Agile Sensor Networks
Adaptive Coverage via Mobile Nodes

Swapna Ghanekar, Fatma Mili, Imad Elhajj

Oakland University
Computer Science and Engineering Department
Rochester, MI 48309
sdghanek@oakland.edu, mili@oakland.edu, elhajj@oakland.edu

Abstract

Advances in sensing and communication technology make sensor networks a convenient and cost effective tool for collecting data in hard to reach and hazardous areas Increasingly, sensor networks are used to monitor the environment and enable swift and accurate intervention. Environmental monitoring is characterized by the facts that the area under surveillance tends to be large whereas incidents tend to be both sparse in time and localized. In this research, we investigate means by which we get good coverage, so that we do not miss events of interest, and we reduce cost, so that we do not deploy too many sensors in areas where nothing is happening.

We propose to use a combination of static and mobile sensors. Initially, the nodes are randomly deployed. While the static sensors remain in place until they die out, mobile sensor nodes are constantly evaluating their position, scouting for "interesting" events. They move to areas where they can contribute useful and relevant information. As the dynamics of the events move, so do the mobile nodes. In this paper, we present the decision making process mobile nodes go through in order to adaptively adjust coverage. This process is simulated and the results presented.

1. Introduction

Sensor networks have been applied to a variety of monitoring applications [1]. Some of these applications are short term interventions surrounding a one-time event such as military interventions [2], natural disasters [3], and other accidents [4]. In these cases, *once* an event has already been detected, the objective is to get the most accurate and timely data about it using all the resources available. By contrast, other applications aim at *detecting* these events *before* or *as soon as* they happen [1]. Most environmental monitoring applications fall in this category. In these cases, it is not known before hand when or where the event will happen, thus a *relatively large area* needs to be monitored for a *relatively long time period*.

A key issue related to the latter category is the accurate detection of the phenomena of interest in a cost-effective manner. Because the area in which the event may occur is extended; For example, sensor nodes can be deployed in a forest to detect fires as they originate [1]. In this case, the area considered can cover hundreds of square miles. It would be costly to flood the area completely with sensors knowing that only a small percentage will ever detect anything. On the other hand, putting too few sensors runs the risk of completely missing phenomena when they do happen, or detecting them inaccurately.

In this paper, we investigate an approach in which we combine two types of sensor nodes. A set of static wireless sensor nodes are used to provide a minimal coverage of the complete area of interest. These static nodes are supplemented with a set of mobile nodes that have sufficient power and possibly some recharging capability to allow them to move from regions where currently nothing is happening towards regions where they can be of some value in collecting relevant and useful information.

In other words, mobile sensor nodes are constantly evaluating their position and scouting for "interesting" events. Their movements reflect the dynamics of the events on the ground. This approach raises a number of new issues not addressed by the traditional approach to sensor networks management. In particular, we are interested in two decisions that mobile nodes need to make at any point in time, namely: to move or not to move, and if they decide to move, what direction.

The literature contains significant work related to coverage especially in robotic applications. However, these approaches attempt to decide on coverage without accounting for the phenomena being sensed. That is they do not have the capability of concentrating on areas of interest instead they simply try to spread out [5]. Another limitation that some of these approaches have is the requirement that nodes be localized and aware of the location of their neighbors [6].

In this paper we introduce an adaptive approach for mobile sensor nodes to cover an area of interest while accounting for the phenomena being monitored. This would result in dense coverage is areas of interest and sparse coverage in areas of limited immediate interest. This approach would result in significantly higher accuracy for queries being sent to the network.

This paper is organized as follows: In section 2, we introduce the model of the sensor network. In this section we also discuss the dynamics, which define the motion of the nodes. In section 3, we describe the network agility. Section 4 gives the simulation results and section 5 concludes the paper.

2. Model: Node View

Given an area of interest, which we represent by a two-dimensional grid, we model the sensors as nodes on the grid. We assume the existence of a sink node submitting queries to the network, and synthesizing the answers received. The sink node decides at any point in time, which nodes to query. Our focus in this paper is on the movement of the mobile nodes. We use the paradigm of classical mechanics, namely Newton's laws of motion, to model the motion of the mobile nodes. Mobile nodes are thought of as objects that are subjected to two external forces, a frictional force and a repulsive force. We define below the nature of these forces and how we use them to make the nodes decide when to move, how fast to move, and, to some extent, where to move.

