Coordinated Locomotion of Mobile Sensor Networks

Seokhoon Yoon, Onur Soysal, Murat Demirbas, and Chunming Qiao

Computer Science and Engineering, State University of New York at Buffalo

Email: {syoon4,osoysal,demirbas,qiao}@cse.buffalo.edu

Abstract

Stationary wireless sensor networks (WSNs) fail to scale when the area to be monitored is open (i.e borderless) and the physical phenomena to be monitored may migrate through a large region. Deploying mobile sensor networks (MSNs) alleviates this problem, as the self-configuring MSN can relocate to follow the phenomena of interest. However, a major challenge here is to maximize the sensing coverage in an unknown, noisy, and dynamic sensing environment while minimizing energy consumption. Another major challenge is to maintain network connectivity for each MSN node during relocations. To address these challenges, we propose a new distributed algorithm, Causataxis[1], that enables the MSN to relocate toward the interesting regions and adjust its shape and position as the sensing environment changes. Causataxis achieves scalable control of the MSN via a backbone-tree infrastructure maintained over clusterhead nodes, and achieves agility via localized cluster formation and dissolution. Unlike conventional cluster-based systems with backbone networks, a unique feature of our proposed approach is its bio-system inspired growing and rotting behaviors with coordinated locomotion.

We compare Causataxis with a custom tuned swarm algorithm, which uses the concept of virtual spring forces to relocate mobile nodes based on local neighborhood information. Our simulation results show that Causataxis can outperform the swarm based algorithm in terms of the sensing coverage, the energy consumption, and the noise tolerance with a slightly high communication overhead.

I. INTRODUCTION

A core problem in geophysical sciences, ecology/habitat monitoring, and military surveillance/reconnaissance fields is that of gathering data with appropriate coverage and sufficient fidelity from a region of interest. Wireless

---

[1]In Latin *causa* means motive/interest. A taxis (plural taxes) is an innate behavioural response by an organism to a directional stimulus. We use *Causataxis* to refer to an *interest driven relocation behavior*.

sensor networks (WSNs) technology has emerged to respond to this need, and received a lot of attention both from the academia and the industry due to its potential for revolutionizing the data gathering process in several of these fields [1], [2]. Due to low cost of WSN nodes, it is possible to deploy WSNs densely for finer-grain, high-fidelity, and real-time monitoring. Unfortunately, when the region to be surveyed is large and/or the phenomena of interest to be monitored may migrate over time in a large area, deploying WSNs becomes infeasible. One such scenario is monitoring of toxin levels in the sea as the toxic material spread and move.

Mobile sensor networks (MSNs) have been advocated [3]–[7] to alleviate the shortcomings of stationary WSN in terms of coverage area. MSN nodes have the capability to relocate to follow the phenomena or position themselves to survey the most interesting regions. Recently, several experimental platforms for MSNs have emerged from the military, industrial, and academic research [8]–[10].

However, monitoring a large area with MSNs brings new challenges, including,

- Self-configuration: once deployed in an unknown area, mobile sensor nodes should configure to cover the area.

- Agility: As the phenomena of interest expand, shrink, or migrate to other places, MSN should adjust to the change of the dynamic sensing environment to maximize the sensing coverage.

- Network connectivity: mobile sensor nodes should have access to a basestation to report the current sensing readings. If only a subset of nodes have direct connectivity to the basestation (e.g. via more powerful transmitters), the rest of nodes should have multihop paths to those that have that capability.

- Energy-efficiency: energy efficiency is critical to lengthen the network lifetime. Therefore, the traveling distance of mobile nodes and the communication overhead should be minimized.

- Noise tolerance: the sensing environment is subject to a high level of spatial and temporal noise as well as the sensor reading error. Regardless, the MSN should be able to find the optimal location of deployment.

A naive approach to MSN locomotion problem is to keep the topology of the MSN fixed by requiring nodes to preserve connectivity to their neighbors. However, such overconstraint restricts the flexibility of the MSN and prevents it from reconfiguring to cover the changing phenomena accurately. Another extreme for locomotion is to give nodes unconstrained flexibility. In this approach, potential field based swarming methods are often used for relocating nodes [4], [11]–[13], and gradient based methods are used to make the density of nodes in more interesting

areas higher. However, these methods all assume a bounded area of deployment (such that the network connectivity is ensured by virtue of the number of nodes in the MSN), and may lead to partitioning for large/unbounded regions. Thus, there is a need for a coordinated locomotion protocol that keeps the network from partitioning while providing enough *flexibility* for reconfiguring topology of the MSN to cover the changing area of interest accurately.

To address the challenges for coordinated locomotion of MSNs, we present a partitioning-free, flexible, and lightweight protocol, Causataxis. Compared to the previous approaches of maintaining a fixed topology MSN and providing unconstrained flexibility to MSN nodes, Causataxis combines the best of these two approaches. Causataxis provides a flexible, reconfigurable topology solution for MSNs, yet also maintains a global infrastructure to constrain the movements of the nodes and to orchestrate an efficient coordinated relocation of the MSN. The global infrastructure facilitates consensus among nodes, ensures connectivity of the MSN, and enables the MSN to move toward the area of most interest.

We compare and contrast the performance of Causataxis with that of a custom-tuned swarm algorithm [2]. Our simulation results show that Causataxis shows much better sensing coverage in noisy environments. The results also show that Causataxis is more energy efficient since the MSN nodes travel less under Causataxis than those under swarm algorithm. Compared to swarm approaches, Causataxis' ability to mitigate noisy environments, satisfy globally enforced constraint (such as network connectivity), and achieve energy-efficient MSN relocation make Causataxis a good candidate for many practical applications.

The rest of the paper is organized as follows. We discuss the system model and the goals in Section II. In Section III, we elaborate on the Causataxis algorithm and in particular, the *growingpoint* selection, *grow* and *rot* algorithms, and the coordinated locomotion algorithm. Our custom-tuned swarm algorithm and related work on swarm are discussed in Section IV. Simulation results and performance analysis follows in Section V. Finally, Section VI concludes the paper.

II. System Model

In our problem, a set of mobile sensor nodes form a mobile sensor network (MSN) and monitor phenomena of interest over a long period of time in an unknown and noisy environment.

---

[2]A Java-applet demonstration for Causataxis and swarm is available at http://ubicomp.cse.buffalo.edu/Causataxis

The MSN is required to monitor the area in which higher interest values will be observed (or sensor readings e.g. the density of chemical, the temperature, the light intensity, the frequency of the human motion activities, or the frequency of the eruption of volcano/intermittent spring). The interest value, $Z(x, y)$, is a non-isotropic function of $x - y$ coordinates. MSN nodes are equipped with sensors that can perceive the embedded interest value in the environment. Figure 1 and Figure 2 show examples of embedded interest values, $Z(x, y)$, in the two areas which are also used for simulation in section V.

