensor network. In *Proceedings of the IEEE Real-Time Systems Symposium*, December 2005.

[26] P. Pahalawatta, T. Pappas, and A. Katsaggelos. Optimal sensor selection for video-based target tracking in a wireless sensor network. In *Proceedings of the International Conference on Image Processing (ICIP '04)*, 2004.

[27] C. Papadimitriou. *Computational Complexity*. Addison Wesley, 1993.

[28] C. Papadimitriou and K. Steiglitz. *Combinatorial Optimization: Algorithms and Complexity*. Dover, 1998.

[29] M. Perillo and W. Heinzelman. Optimal sensor management under energy and reliability constraints. In *Proceedings of the IEEE Conference on Wireless Communications and Networking*, March 2003.

[30] A. Sekhar, B. Manoj, C. Siva, and R. Murthy. Dynamic coverage maintenance algorithms for sensor networks with limited mobility. In *Proceedings of the IEEE Conference on Pervasive Computing and Communications*, March 2005.

[31] K. Shih, Y. Chen, C. Chiang, and B. Liu. A distributed active sensor selection scheme for wireless sensor networks. In *Proceedings of the IEEE Symposium on Computers and Communications*, June 2006.

[32] G. Wang, G. Cao, and T. L. Porta. Movement-assisted sensor deployment. *IEEE Transactions on Mobile Computing*, June 2006.

[33] G. Wang, G. Cao, and T. L. Porta. A bidding protocol for deploying mobile sensors. In *Proceedings of the IEEE Conference on Network Protocols (ICNP '03)*, November 2003.

[34] H. Wang, K. Yao, G. Pottie, and D. Estrin. Entropy-based sensor selection heuristic for target localization. In *Proceedings of the third international symposium on Information processing in sensor networks*, 2004.

[35] T. Yan, T. He, and J. Stankovic. Differentiated surveillance for sensor networks. In *Proceedings of 1st International Conference on Embedded networked sensor systems*, 2003.

[36] F. Zhao, J. Shin, and J. Reich. Information-driven dynamic sensor collaboration. *IEEE Signal Processing Magazine*, 19(2):61–72, March 2002.