2.1 Usefulness: An Information Value Based Concept

The effect of frictional forces in mechanic is to resist movement. We model the frictional force that a node is subjected to by the *usefulness* of that node at its current position.

The motivation behind modeling the frictional force as usefulness is the fact that *useful* nodes should resist motion more than *un-useful* nodes. The concept of usefulness of a sensor node was introduced in [7]. In that paper we advocated the use of the inherent redundancy in sensor networks in order to use them efficiently, increase the accuracy of the queries submitted to them, and lengthen their life. In particular, we use the fact that most queries submitted to long term monitoring sensor networks are aggregate queries (such as max, sum, and average). By nature, these aggregate functions are tolerant of incomplete data sets.

Of particular interest is the fact that not all measurements play an equally important role in computing these functions. At one extreme, in computing the function max, we can omit all measurements but the max (assuming we know which one it is before hand) and still obtain the correct value. In this case, the higher the value sensed by a node, the higher is its information value. In general we define the *usefulness* of a node i for a query q as a metric taking values between 0 and 1 that captures the information value of the data sensed by that node. The formula of usefulness varies with the query being computed. We show the formula for the query max:

$$F_{frictional}(i, Max) = usefulness\ (i, Max) = P(X<x_i)$$

where x_i is the value sensed at node i and X ranges over the set of all the sensed values in the network (or cluster) of interest.

2.2 Redundancy: A Measure of the Node's Commonality

The effect of repulsive forces in classic mechanics is to propel an object away from its current position. We model the repulsive force that a node is subjected to by the *redundancy* of that node at its current position; that is, the commonness of the nodes measured value.

The motivation behind modeling the repulsive force as redundancy is the fact that a node returning information, useful or not, should be encouraged to move away if there is a significant number of other nodes around it sensing a similar value. The usefulness of a node is an absolute measure of its potential contribution to a query and does not capture well whether other nodes carry the same information or not. For example, the usefulness of a node measuring 10 is the same whether X={1, 1, 10} or X={9, 9, 10}. In both cases, the 10 will have the same usefulness, although intuitively, it is much "more redundant" in the second set. In other words, a node measuring 10 in the second set can be skipped without much loss of information, whereas the measurement of 10 in the first set is critical. Its absence dramatically impacts the final result. The concept of redundancy of a node i relative to a query Max is captured by the following expression:

$$F_{repulsive}(i, Max) = Redundancy\ (i, Max) = 1\text{-}1/N$$

where N is the number of nodes in the neighborhood of node i sensing values, V_n, such that,

$$Vi - \varepsilon \le V_n \le Vi + \varepsilon$$

where ε is a predefined constant, called coefficient of redundancy. Because in this case we are interested in redundancy as a reflection of local spatial redundancy, it is measured within a neighborhood of diameter d, which translates to nodes within a transmission range d. Clearly, redundancy is zero when a node is unique and it tends to 1 as the redundancy increases.

2.3 Dynamics

The motivation behind node motion is that it can be of more value elsewhere and it can dynamically adjust to changes in the phenomena being monitored. The decision to move is taken by each node individually based on the frictional force (usefulness) and the repulsive force (redundancy). The usefulness of a node is computed by the sink in light of information received by all nodes queried. As a service to the nodes, whenever the sink submits a query to a node, it sends the node's current usefulness value. The redundancy of a node is calculated from information it learns about its neighborhood. Every node sets a listening range *d* (the radius of its neighborhood) and hears values sent by nodes within this listening range to the sink. The more nodes in the listening range return similar values to its own the higher is the node's redundancy value.

According to Newton's Second Law of Motion, an object's momentum is proportional to the sum of the forces, *F*, it is subjected to. In fact, $F = m.a$ where *m* is the mass and *a* is the acceleration vector, and *F* is the vector sum of forces. We take a somewhat simplified interpretation of this law whereby, the force $F= F_{repulsive} - F_{frictional}$. A positive value of *F* sets an object in motion. That is in order to cause an object to move the following condition has to apply,

$$F_{frictional} < F_{repulsive}$$

In other words, when a mobile node satisfies the condition,

$$P(X<x_i) < 1- 1/N$$

where all the symbols are as defined before, the node is propelled to move. However, if all the nodes with this condition satisfied move then too much mobility occurs and potentially most of the nodes would leave that area. For example, if two mobile nodes *i* and *j* are redundant when compared to each other, ideally, only one of them should move. Therefore, a node eligible to move, that is satisfying the above condition, should move with a certain probability. In addition, having mobile nodes take turns moving and stopping should be avoided. Ideally only a few nodes decide to move, and once they do, they keep moving with a high probability as long as Newton's Second Law of Motion allows them to. This behavior is captured by Newton's First Law of Motion that makes objects resist change. Non moving nodes "prefer" not to move and moving nodes "prefer" to keep moving.