We assume that MSN nodes are initially deployed at the same position which is randomly chosen in a large and unbounded area. We also assume that every MSN node knows its coordinates (nodes may be equipped with GPS), and nodes can control and steer their motion.

Our primary objective is to enable the MSN to locate toward and to deploy at the area of most interest while maximizing the sensing coverage. Our objectives also include minimizing energy consumption while preventing the MSN from being partitioned to ensure the network connectivity.
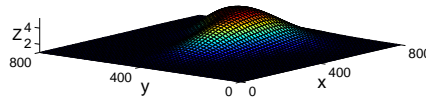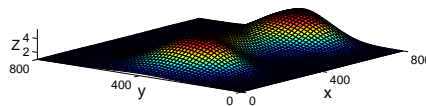

Fig. 1. Area with one peak of interest value

Fig. 2. Area with two peaks of interest value

III. CAUSATAXIS ALGORITHM FOR COORDINATED LOCOMOTION OF MOBILE SENSOR NETWORKS

Overview. In our Causataxis algorithm, the MSN is treated as an organism, which can grow and rot, and as a result, move. The MSN *grows* toward the area where higher interest values are expected, while the part of the MSN that has the low interest values *rots*. The grow and rot processes execute until the MSN can no longer obtain higher interest values. These processes resume when the MSN detects a change in the environment, such as migration of the phenomena of interest.

4

Causataxis achieves scalable control over the MSN by exerting a cluster-based two-tiered hierarchy consisting of clusterheads and cluster member nodes. Some mobile nodes become stationary (ossify) and take on the duty of serving as a clusterhead for their vicinity. These clusterheads form a backbone network over the MSN for coordinating the mobile nodes. This backbone network enables efficient information sharing among clusterheads, using which the MSN determines the most desirable direction. A spanning tree is maintained on this backbone network to facilitate aggregation of sensing data, and reporting of the data to the basestation. The backbone tree also assures network connectivity: since only clusters on the leaf of the backbone tree can rot and dissolve, the MSN is safe from the risk of partitioning.

In order to expand toward the area of more interest, one clusterhead is selected as the *growingpoint* or (*GP*). The *GP* creates new clusters by enabling its member nodes to become new clusterheads. The *GP* and clusterheads that have high interest value take cluster-member nodes from the clusterheads that have low interest values. If a cluster with a low interest value can not keep a certain number of member node, the cluster rots and its member nodes are absorbed into neighboring clusters.

The description of the cluster formation/network backbone establishment, and coordinated locomotion of the MSN are presented next. In particular, the selection of the GP, *grow*/*rot* algorithms, and the rate control of cluster data exchange for coordinated locomotion are discussed in detail.
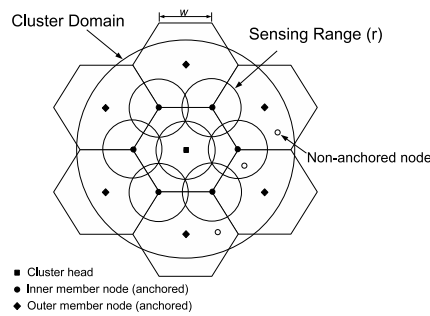

Fig. 3.   Node distribution in a cluster domain

A. *Cluster and backbone tree formation*

Each mobile node has to belong to at least one cluster, which is governed by a clusterhead. Initially, there is only one clusterhead (which is also *GP*), and the rest of the nodes are member nodes of the clusterhead. (Since

the nodes are deployed as singlehop initially, the initial cluster can be established trivially by a using timer based approach [14].)

As the network *grows* from the *GP*, new clusterheads establish direct one-hop wireless connections with neighboring clusterheads. The connections among the clusterheads result in the wireless mesh backbone network.

Each clusterhead instructs twelve of its member nodes to move to the desirable positions in its cluster domain. These member nodes are called 'anchor node', while others are 'non-anchor' nodes. In Causataxis, the member nodes try to form a hexagonal pattern in order to maximize the sensing coverage while avoiding any coverage hole.

Figure 3 shows a possible distribution of the anchor nodes in a cluster domain. A cluster can have up to 12 member nodes (six inner member nodes and six outer member nodes). The radius of the cluster domain is less than or equal to the radius of the reliable transmission range, R, so that a clusterhead can directly communicate with its member nodes. The system parameter w represents the distance between two neighboring nodes. To avoid any coverage hole, w should be less than or equal to $\sqrt{3}r$, where r denotes the sensing radius. Note that in practice, the sensing and communication range of different sensors vary and their movement locations can not be precisely obtained. However, our algorithms will still work in the presence of theses variations. To facilitate the following presentation, we will ignore these variations for the time being and model these variations later in our performance study.

1) Fault Tolerance to MSN Node Failure: Each node maintains *Heartbeat Timer* and to periodically broadcast *Heart beat* message. Upon receiving *Heart beat* messages, clusterheads update the membership of its cluster.

The cluster member node failure is recovered by the normal behavior of clusterheads. Specifically, when a clusterhead misses three consecutive *Heart beat* messages from a member node, the clusterhead regards the member node as unavailable and changes the corresponding member position status to *Vacant*, which will trigger a normal member position filling process.

A clusterhead failure can be recovered similarly to the way a cluster member node failure is recovered. Note that a clusterhead is also an outer member node of other clusterheads. If a $clusterhead\ A$ detects a failure of its outer member node B which is also a clusterhead, $clusterhead\ A$ fills the position by recruiting an available node and allows it to become a new clusterhead. Then, $clusterhead\ A$ sends the copy of the aggregated cluster data to
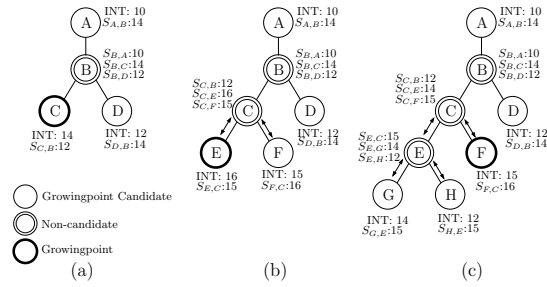
Fig. 4. Update of the interest level using network backbone

the new clusterhead so that the new clusterhead can know about the current BFS spanning tree information. Note that, unless the clusterhead and its all six neighboring clusterheads fail at the same time, a clusterhead failure can be recovered.

B. Coordinated Locomotion

Here we discuss three components of the Causataxis algorithm, 1) growingpoint selection and spanning tree formation, 2) grow and rot, 3) rate control of cluster data exchange.

1) Growingpoint Selection and Spanning Tree Formation: Candidates of the *GP* are clusterheads that have less than six neighboring clusterheads or that have a vacancy in the member positions. Among those candidates, the clusterhead that has the highest cluster interest value [3] becomes the *GP*. For the determination of the *GP*, clusterheads exchange data which include the information on the cluster interest value, the list of neighboring clusters, and member nodes in the cluster domain. Causataxis employs both global and local approaches to the *GP* selection problem to improve its performance.

