Distributed Coverage Control and Data Collection With Mobile Sensor Networks

Minyi Zhong, *Member, IEEE*, and Christos G. Cassandras, *Fellow, IEEE*

***Abstract*—We study wireless sensor networks whose objective is to control the locations of mobile nodes so as to maximize the probability of detecting randomly occurring events in a mission space and to extract information from data sources, when detected, with maximal effectiveness. The control system for the mission is composed of three components executed in parallel on sensor nodes: coverage control, data source detection, and data collection. In order to maximize the joint detection probability of random events in a given mission space, we build upon a previously developed distributed gradient-based coverage control scheme to include three additional capabilities: allowing polygonal obstacles, including limited sensing field-of-view constraints, and preserving network connectivity through a provably correct algorithm using each node's routing information. In order to combine coverage and data collection, we formulate and solve a modified optimization problem with these two objectives. The interactions among the three components of the sensor network control system are discussed and simulation examples are presented to illustrate our results.**

***Index Terms*—Cooperative control, distributed optimization, distributed systems, sensor networks.**

I. Introduction

MANY civilian and military missions involve searching and collecting information in highly dynamic and potentially adversarial or hazardous environments. Examples include pollution detection, wildfire monitoring, search and rescue missions, reconnaissance, and surveillance. Thus, it is desirable to develop autonomous mobile sensor network systems which are capable of dynamic self-deployment and can replace human operators for potentially long unattended periods of operation. The performance of such a mobile sensor network generally depends on how its nodes are located within a "mission space," which gives rise to the fundamental problem of *coverage control* or *active sensing* [1]–[3]. In particular, nodes must be deployed so as to maximize event detection in the mission space while maintaining acceptable levels of communication and energy consumption. At the other end of the spectrum, once data sources are known, the problem becomes that of efficient *data collection* and there is a basic trade-off to be managed between these two objectives. In this paper, we build upon our previous work in distributed coverage control [4], [5] and develop a framework for providing an end-to-end solution to sensor network missions where mobile sensor nodes are deployed to detect and extract information from events occurring in a mission space. As illustrated in Fig. 1, we view a sensor network as a system that must perform the following three tasks simultaneously:

Task 1: Coverage Control: Sensor nodes perform motion control to reach an optimal deployment which maximizes the probability of event detection based on some (possibly none) prior knowledge of the event density over the mission space. After an event is detected, it is referred to as a *data source*.

Task 2: Data Source Detection: Sensor nodes actively run a procedure for detecting and locating emerging data sources. This usually returns binary results tainted with both false alarm and missed detection errors. Thus, we need to model such uncertainty in order to provide reliable information to Task 3.

Task 3: Data Collection: Sensor nodes gather information from detected data sources. To achieve maximal data collection quality, the nodes need to adjust their locations according to the known or estimated data source locations.

The interactions among these three tasks (see also Fig. 1) are important as they give rise to critical trade-offs that we will try to analyze and use as the basis for formulating and solving relevant optimization problems. We summarize these interactions as follows: *(i)* Task 1 aims to increase the probability of data source detection performed under Task 2. *(ii)* The history of detected data sources in Task 2 can be used to update the event density used in Task 1 for predicting where new data sources will appear in the future. *(iii)* Task 2 provides estimates of data source locations to Task 3. *(iv)* As a byproduct of Task 3, since nodes move closer to probable (but still unconfirmed) data source locations, the uncertainty in the data source occupancy map generated by Task 2 tends to be reduced. Observe that optimal coverage (under Task 1) and optimal data collection (under Task 3) have conflicting goals: when nothing is known about the data source locations in the mission space, then coverage control is the first step in the sensor network's operation; conversely, if there is always full knowledge of the data sources, then the system operates exclusively in data collection mode.

In the following, we provide some background information on the three tasks of Fig. 1 and describe our contributions toward providing solutions to them.