To capture the probabilistic behavior and Newton's First Law of Motion, a node's probability of motion is defined as,

$$P(Motion)=(F_{repulsive} - F_{frictional})*K=(1- 1/N - P(X<x_i))*K$$

where

$$K = \begin{cases} C \ if \ node \ was \ in \ motion \\ \frac{1}{C} if \ node \ was \ not \ in \ motion \end{cases}$$

where, *C* is a contact greater than 1, called the coefficient of first law of motion.

The motivation for this definition is that based on the second law of motion the term $F=F_{repulsive} - F_{frictional}$ can be perceived as the acceleration of the node and multiplying by *K* captures the first law of motion behavior. So if the node was moving then *P(Motion)* increases and if it was stationary then *P(Motion)* decreases.

Also the term $F=F_{repulsive} - F_{frictional}$, which is the acceleration, is used by the node to deduce the speed of motion. The node keeps track of its current speed and based on the acceleration obtained it calculates the new speed of motion.

We use the Laws of motion to control the decision to move or not and with what speed, but we do not use them directly to control the direction of movement. This reflects the fact that typical sensors do not have an accurate and reliable mechanism to perceive distances and direction. For example, when a node listens to transmission activity in its neighborhood, it hears activity within the predefined listening range but is not able to efficiently nor accurately determine the two-dimensional topology of the nodes it hears. Therefore, in computing the repulsive force, for instance, we account for the intensity of the force – captured by the number of nodes— but not its direction. We decouple the decision to move (intensity) from that of where to move (direction).

The requirements on criteria used to select the direction of movement are similar to those used to decide to move, namely: the decision must be made based on information available or easily obtainable by the node in question, and the decision must not require extensive storage or computation. We consider two different approaches:

Random direction: At every step, a direction is chosen at random and used. This approach presents the advantage of not requiring any information or any memory. It counts on the fact that as long as the node has not reached an interesting area, it will keep moving. As shown in the simulation section, given enough time, this approach does converge and leads most mobile nodes to move towards interesting regions. The key disadvantage of this approach is that it converges too slowly and uses up too much energy moving in the wrong direction. Another shortcoming of this approach is that it also conflicts with Newton's first law of motion which dictates that an object should resist change in motion, i.e. it should maintain the same direction of movement—among other things—unless compelled otherwise by the forces it is subjected to.

Gradient-based direction: Using locally available information, sensor nodes can determine the intensity of the force or in other words the acceleration, $F= F_{repulsive} -$

$F_{frictional}$, they are subjected to, but not its direction. But, using some limited memory, they can compute an approximation of the direction and use it to move. The first time a node is propelled to move; it will maintain that same direction for a predetermine number of steps, *M*, and remember the acceleration at the beginning of the journey. After the *M* steps, the node examines the difference between the current acceleration and the acceleration at the beginning of the itinerary and adjusts its direction accordingly. If we limit ourselves to 4 different directions, (N, S, E, W), the node maintains the same direction if the acceleration decreased; reverses direction if it increased, and picks one of the orthogonal direction if the acceleration changed very little. This approach allows the nodes to minimize random wandering and converging towards better regions in less number of iterations.

Since localization typically requires a significant amount of power, in the above discussion we have used the assumption that sensors do not have a highly accurate mechanism of sensing exact location (their own as well as that of nodes around them) [8]. In case some of the nodes are location aware, the mobile nodes no longer need to move "blind-folded". They would have full information about both intensity and direction of the forces they are subjected to and move accordingly.

3. Model: Network View

We consider now the network as a whole. For the sake of abstraction and readability, we assume that the whole network forms a single cluster, i.e. the network consists of a sink node and a set of static and mobile nodes. The sink node receives or formulates a query; it is responsible for querying the nodes of the network, receiving their answers, and synthesizing them into an answer to the query. In the context of the applications of interest, the queries are typically long-running aggregate queries. They are long running in the sense that the same query is run repeatedly with a predefined frequency (i.e. every second, every hour, etc.). The queries typically compute aggregate functions such as minimum, maximum, sum, and average.