The global approach to *GP* selection. In the global approach, a spanning tree constructed over clusterheads is employed for efficient cluster data exchanging, which is performed periodically with a tunable interval t. The cluster data exchange process consists of two phases: cluster data aggregation and cluster data dissemination. The cluster data aggregation phase is initiated by the root node of the spanning tree. The root broadcasts *Request Cluster Information* message using the tree structure, and the cluster data are collected and forwarded to the root by the tree structure (intermediate nodes aggregate and forward data from the children nodes to their parents).

---

[3]The cluster interest value can be defined in many ways. In this work, for simplicity, it is defined to be the average of the sensor readings of all nodes in a cluster.

The spanning tree structure is also used for the data dissemination phase where each node receives the aggregated data. Upon receiving the aggregated data, each clusterhead determines if it is eligible to be the new root and the *GP*. The clusterhead that has the highest cluster interest value becomes the new root of the spanning tree and the rest of clusterheads become either non-leaf or leaf nodes according to the breadth first search (BFS) algorithm (note that, analogous to the link-state algorithm, each node has knowledge of the global network topology, thus determination of the root and BFS spanning tree can be performed locally). The new root initiates the next cluster data exchange after t seconds. The new root also becomes the *GP*, and performs *grow* process by creating new clusters.

The local approach to *GP* selection. The local approach is executed in between two consecutive global cluster data exchange processes. In the local approach, spanning tree maintenance and *GP* selection is performed by using local neighborhood information taking advantage of the spatial locality of the interest value i.e., the next *GP* is most likely found near the current *GP* position.

To facilitate the local operations, each clusterhead in the tree maintains a data structure to denote the maximum cluster interest value for disjoint subgraphs of the BFS tree. More specifically, suppose *clusterhead V* has k edges and k neighboring clusterheads. Let e_i and N_i represent the i_{th} edge of *clusterhead V* and the neighboring clusterhead that shares the edge e_i with *clusterhead V* respectively (where $1 \leq i \leq k$). Also let S_{V,N_i} be the subgraph that contains N_i and its all descendant nodes assuming *clusterhead V* is the root of the tree. For example, in Figure 4(b), the subgraph $S_{B,A}$ has only one node *A*, and the subgraph $S_{B,C}$ has three nodes *C*, *E*, and *F*. Then, *clusterhead V* maintains its own cluster interest value (INT field) and the highest cluster interest value of each $subgraph\ S_{V,N_i}$. For example, Figure 4(a) shows the data structure of each clusterhead shortly after the global cluster data exchange. Here, the notation $(S_{X,Y} : Z)$ denotes the highest cluster interest value of the *subgraph* $S_{X,Y}$ is Z.

In Figure 4(b), *clusterhead C* makes its all outer member nodes become new clusterheads (*clusterhead E* and *clusterhead F*). Then, *clusterhead C* adds the new clusterheads to its BFS tree structure, and *clusterhead C* and the new clusterheads update the data structure as shown in Figure 4(b). From the updated data structure, *Clusterhead E* knows that it has the highest cluster interest value in the network (the maximum value at the subgraph through *clusterhead C*, 15, is less than its own interest value, 16), and becomes the *GP*.
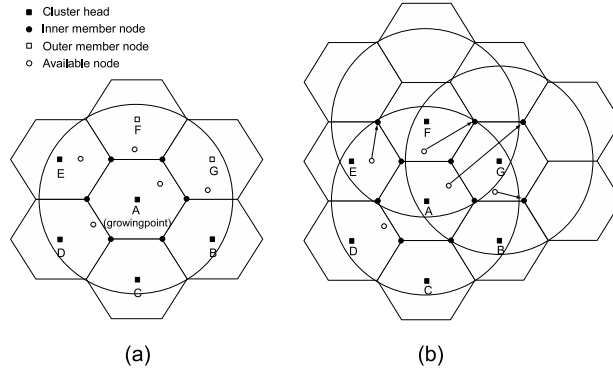
Fig. 5.   Growth of the network from Growingpoint

Note that the cluster data exchange occurs only at the subgraph that has *clusterhead C, E,* and *F*. In the data structure of *clusterhead B*, the maximum cluster interest value at the *subgraph* $S_{B,C}$ is 14. Therefore, if the *subgraph* $S_{B,C}$ has the value larger than 14, *clusterhead C* does not need to forward the new information to the rest of the network, because the new *growingpoint* should be at the *subgraph* $S_{B,C}$.

In the example shown in Figure 4 (c), the new clusterheads (G and H) know that they do not have a higher cluster interest value than the current *GP* (*clusterhead E*). Note that, in this case, *clusterhead E* forwards the updated information to *clusterhead C*, because the subgraph $S_{C,E}$ has a smaller cluster interest value than *clusterhead C* think that the subgraph has. Eventually, *clusterhead F* detects that it has the highest interest value after receiving the update of cluster information, and becomes the new *growingpoint*. Similarly to the previous case, the cluster data exchange is limited to the subgraph $S_{B,C}$.

If the new *GP* can not be selected at the subgraph $S_{B,C}$ due to no existence of a clusterhead that has a enough cluster interest value (larger than 12 in this example), the MSN waits for the next data exchange process where the new *GP* is selected by using the global information.

2) Grow and Rot: For the *grow* operation, the *GP* sends the *Create New Cluster* message to its outer member nodes. Figure 5 (a) shows the moment that *clusterhead A* becomes the *GP*. Suppose that *clusterhead A* has six outer member nodes and, among the outer member nodes, node B, C, D, and E are already clusterheads (the cluster domains of those clusterheads are omitted for simplicity). As shown in Figure 5 (b), *clusterhead F* and *clusterhead G* establish new cluster domains upon receiving the *Create New Cluster* message from *Clusterhead A*. Then, *Clusterhead F* and *clusterhead G* start filling the member node positions with available nodes in their cluster

9

domain. If a clusterhead can not find enough number of available nodes in its cluster domain, it recruits available nodes from other clusterheads. The clusterhead first checks with its neighboring clusterheads by sending a *Request for Nodes* inquiry. If it still can not find enough number of nodes, it sends a *Request for Nodes* inquiry via multi-hop fashion to the clusterheads that have reported available nodes during the cluster data exchange process. When a cluster receives a node from a non-adjacent cluster, a pipelining-like technique is used to reduce the latency that can be caused from a long distance traveling of the node. More specifically, the recruiting clusterhead first takes member nodes from its neighboring clusters, and asks the neighboring clusterheads to receive the available nodes.

The *GP* also recruits available nodes for its member positions. If the *GP* has not received a positive response to the *Request for Nodes* inquiry from the clusterheads that reported available nodes, the *GP* takes member nodes from a cluster that has the lowest cluster interest value among the leaf nodes in the BFS spanning tree.