Coverage Control: The *static* version of coverage control involves positioning sensors without any further mobility and optimal locations can be determined by an off-line scheme akin to widely studied facility location optimization problems [6], [7], where the goal is to decide the optimal location of a facility in order to minimize the sum of weighted Euclidean distances

Manuscript received March 08, 2011; accepted June 24, 2011. Date of publication August 08, 2011; date of current version October 05, 2011. This work was supported in part by the National Science Foundation (NSF) under Grant EFRI-0735974, by AFOSR under grants FA9550-07-1-0361 and FA9550-09-1-0095, by the Department of Energy (DOE) under grant DE-FG52-06NA27490, and by ONR under grant N00014-09-1-1051. Recommended by Associate Editor I. Paschalidis.

The authors are with the Division of Systems Engineering and Center for Information and Systems Engineering, Boston University, Brookline, MA 02446 USA (e-mail: mzhong@bu.edu; cgc@bu.edu).

Digital Object Identifier 10.1109/TAC.2011.2163860

0018-9286/$26.00 © 2011 IEEE

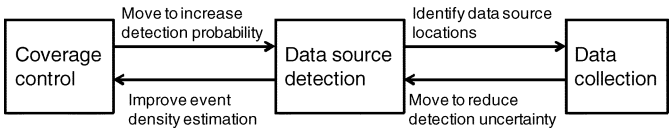

Fig. 1. Three main tasks of a sensor network and their interactions.

from this facility to several fixed demand points. In contrast to this problem and its variations, coverage control aims to find the optimal locations to "service" the whole region instead of a few discrete points in the region.

The *dynamic* version allows the coordinated movement of sensors, which may adapt to changing conditions in the mission space, typically deploying them into geographical areas with the highest information density. In [2], a decentralized coverage control algorithm is proposed based on centroidal Voronoi partitioning and a dynamic version of the Lloyd algorithm [8] has been used to iteratively find such a configuration. Other related work based on Voronoi partitions includes [9] and [10]. In [1] a coverage control scheme is used aiming at the maximization of target exposure in some surveillance applications, while in [11] and [12], heuristic algorithms based on potential fields and "virtual forces" are applied to push nodes away from each other and disperse them into the unoccupied areas in the mission space to enhance the coverage of a sensor network.

Methods based on partitioning the mission space assume that the sensing quality at a point in space is affected by only one sensor, usually the closest one, and overlook the fact that the overall sensing performance may be improved by sharing the observations made by multiple sensors with overlapping coverage area. In addition, many approaches assume uniform sensing quality and an unlimited sensing range, which is not the case for most sensing devices used in practice. A number of solution techniques are also based on a centralized controller, which is inconsistent with the distributed communication and computation structure of sensor networks. Moreover, the combinatorial complexity of the problem constrains the application of such schemes to limited-size networks. Finally, another issue that appears to be neglected is the movement of sensors, which not only impacts sensing performance but it also affects other quality-of-service aspects in a sensor network, especially those related to wireless communication: because of the limited on-board power and computational capacity, a sensor network is not only required to sense but also to collect and transmit data as well. For this reason, both sensing quality and communication performance need to be jointly considered when controlling the deployment of sensors. In order to address all these issues, a distributed coverage control algorithm was developed in [4] and [5] which uses a distance-dependent probabilistic sensing model and incorporates communication costs. The environment assumed in [5] does not take into account boundaries in the mission space and allows no obstacles. It also assumes omnidirectional sensors and deals with communication limitations by including a cost in the objective function, but does not explicitly guarantee network connectivity. One of the contributions of this paper is to provide these three extensions to the original distributed coverage algorithm in [5].