One of the key issues related to such networks are framed in terms of query optimization [9] taking into account the redundancy inherent in these networks and their associated queries as well as the cost functions. In [7], we formulate a number of metrics that are used by the sink nodes to assess the utility of individual nodes (combination of usefulness, cost, and power available) and decide accordingly whether to query them or not. These metrics present the advantage of being easy to compute, based on information already collected for other purposes, and of reflecting the dynamics on the ground as they evolve.

All the features presented in [7] still apply here. The sink keeps a record of the usefulness of the nodes (among other things). These usefulness values are used to decide whether to query a node or not, and are updated whenever the node is queried. As a result, a node that is not queried frequently has a utility value (in the sink) that is relatively outdated. When the nodes do not move, the speed of change of their utility is relatively uniform and reflects the speed of change of the phenomenon being monitored. On the other hand, when some nodes are mobile, they *force* a speed of change to their utility that is higher than the speed of change to other nodes. If they start with a low usefulness the likelihood of their being queried is low, and so is the frequency with which their utility will be updated. This may mislead them into believing that they are not making any progress when they actually may be. In contrast with the rest of the nodes, mobile nodes need accurate and timely information about their usefulness. We will assume here that they are updated at every iteration. Alternative approaches involving distributed calculation of an approximate utility are under experimentation.

In summary, the mobile nodes need the following information and data structures to make their decisions:

- Whether the node is globally useful or not. The usefulness value is computed by the sink (in light of the values of all nodes in the cluster) and transmitted to the node.
- Whether the node is redundant or not. The redundancy is a reflection of whether it is currently providing a unique –or rare— service or not and is computed by listening to the values returned within listening range *d*. The node needs a storage array in which it keeps the values heard at the last iteration.
- Inertia state of the node. This consists of the node's speed (0 if it is not moving) and direction and the last value of *F* (to see if it is increasing or decreasing).

The storage requirements overhead that this imposes on mobile nodes are therefore limited.

4. Simulation Results

We have set up Matlab simulations to test the feasibility of the approach proposed. In particular, we wanted to seek answers to the following questions:

1. Can the "blind-folded" mobile nodes eventually make their way towards interesting regions, i.e. regions with values of high utility and low density?
2. What is the difference between gradient-based and random direction choices?
3. What is the impact of the moving nodes on the accuracy of the queries?
4. How well does the network adapt with a dynamic phenomenon?

We simulated a setup in which we created a region of *20x20* as shown in Figure 1. The data values in the region vary linearly from *5* to *100* from the "bottom" to the "top".

Given that query Max was simulated, the upper regions have high values for the phenomenon being monitored and the lower regions have low values. Conceptually, the area has four regions going from the upper left corner in a clockwise movement the regions are (high usefulness and high density, high usefulness and low density, low usefulness and low density, and low usefulness and high density) as shown in Figure 1. This "simplistic" partitioning was selected to visually test the questions. If the approach is valid, mostly nodes from the lower left corner, where there is low friction and high redundancy, would move towards the upper right corner, where there is high friction and low redundancy.

The sensor network is initialized with 20 mobile and 20 stationary nodes randomly placed in the "left" two regions and 10 mobile and 10 stationary nodes randomly placed in the "right" two regions. So 60 sensors are used in total compared to the 400 locations possible resulting in 15% coverage. The simulation uses a coefficient of first law of motion C=1.1 and a coefficient of redundancy ε=15. The initial configuration of the nodes is shown in Figure 2. Circles represent stationary nodes and squares represent mobile nodes.

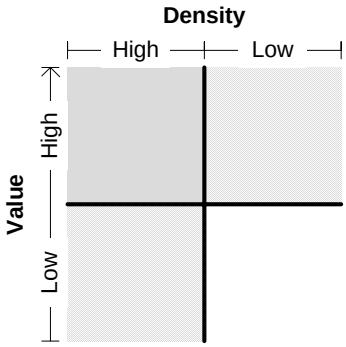

Figure 1: Sensor network area being simulated.