A clusterhead initiates the *rot* process (in which its cluster dissolves) with a probability p_r, when the following conditions are satisfied: 1) the cluster has lost members due to low cluster interest value and the number of members becomes less than a threshold ρ, 2) the clusterhead is a leaf node in the spanning tree, and 3) the cluster is not expecting a node that is moving toward it. Note that the second and third conditions prevent the MSN from possible network partitioning.

The dissolving cluster sends member nodes to its neighboring clusterheads, if those member nodes are not shared with other clusterheads. The member nodes switch the availability status to *Available* so that other clusterheads recruit them.

3) Rate Control of Cluster Data Exchange, Grow, and Rot: If the current deployment of the MSN is close to the optimal in which the maximum total interest value can be obtained, the frequent cluster data exchange causes unnecessary communication overhead, which will also trigger unnecessary movements of MSN nodes through the *grow* and *rot* operations.

Causataxis enables the MSN to control the rate of the data exchange and the rate of the *grow* and *rot* operations by estimating the closeness to the optimal distribution of MSN nodes. More specifically, when the distribution of nodes in the MSN approaches to the optimal distribution, the root of the spanning tree increases the system parameter t which determines the frequency of the cluster data exchange. Similarly, when the distribution of MSN

nodes is near optimal, clusterheads have a small value of the rot probability p_r.

To determine how close the current the distribution of MSN nodes is to the optimal, the following observation is used. When the MSN is placed optimally such that the maximum total interest value is obtained, the boundary clusterheads likely have less interest values than clusterheads that are not boundary clusterheads. (Note that, in an ideal case, the boundary clusterheads form a contour line on which the cluster interest values are almost same, and the clusterheads inside of the network body have higher interest values than those on the boundary.)

Let b be the total number of total boundary clusterheads, and also let m the number of boundary clusterheads that have a higher cluster interest value than at least one clusterhead inside of the network. Then, a large value of $\frac{m}{b}$ indicates that the deployment of the MSN is far from the optimal, a small value of $\frac{m}{b}$ near optimal. Let t_{max} and t_{min} be the maximum and minimum value that the parameter t can have. Also, let δ_l and δ_h be the two threshold values for $\frac{m}{b}$. Then, the value of t is determined by using an exponential model such that the value of t is sharply increasing as the distribution of MSN nodes are close to the optimal state. That is,

$$t = \begin{cases} t_{max} & \text{if } 0 \leq \frac{m}{b} \leq \delta_l; \\[2mm] (t_{max} - t_{min})e^{-\lambda(\frac{m}{b} - \delta_l)} + t_{min} & \text{if } \delta_l \leq \frac{m}{b} \leq \delta_h; \\[2mm] t_{min} & \text{if } \delta_h \leq \frac{m}{b} \leq 1. \end{cases} \tag{1}$$

where λ is a system parameter.

The rot probability p_r is determined in a similar way in $[p_{r(max)}, p_{r(min)}]$.

IV. SWARM ALGORITHM

In this section we present our swarm robotics inspired algorithm for MSN deployment. In our swarm algorithm, we avoid maintaining persistent state on nodes. This way temporary errors are corrected in subsequent iterations, and we are able to provide a fault tolerant self healing algorithm.

The only form of communication used in this algorithm is periodic broadcasts containing information on location and perceived interest level of the node. Received packets are timestamped at receiver and old packets are discarded periodically, allowing the node to deal with changes in environment. For instance when a neighboring node moves after its broadcast, the received data is no longer accurate. As all nodes broadcast periodically, neighboring will receive new information to replace this old information.

11

The swarm has a global goal of effectively covering a region of interest. This translates to three objectives at the node level: keeping a minimum distance with neighbors, trying to stay connected with neighbors, and moving toward more interesting regions. We create virtual spring forces with relative direction of neighboring node as follows:

- Each node calculates, its neighbors' relative distance. If the distance is less than the minimum distance, the neighboring node applies a virtual force repulsion force that pushes the node away from it. Similarly if the distance is larger than the minimum distance, that neighboring node applies a virtual attraction force that pulls the node closer. We denote for sensing radius with r, distance between sensor nodes with d, spring constant of attraction force with k_a and sprint constant of repulsion force with k_r. Using these parameters, formation force $F_{formation}(d)$ can be defined as follows:

$$F_{formation}(d) = \begin{cases} k_r d & d < 2r \\ k_a d & o.w \end{cases}$$

- In order to move closer to interesting regions, the difference between interest levels is utilized. A neighboring node applies a virtual attraction force if it has a higher interest level and applies a virtual repulsion force if it has a lower interest. Here, k_i corresponds to interest weight, Δz corresponds to interest difference between nodes and Δz_{max} corresponds to maximum interest difference. We introduce maximum interest difference to deal with cases where interest gradient is too steep. We denote interest force with $F_{interest}(\Delta z)$ which is equal to:

$$F_{interest}(\Delta z) = \begin{cases} k_i \Delta z & \Delta z < \Delta z_{max} \\ k_i \Delta z_{max} & o.w \end{cases}$$

In each controller step, each node calculates the sum of all virtual forces acting on it. This total force is then used to modify current movement speed and direction. The magnitude of forces are modeled using simple spring model, where the applied force is linear with extension/compression of the spring. Different constants for compression and extension are used to get a better coverage. This allows the minimum distance constraint, which is crucial for good coverage, to be enforced more vigorously. We also introduce a virtual friction force, f_s, to reduce extra movement of the nodes. This force is proportional to the node speed and opposite to the movement direction.

Related Work. There has been a significant number of studies on swarm behavior using mobile robots. The
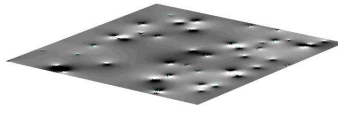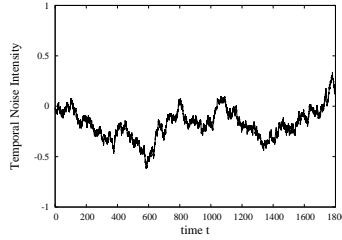
Fig. 6. Spatial Noise Model

Fig. 7. Temporal Noise Model

authors in [15]–[18] investigated flocking and swarm algorithms for coordination of mobile nodes. The work in [19] used three behavioral rules (collision avoidance, velocity matching, flock centering) for animated aggregate motion of the flock. The work in [18] showed that more sophisticated flocking behavior (including homing, exploring, and shepherding) can be achieved in complex environments by providing the flock with a global roadmap of the environment. [16] showed that a leaderless coordination of autonomous agents can be achieved by each agent periodically updating the move direction based on the average directions of its direct neighbors and itself. The simulation results in [16] showed that the direction of all agents can eventually converge without any centralized control or coordination. The authors of [17] also showed that the direction of a group of agents can be controlled by adding a special agent with a fixed direction and speed. The authors of [15] proposed three flocking algorithms for free flocking (flocking in free space) and constrained flocking (flocking in space with obstacles).