First, we will provide a distributed optimization solution to the coverage control problem which incorporates mission space boundaries and allows an arbitrary number of polygonal obstacles. This setting is similar to the classic Art Gallery problem [13], [14] where the goal is to ensure that every point in the art gallery, represented as a simple polygon, is "visible" by at least one omnidirectional security guard with unlimited sensing range and establish bounds on the minimum number of guards needed. In contrast to that setting, we consider a more general objective of maximizing the joint probability of detecting random events in the mission space with a given event density function. In addition, the nodes may have different sensing characteristics and do not have unlimited range. We formulate this problem and develop a gradient-based distributed coverage control scheme (first presented in [15]) which allows polygonal obstacles in the environment. Second, we address the issue of limited sensor field of view in coverage control (e.g., when a sensor is a video camera or a laser range finder). We model this limitation by a *sensing cone* which gives rise to discontinuities similar to those caused by obstacles. We then derive a more general coverage control algorithm which handles both obstacles and limited field of view involving additional controllable variables. Third, we study how the mobile nodes operating with asynchronous distributed optimization algorithms such as the coverage control algorithm we derive can preserve connectivity of the underlying communication network so as to exchange information either directly or indirectly. In [16], a decentralized power iteration algorithm is proposed to estimate the algebraic connectivity of a graph. In [17], each node maintains a local estimate of the network graph and a market-based auction mechanism is proposed so that deletion of a communication link will not disconnect the graph. The approach we take (introduced in [18]) utilizes the information in each node's local routing information to a base station, which is maintained by a wireless routing algorithm [19], [20] running in parallel with the optimization process.

Data Collection: After events are detected and confirmed by the sensor nodes, they become *data sources*. When data sources contain information that can be collected by a single visit, some of the sensor nodes can visit them by applying a multi-node reward maximization algorithm [21] and return to executing the coverage mission after collecting the reward. However, when the data sources generate valuable information over a long period of time (e.g., in a surveillance mission), sensor node motion control now has a long-term second task, which is to improve the quality of monitoring and information extraction from these detected data sources. This second task is formulated as an optimization problem which is combined with the coverage control problem discussed above to capture the trade-off between the two objectives.

Data Source Detection: A critical link between coverage control and data collection is the data detection process, which benefits from improved event detection probability and supplies a list of detected data sources to the data collection task. To achieve reliable data source detection with error-prone binary sensor measurements, we adopt the Bayes estimation idea commonly used in robotics [22], [23] and autonomous vehicle driving [24] to recursively integrate new sensor observations into a probabilistic *occupancy grid* map of the data sources. De-

tails of this mechanism are not covered in this paper but can be found in [25].

The rest of the paper is organized as follows. In Section II, we first formulate the coverage control problem and review the distributed optimization approach of [5] for a convex mission space with no obstacles. We then provide the analysis that enables us to incorporate polygonal obstacles and limited sensor field of view constraints. A modification of the algorithm is introduced in Section IIB to enhance its performance when high detection probability can be traded off against a more balanced overall space coverage. In Section IID a connectivity preservation algorithm is introduced and its correctness is verified. In Section III, we introduce data collection as a second objective in the motion planning of sensor nodes. We conclude the paper and outline some future work in Section IV.

II. Distributed Coverage Control

At the start of a sensor network mission, the locations of data sources are usually either unknown or uncertain. To increase the probability of detecting future or existing data sources, the sensor nodes solve a coverage control problem as formulated in [5] and [15], which we first concisely review in order to introduce the notation needed for the extensions to this problem presented later in this section.

We model the *mission space* $\Omega \subset \mathbb{R}^2$ as a non-self-intersecting polygon, i.e., a polygon such that any two non-consecutive edges do not intersect. An event density function $R(x) : \Omega \to \mathbb{R}$ captures the frequency of random event occurrences at some point $x \in \Omega$. We assume that $R(x)$ satisfies $R(x) \geq 0$ for all $x \in \Omega$ and $\int_\Omega R(x)\,dx < \infty$ and that when an event takes place, it will emit some signal which may be observed by at least some sensor nodes.

The mission space may contain *obstacles* which can interfere with the movement of sensor nodes and the propagation of event signals. We model the boundaries of these obstacles as m non-self-intersecting polygons properly contained in Ω and denote them by $P_j \subset \Omega$, $j = 1, \ldots, m$. Each P_j divides Ω into two disjoint regions, P_j's exterior and interior. The interior of P_j, denoted by $\mathring{P}_j$, is infeasible for the sensor nodes to navigate in. Thus, the overall feasible (or navigable) subspace of Ω is $F = \Omega \backslash \left(\mathring{P}_1 \cup \cdots \cup \mathring{P}_m\right)$. Equivalently, we may consider the mission space as the interior and boundary of a polygon with holes, each hole corresponding to an obstacle and the interior of the holes considered to be on the exterior of the polygon [26].