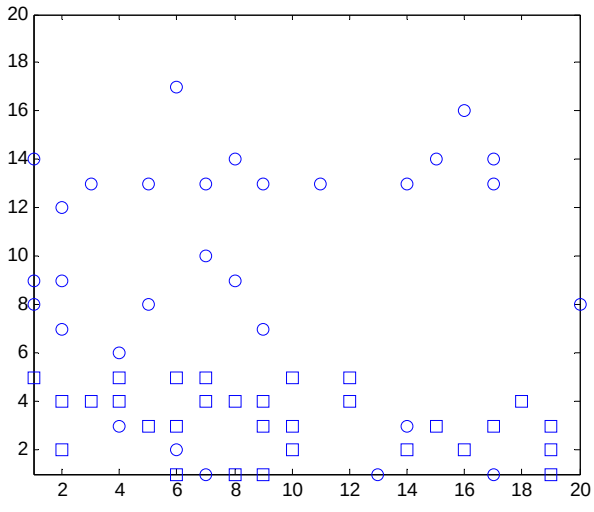

Figure 2: The initial location of the nodes.

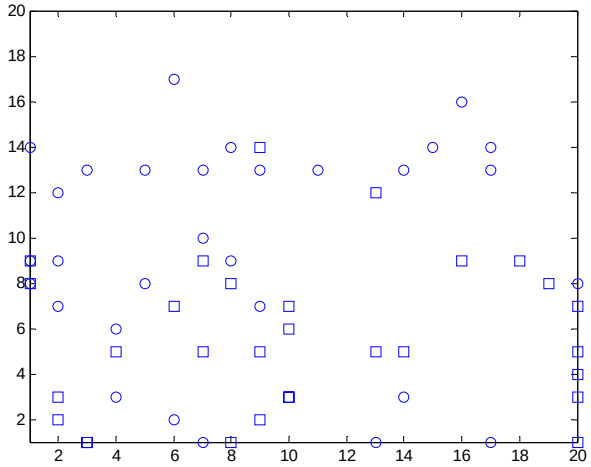

Figure 3: Location of the nodes after 50 iterations using random direction.

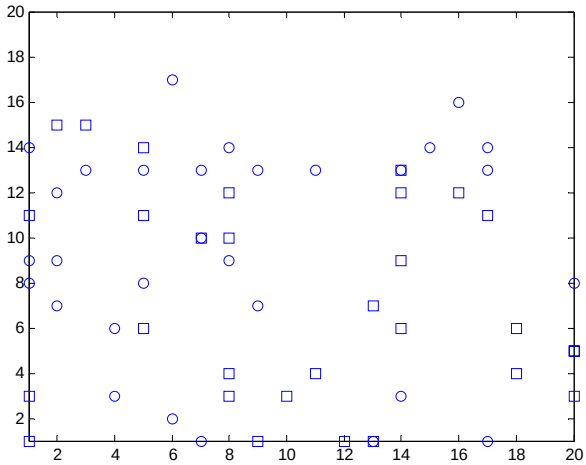

Figure 4: Location of the nodes after 50 iterations using gradient-based direction.

We have simulated the behavior of the network using random-based movement and gradient-based movement. The distribution of the nodes after 50 iterations is shown in Figure 3 and Figure 4 for random and gradient-based respectively. It is clear that mobile nodes moved and scattered in areas of high values. As shown, the network exhibits the behavior expected. The mobile nodes scatter around to create a coverage which is sparse in areas of minimal interest and dense in areas of more interest. This clearly answers the first question. Regarding the second question, comparing Figure 3 and Figure 4 it is clear that gradient-based out performs random-based since more mobile nodes have moved to the areas of interest during the same amount of iterations.

To understand the impact of the moving nodes on the accuracy of the queries, consider Figure 5, which shows the Max query percentage error as time goes on. The figure shows the error relatively quickly decreasing to zero and

maintains low values. As for the performance with dynamic phenomenon, the simulation was conducted with data values dynamically changing. The values distributed followed a time varying sine wave which moved the maximum from the top to the bottom of the area and then back to the top. Figure 6 shows the Max query percentage error as time goes on. It is clear that as the values change the error increases till the network adapts at which point the error decreases again. This illustrates that the network is capable of adapting coverage according to the dynamics of the phenomenon being measured.