Gradient climbing problem using mobile sensor nodes has been investigated in [5]–[7]. The work in [6] provides a theoretical background to the gradient climbing strategy. In [5], authors study cooperative control of underwater vehicles for adaptive ocean sampling based on [6]. In [7], mobile sensor nodes estimate the location of the target based on the gradient toward the target. These work differ from our work since they focus on finding a maxima (minima) or acquisition of a single target rather than considering locomotion/deployment/coverage problem of mobile sensor networks. In contrast Causataxis addresses the efficient deployment and effective coverage problems, as well as ensuring connectivity of the MSN during relocation.

13

V. PERFORMANCE ANALYSIS

In this section, we compare the performance of Causataxis and the custom-tuned swarm algorithm for the unknown, unbounded, and noisy area monitoring problem. A set of MSN nodes (up to 200 nodes) is assumed to be initially deployed at a random position. The maximum moving speed of each MSN node is set to 10 m/s (or 36km/hour). Also, the reliable communication range and the sensing range of each MSN node are set to 100m and 25m respectively. Each MSN node can obtain the interest value $Z(x, y)$, which is associated with its current position (x, y). The embedded interest values used for simulation are shown in Figure 1 and Figure 2 (Note that in our work there is no limitation of the size of area. However, for visualization purpose, only a rectangle area of $800m \times 800m$ is shown).

In order to set up a simulation environment which is close to a real sensing environment, the spatial/temporal noise and the sensor reading error are also added to the area. More formally, let $n_s(x, y)$ and $n_t(t)$ be the spatial noise and temporal noise which are associated with the position (x, y) and time t respectively. Also let $\nu(x, y, t)$ be the actual sensor reading that an MSN node at a position (x, y) obtains at time t. Then, $\nu(x, y, t)$ is

$$\nu(x, y, t) = Z(x, y) + n_s(x, y) + n_t(t) + eZ(x, y) \tag{2}$$

$$L \leq n_s(x, y), n_t(t) \leq U \tag{3}$$

where e is the range coefficient of the sensor reading error, and L and U are the lower bound and the upper bound of the noise respectively.

The lower bound and the upper bound of the noise are determined using the maximum interest value (M) in the area i.e., $L = -\gamma M$ and $U = \gamma M$ where γ is the coefficient for the noise strength. The default value of γ is set to $\frac{1}{3}$.

Figure 6 shows the spatial noise model $n_s(x, y)$ that is used for the simulation. The degree of darkness of a position (x, y) represents the noise level at the position which results from the geographical properties or environmental characteristics (e.g. obstacles or plants). More specifically, the darkest position has a noise value of U, and the brightest position L.

The random walks process, which can model the thermal noise phenomena [20], is used for modeling temporal noise. Suppose that the temporal noise intensity is c at time t, then the noise intensity at time $t+1$ becomes either

$c + \alpha$ with probability $\frac{1}{2}$ or $c - \alpha$ with the same probability where α is a unit step. Figure 7 shows the variation of the temporal noise intensity which is used for the simulation (the value 1 represents the upper bound U of the noise, and -1 the lower bound L).

The range coefficient of the sensor reading error, e, is chosen randomly in $[-\frac{1}{2}, \frac{1}{2}]$, which results in up to 50% of the sensor reading error.

The Causataxis parameter values of t_{max} and t_{min} (the maximum and the minimum value of the cluster data exchange interval) are heuristically chosen to be 7 seconds and 1 second respectively. Similarly, the threshold values of δ_l and δ_h, which are used to determine the closeness to the optimal deployment, are set to 0.2 and 0.7 respectively.

For the swarm algorithm, we used the following parameter values determined experimentally: repulsion spring constant $k_r = 50$, attraction spring constant $k_a = 1$, interest spring constant is $k_i = 1$ and friction force $f_s(v) = 0.1v$ where v is the current speed of the node.

Figure 8 and Figure 9 show examples of initial deployment, locomotion, and settlement of Causataxis and swarm in the area that has one peak of interest value (Figure 1). As many as 100 Causataxis and Swarm nodes are initially deployed at the upper left corner of the area (the initial deployment position is predetermined for visualization purpose. However, for simulation the initial deployment position of the MSN is randomly chosen). As shown in Figure 8 (b) and Figure 9 (b), swarm has a faster initial expansion speed due to the exceedingly high repulsive force among swarm nodes as nodes are deployed at almost same position. Figure 8 (c) shows the locomotion of Causataxis toward more interesting region, and at time 300 seconds, Causataxis settles down with a near optimal distribution of MSN nodes. Figure 9 (c), (d), and (e) show the locomotion and settlement of swarm nodes.

In order to more systematically compare the performance of Causataxis and Swarm algorithm, following performance metrics are collected while varying parameters such as the gradient of the interest value, noise intensity, migration speed of phenomena, and the number of MSN nodes.

- Weighted coverage: the sensing area coverage weighted by the interest value.
- Coverage efficiency: the ratio of the weighted coverage to the approximated maximum possible weighted coverage.
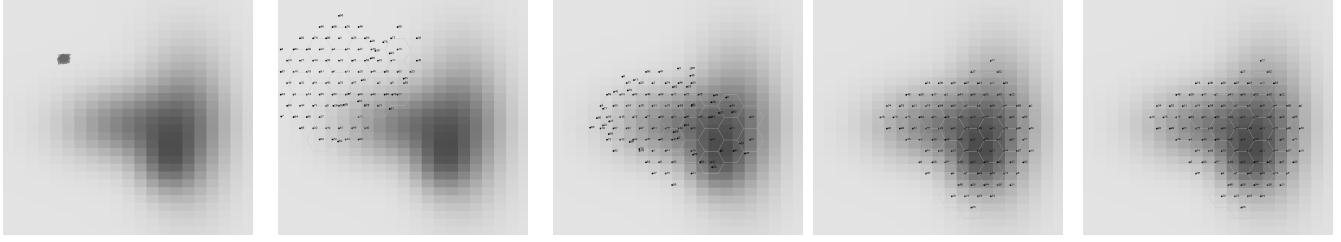
(a) Initial Deployment   (b) time: 50 seconds   (c) time: 150 seconds   (d) time: 300 seconds   (e) time: 450 seconds

(time: 0 seconds)

Fig. 8. Snapshots for initial deployment, locomotion, and settlement of Causataxis
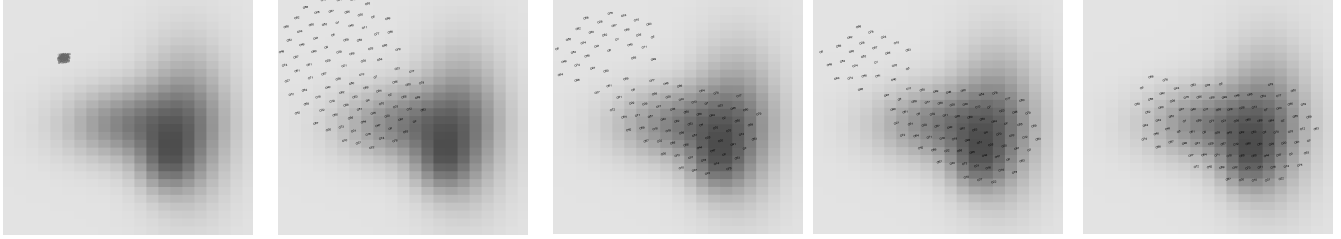
(a) Initial Deployment   (b) time: 50 seconds   (c) time: 150 seconds   (d) time: 300 seconds   (e) time: 450 seconds

(time: 0 seconds)

Fig. 9. Snapshots for initial deployment, locomotion, and settlement of Swarm algorithm

- Traveling distance: the total traveling distance of the MSN nodes

- Communication overhead: the total number of bytes of packets transmitted in the entire network.