We will assume that $R(x) = 0$ for $x \notin F$ either because there is "nothing interesting" happening outside of F or because the sensor nodes ignore all points outside of F. In our coverage control problem, the number of sensor nodes deployed in Ω may be a fixed integer N or time-varying $N(t)$ if nodes are allowed to "die" or new nodes are occasionally introduced. When the number of nodes is N, their location is denoted by a $2N$-dimensional vector $\mathbf{s} = (s_1, \ldots, s_N)$ with $s_i \in F$, $i = 1, \ldots, N$.

Next, we discuss the sensing model used. If there is no obstacle and Ω is convex (thus, there is a clear line of sight between any two points in Ω), the probability that sensor node i detects an event occurring at $x \in F$ is denoted by $p_i(x, s_i)$. The received signal strength generally decays with $\|x - s_i\|$, the Euclidean distance between the source and the sensor. We represent this degradation by a monotonically decreasing differentiable function $p_i(x, s_i)$. An example of such a function is

$$p_i(x, s_i) = p_{0i} e^{-\lambda_i \|x - s_i\|} \tag{1}$$

where $p_{0i} \in (0, 1]$ and λ_i are positive constants. Notice that (1) is an omnidirectional model and can be viewed as a function of $\|x - s_i\|$.

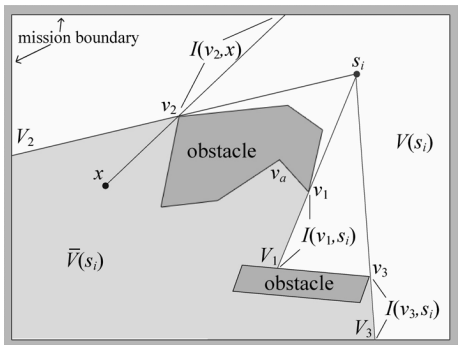

Fig. 2. Mission space with two polygonal obstacles.

Similar to the geometric terms in [14], we use the prefix ∂ to denote the boundary of a topological set, so that $\partial F = \partial\Omega \cup P_1 \cup \cdots \cup P_m$, and let T be the set of vertices of ∂F. A vertex v in T is a *reflex vertex* if the two edges incident at v form an angle strictly greater than π in the interior of F. In Fig. 2, for example, all vertices of the two obstacles, except v_a, are reflex vertices.

A point $x \in F$ is *visible* from $y \in F$ if the line segment defined by x and y is contained in F, i.e., $(\lambda x + (1 - \lambda) y) \in F$, for all $\lambda \in [0, 1]$. The *visibility polygon* $V(x) \subset F$ from a point $x \in F$ is the set of points in F visible from x. Let $\overline{V}(x) = F \backslash V(x)$ denote the invisibility polygon with respect to x. For example, in Fig. 2, the white area is the visibility polygon $V(s_i)$ of s_i and the grey area is $\overline{V}(s_i)$.

Let v be a reflex vertex and let $x \in F$ be a point visible from v. We define a set of points $I(v, x)$ as follows:

$$I(v, x) = \{q \in V(v) : q = \lambda v + (1 - \lambda) x,\ \lambda > 1\}. \tag{2}$$

To give a graphical interpretation of $I(v, x)$, consider a ray extending from v in the direction of $v - x$. This ray travels inside F until it hits ∂F at an *impact point*. The line segment between v and the impact point is $I(v, x)$ (see also Fig. 2).

Sensing Model: If $x \notin V(s_i)$, then the ability of node i to detect an event occurring at x will be reduced, possibly to zero. In general, an event signal might still be able to travel through obstacles, but its intensity will be attenuated more so than in open space depending on factors such as the size of the obstacle and the number of obstacles in the line of sight. Thus, the modified probability that sensor node i detects an event occurring at x is

$$\widehat{p}_i(x, s_i) = \begin{cases} p_i(x, s_i) & \text{if } x \in V(s_i) \\ \widetilde{p}_i(x, s_i) & \text{if } x \in \overline{V}(s_i) \end{cases} \tag{3}$$