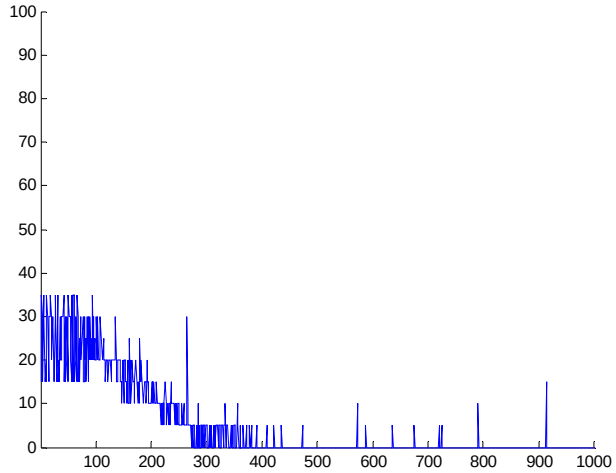

Figure 5: Query percentage error versus iteration with static phenomenon.

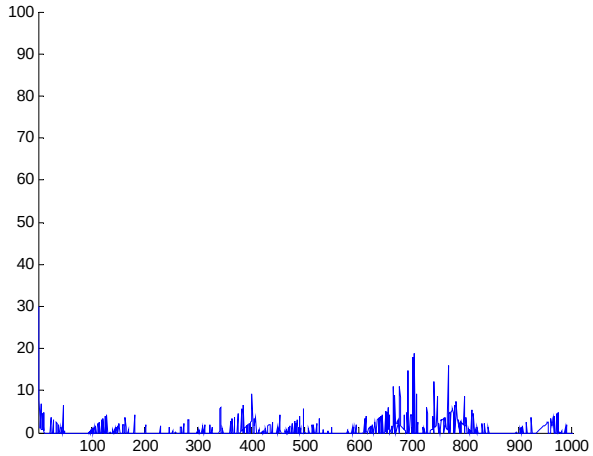

Figure 6: Query percentage error versus iteration with dynamic phenomenon.

5. Conclusions

This paper developed a scheme that allows adequate coverage of large areas using a relatively small number of sensor nodes. The key feature of the approach is a mix of static and mobile nodes. The static nodes ensure continuous connectedness of the network and the mobile nodes ensure accurate detection of dynamic phenomena. The approach is distributed and based on local information only which makes it scalable. Also the approach does not require position information thus there is no need for localization. The simulation presented confirmed the theory developed and its feasibility. Several issues remain to be investigated concerning the ideal parameters to use in the algorithm and the performance of the algorithm with highly dynamic phenomena. In particular, this approach will be tested using data from actual environmental phenomenon.

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci, “Wireless sensor networks: a survey,” Computer Networks, 38(4), p. 393-422, 2002.

[2] Chee-Yee Chong, Srikanta Kumar, “Sensor Networks: Evolution, Opportunities, and Challenges,” Proceedings of the IEEE, Vol. 91, No. 8, p. 1247-1256, August 2003.

[3] P. Bonnet, J. Gehrke, P. Seshadri, “Querying the physical world,” IEEE Personal Communications p. 10–15, October 2000.

[4] Andreas Savvides, Jia Fang, Dimitrios Lymberopoulos, “Using Mobile Sensing Nodes for Dynamic Boundary Estimation,” WAMES workshop in conjunction with MobiSys, Boston, 2004.

[5] Maxim A. Batalin and Gaurav S. Sukhatme, “Sensor Coverage using Mobile Robots and Stationary Nodes,” In Procedings of the SPIE, Vol. 4868, p. 269-276, Boston, MA, August 2002.

[6] Andrew Howard, Maja J Matari´c, and Gaurav S Sukhatme, “Mobile Sensor Network Deployment using Potential Fields: A Distributed, Scalable Solution to the Area Coverage Problem,” In Proceedings of the 6th International Symposium on Distributed Autonomous Robotics Systems (DARS02) Fukuoka, Japan, June , 2002.

[7] Mark Rossman, Iris Bass, Fatma Mili, Imad Elhajj, “Query Optimization in Wireless Sensor Networks,” IEEE International Conference on Robotics and Automation, Florida, 2006. (Submitted)

[8] Lingxuan Hu and David Evans, “Localization for Mobile Sensor Networks,” Annual International Conf. on Mob ile Computing and Networking, 2004.

[9] S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, “TinyDB: An Acquisitional Query Processing System for Sensor Networks,” ACM TODS 2005.