Calculating the real coverage is challenging due to circular area of interest model. In the worst case this calculation requires computation of $O(2^n)$ terms corresponding to different intersection regions. Instead of this, we approximated the covered region by using a grid of squares. We approximate the average interest level in a square s with the average of interest values of its four corners and denote this with $z_{avrg}(s)$. Weighted coverage of a region S is sum of approximated average interest level of each square times their area. We denote weighted coverage with $W(S)$.

The maximum possible weighted coverage is approximated off-line. We first calculate maximum total sensing area, A_{max}, which is sum of sensing regions. We then find the set of squares (S^*) with total area equal to A_{max}, whose total coverage is maximum:
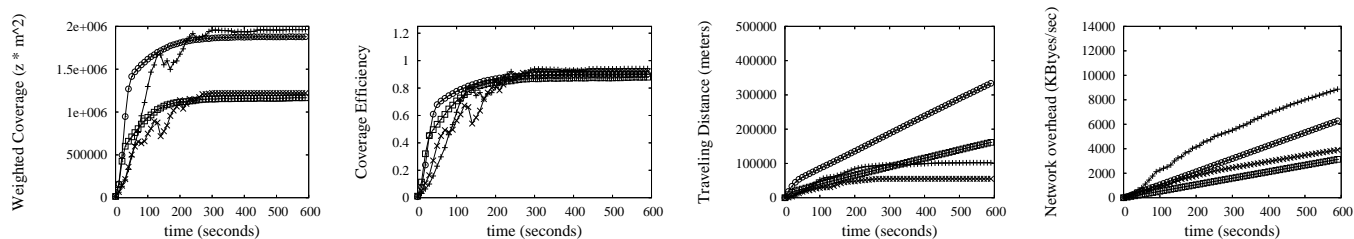
$$S^* = \arg\max_S \sum_{s \in S} Area(s) z_{avrg}(s)$$

Coverage efficiency, $C(S) = W(S)/W(S^*)$, is the ratio of weighted coverage to maximum weighted coverage.
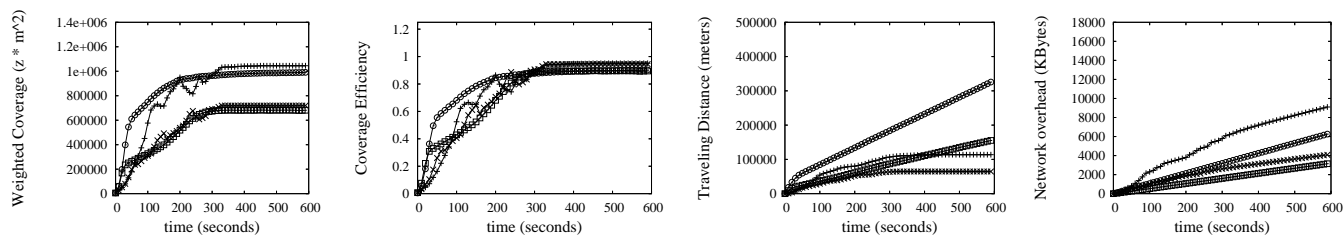
(a) Area Interest Coverage   (b) Coverage Efficiency   (c) Traveling distance   (d) Network Overhead

Fig. 10.   Plain Gradient, Area of simple interest. (×: Causataxis with 100 nodes, +: Causataxis with 200 nodes, □: Swarm with 100 nodes, ○: Swarm with 200 nodes)
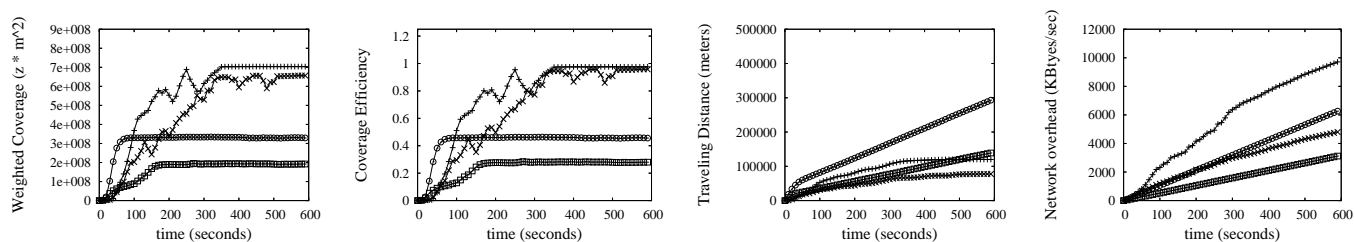

(a) Area Interest Coverage   (b) Coverage Efficiency   (c) Traveling distance   (d) Network Overhead

Fig. 11.   Plain Gradient, Area of complex interest.(×: Causataxis with 100 nodes, +: Causataxis with 200 nodes, □: Swarm with 100 nodes, ○: Swarm with 200 nodes)

Figure 10 and Figure 12 show the collected performance metrics over the time. Figure 10 compares the results where the Causataxis and swarm nodes are deployed in the area of one interest peak (Figure 1), and the phenomena of interest are stationary. As shown in Figure 10 (a) and (b), Causataxis show slightly better coverage and coverage efficiency than swarm. Figure 10 (a) and (b) also show that swarm has a fast initial coverage speed.

Figure 10 (c) and (d) show the cumulative traveling distance and the network overhead. As shown in Figure 10 (c), swarm nodes travel more than Causataxis nodes. The swarm nodes rely on movement to find the optimal position at which several virtual forces are in an equilibrium. This process results in a large movement in an unknown, noisy,

(a) Area Interest Coverage   (b) Coverage Efficiency   (c) Traveling distance   (d) Network Overhead

Fig. 12.   Sharp Gradient, Area of complex interest.(×: Causataxis with 100 nodes, +: Causataxis with 200 nodes, □: Swarm with 100 nodes, ○: Swarm with 200 nodes)

and dynamic environment. On the other hand, the use of the global information makes it possible for Causataxis nodes to relocate toward optimal positions without significant dithering and needless exploration. The use of the global information, however, causes a higher communication overhead as shown in Figure 10 (d). Two hundred Causataxis nodes transmit about 10,000 Kbytes of data through the wireless channel during the simulation period (600 seconds). However, the resulting data transmission rate, less than 140 Kbps (10,000 Kbytes $\times$ 8 /600seconds) for the entire network is still acceptable.

To approximate the energy consumptions of both schemes, the results of [21], [22] are used. In [22], the authors performed experiments to quantify the energy consumption for movement of mobile sensor nodes. According to the measurement data of the experiments in [22], a mobile sensor node uses 0.21 J (joule) to travel 1 inch (or 39.4 J/meter). Therefore, from Figure 10 (c), 200 Causataxis and swarm nodes use about 100000*39.4 J and 300000*39.4 J respectively. In [21], the authors derive the energy consumption model when a station transmits data in a IEEE 802.11 network. According to [21], a node uses less than 1.5 J to transmit 1 MBytes data. Therefore, from Figure 10 (d), 200 Causataxis and swarm nodes use about 1.5*10*2 J and 1.5*6*2 J respectively, assuming that the reception of a packet uses the same amount of energy as transmission. From this approximation, we can infer that node movement is the dominant factor in energy consumption and Causataxis is more energy efficient.

Figure 12 shows the case where the area of two interest peaks (Figure 2) is used. In order to test the area that has a large gradient of the interest value, the interest values are raised to power of 5, i.e. $Z'(x,y) = \{Z(x,y)\}^5$ is used. (The case of $Z(x,y)$ is shown in Figure 11, which shows almost same results as shown in Figure 10). As shown in Figure 12, Causataxis outperforms Swarm in terms of coverage and coverage efficiency. The sharp gradient indicates that most phenomena of interest are centered at the peak, and most region other than the center area has a relatively high noise and a low interest value. This indicates that the low value of the interest value to noise ratio degrades the performance of swarm.

Figure 13, Figure 14, and Figure 15 compare the performance of Causataxis and swarm with variation of values of parameters (noise intensity, phenomena migration speed, and the number of MSN nodes).

Figure 13 shows the simulation results where the noise strength varies. The X axis represents the noise strength parameter, γ, which determines the upper and lower bound of the noise in (3). Causataxis and swarm show almost
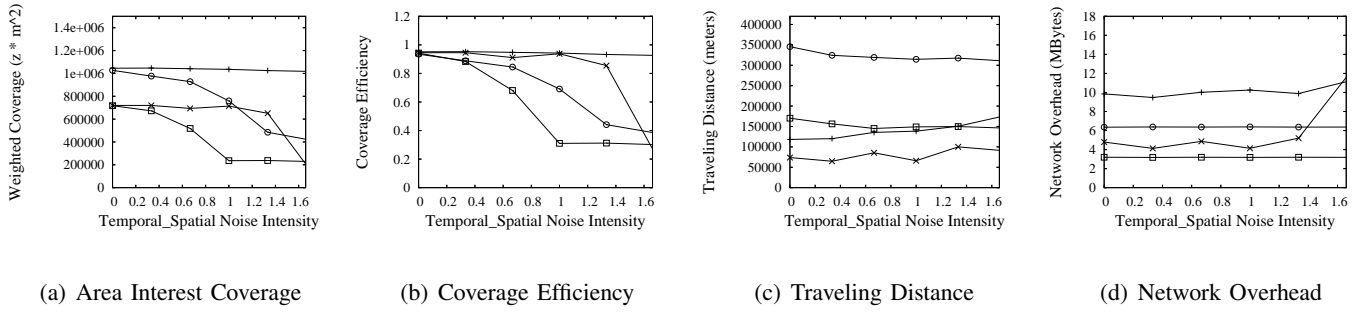
(a) Area Interest Coverage (b) Coverage Efficiency (c) Traveling Distance (d) Network Overhead

Fig. 13. Variation of temporal spatial noise intensity. (×: Causataxis with 100 nodes, +: Causataxis with 200 nodes, □: Swarm with 100 nodes, ○: Swarm with 200 nodes)

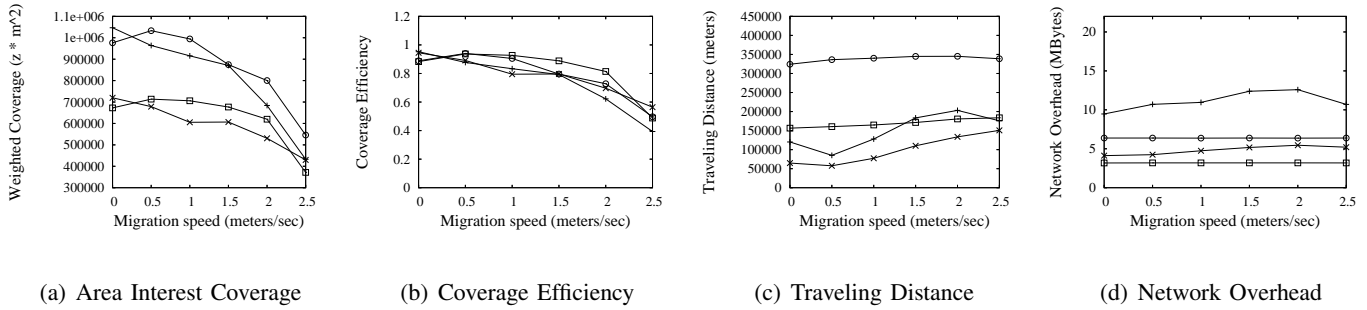(a) Area Interest Coverage (b) Coverage Efficiency (c) Traveling Distance (d) Network Overhead

Fig. 14. Variation of moving speed of the area of interest. (×: Causataxis with 100 nodes, +: Causataxis with 200 nodes, □: Swarm with 100 nodes, ○: Swarm with 200 nodes)

same coverage and the coverage efficiency when there is no noise. As the noise strength grows, however, the coverage and coverage efficiency of swarm sharply decreases due to the distortion of the local neighborhood information caused by the noise.

Figure 14 shows the performance results where the phenomena of interest migrates. In this simulation, the interest values embedded at the area move with the given speed (0 m/s to 2.5 m/s). In order to measure more accurate movement speed of the MSN, which follows the migration of the interest, the mobiles nodes are initially deployed close to the peak of the interest value. As shown in Figure 14 (a) and (b), both Causataxis and swarm lose the coverage and the coverage efficiency as the speed of migration increases. Swarm shows a better performance at
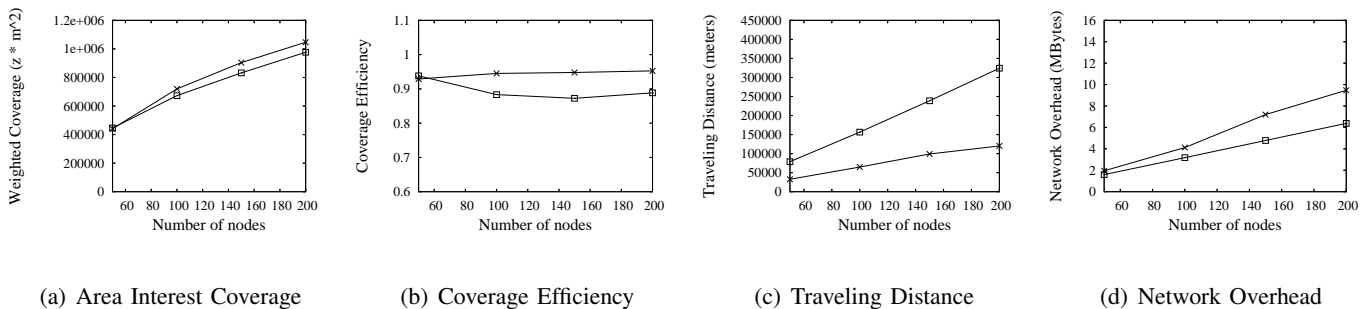

(a) Area Interest Coverage (b) Coverage Efficiency (c) Traveling Distance (d) Network Overhead

Fig. 15. Variation of number of nodes. (×: Swarm , □: Causataxis)

the high speed of migration (2.5m/s). When the migration speed is low (1 m/s), both Causataxis and swarm show an acceptable performance.

Figure 15 shows the case that the number of nodes varies. As the number of nodes grows, the weighted coverage of Causataxis and swarm increase. The traveling distance of swarm nodes increases more sharply as the number of nodes grows as shown in Figure 15 (c), while the communication overhead of Causataxis nodes increases with a higher slope as shown in Figure 15 (d).

VI. Concluding Remarks

In this paper, we have considered the problem of monitoring a changing physical phenomena in an open (large) area using MSN. To address this problem, we have proposed a novel distributed algorithm, Causataxis, that enables the MSN to efficiently relocate toward the more interesting regions and adjust its shape as the sensing environment changes. We have compared the performance of Causataxis with that of a custom-tuned swarm algorithm. The simulation results have showed that Causataxis outperforms the swarm algorithm in terms of sensing coverage, energy efficiency, and noise tolerance with a slightly high communication overhead. Compared to swarm approaches, Causataxis' ability to mitigate noisy environments, satisfy globally enforced constraints (such as network connectivity), and achieve energy-efficient MSN relocation make Causataxis a good candidate for many practical applications.

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Communications Magazine*, 2002.

[2] D. Estrin, R. Govindan, J. S. Heidemann, and S. Kumar, "Next century challenges: Scalable coordination in sensor networks," in *Mobile Computing and Networking*, 1999, pp. 263–270.

[3] Y. Zou and K. Chakrabarty, "Sensor deployment and target localization based on virtual forces," in *Proceedings of INFOCOM*, March 2003.

[4] G. Wang, G. Cao, and T. L. Porta, "Movement-assisted sensor deployment," in *IEEE INFOCOM Conference Proceedings*, March 2004.

[5] E. Fiorelli, N. Leonard, P. Bhatta, D. Paley, R. Bachmayer, and D. Fratantoni, "Multi-auv control and adaptive sampling in monterey bay," in *Proceedings of IEEE Autonomous Underwater Vehicles: Workshop on Multiple AUV Operations (AUV04)*, June 2004.

[6] P. Oğren, E. Fiorelli, and N. E. Leonard, "Cooperative control of mobile sensor networks: Adaptive gradient climbing in a distributed environment," *IEEE Transactions on Automatic Control*, vol. 49, no. 8, 2004.

[7] Q. Zhang, G. Sobelman, and T. He, "Gradient-driven target acquisition in mobile wireless sensor networks," in *The Second International Conference on Mobile Ad Hoc and Sensor Networks (MSN 2006*, 2006.

[8] P. De, A. Raniwala, S. Sharma, , and T. cker Chiueh, "Mint: A miniaturized network testbed for mobile wireless research," in *Proceedings of IEEE Infocom*, 2005.

[9] F. Lin, Y. Wang, and H. Wu, "Testbed implementation of delay/fault-tolerant mobile sensor network (dft-msn)," in *Proceedings of the Fourth Annual IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOMW 06)*, 2006.

[10] T. X. Brown, B. Argrow, C. Dixon, and S. Doshi, "Ad hoc uav ground network (augnet)," in *AIAA 3rd Unmanned Unlimited Technical Conference*, Chicago, IL, September 2004.

[11] K. Kotay, R. Peterson, and D. Rus, "Experiments with robots and sensor networks for mapping and navigation," in *International Conference on Field and Service Robotics*, Port Douglas, Australia, 2005.

[12] D. Koditschek, "Robot planning and control via potential functions," *Robotics Review I*, pp. 349 – 367, 1989.

[13] A. Howard, M. J. Mataric, and G. 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.

[14] M. Demirbas, , A. Arora, V. Mittal, and V. Kulathumani, "A fault-local self-stabilizing clustering service for wireless ad hoc networks," *IEEE Transactions on Parallel and Distributed Systems, Special Issue -Localized Communication and Topology Protocols for Ad Hoc Networks*, vol. 17, pp. 912 – 923, September 2006.

[15] R. Olfati-Saber, "Flocking for multi-agent dynamic systems: Algorithms and theory," *IEEE Transactions on Automatic Control*, vol. 51, pp. 401 – 420, March 2006.

[16] T. Vicsek, A. Czirok, E. B. Jacob, I. Cohen, and O. Schochet, "Novel type of phase transitions in a system of self-driven particles," *Physical Review Letters*, vol. 75, pp. 1226 – 1229, 1995.

[17] A. Jadbabaie, J. Lin, and A. S. Morse, "Coordination of groups of mobile autonomous agents using nearest neighbor rules," *IEEE Transactions on Automatic Control*, vol. 48, pp. 988 – 1001, June 2003.

[18] O. B. Bayazit, J.-M. Lien, and N. M. Amato, "Better group behaviors in complex environments using global roadmaps," in *Pacific Graphics*, 2002.

[19] C. W. Reynolds, "Flocks, herds, and schools: a distributed behavioral model," *Computer Graphics*, vol. 21, no. 6, pp. 25–34, July 1987.

[20] A. Papoulis and S. U. Pillai, *Probability, Random Variables and Stochastic Processes 4th Edition*. McGraw-Hill, pp. 435–436.

[21] M. Ergen and P. Varaiya, "Decomposition of energy consumption in ieee 802.11," in *The IEEE International Conference on Communications (IEEE ICC)*, Glasgow, Scotland, 2007.

[22] M. Rahimi, H. Shah, G. Sukhatme, J. Heidemann, and D. Estrin, "Studying the feasibility of energy harvesting in a mobile sensor network," in *The IEEE International Conference on Robotics and Automation*, Taipai, Taiwan, 2003, pp. 19–24.