where $\widetilde{p}_i(x, s_i) \leq p_i(x, s_i)$. For a totally "opaque" obstacle, we have $\widetilde{p}_i(x, s_i) = 0$. Note that for a fixed point x, $\widehat{p}_i(x, s_i)$

is not a continuous function of s_i. There is a jump whenever s_i crosses a line $I(v,x)$, where v is a reflex vertex visible from x. For example, in Fig. 2 assuming "opaque" obstacles, s_i cannot see x, so $\widehat{p}_i(x,s_i)=0$. But if s_i moves left and eventually reaches $I(v_2,x)$, then x becomes visible with respect to s_i and $\widehat{p}_i(x,s_i)$ jumps to a nonzero value $p_i(x,s_i)$.

Coverage Optimization Problem: Since multiple sensor nodes are deployed to cover the mission space, assuming node independence the joint probability that an event occurring at x is detected, denoted by $P(x,\mathbf{s})$, is given by

$$P(x,\mathbf{s})=1-\prod_{i=1}^{N}\left[1-\widehat{p}_i(x,s_i)\right] \tag{4}$$

and the optimization problem of interest is

$$\max_{\mathbf{s}} \int_{\Omega} R(x)\,P(x,\mathbf{s})\,dx \quad \text{s.t. } s_i\in F,\ i=1,\ldots,N. \tag{5}$$

Our goal is to develop a distributed algorithm to solve this optimization problem with each node performing a limited number of computations based on local information only. This eliminates the communication burden of transferring information to and from a central controller and the vulnerability of the whole system which would be entirely dependent on this controller. Since we have assumed $R(x)=0$ for $x\notin F$, we rewrite the objective function in (5) as

$$H(\mathbf{s})=\int_{F} R(x)\,P(x,\mathbf{s})\,dx. \tag{6}$$

If the mission space is convex and contains no obstacles, we have $F=\Omega$ and $x\in V(s_i)$ for any $x,s_i\in F$. Thus, $\widehat{p}_i(x,s_i)$ in (4) can be replaced by $p_i(x,s_i)$ and $P(x,\mathbf{s})$ is a continuous function of s_i if $p_i(x,s_i)$ is continuous. Therefore

$$\frac{\partial H(\mathbf{s})}{\partial s_i}=\int_{\Omega} R(x)\frac{\partial P(x,\mathbf{s})}{\partial s_i}dx. \tag{7}$$

As shown in [5], this derivative can be expressed as

$$\frac{\partial H(\mathbf{s})}{\partial s_i}=\int_{\Omega_i} R(x)\prod_{k\in\mathcal{B}_i}\left[1-p_k(x,s_k)\right]\frac{dp_i(x,s_i)}{dd_i(x)}\frac{s_i-x}{d_i(x)}dx \tag{8}$$

where $d_i(x)\equiv\|x-s_i\|$. If δ denotes the sensing radius of node i, then the node's region of coverage is represented by $\Omega_i=\{x: d_i(x)\le\delta\}$. $\mathcal{B}_i$ is a set of neighbor nodes with respect to i

$$\mathcal{B}_i=\{k:\|s_i-s_k\|<2\delta,\ \ k=1,\ldots,N,\ \ k\neq i\}.$$

In (8), all information is locally available to node i and the gradient can be used to determine the next waypoint on the ith mobile sensor's trajectory through

$$s_i^{k+1}=s_i^k+\eta_k\frac{\partial H(\mathbf{s})}{\partial s_i^k} \tag{9}$$

where k is an iteration index, and the step size sequence $\{\eta_k\}$ is selected according to standard rules (e.g., see [27]) when the convergence of motion trajectories must be guaranteed.

A. Mission Space With Obstacles

Node i's position in $\mathbb{R}^2$ is represented by $s_i=(s_{x_i},s_{y_i})$, where x and y in the subscript indicate the x-axis and y-axis component of a point in $\mathbb{R}^2$ respectively. For simplicity, we will drop the subscript i in s_i, s_{x_i}, s_{y_i} and focus on a typical node whose position is denoted by s. In addition, in the following discussion, we will assume that obstacles are fully opaque, i.e., $\widetilde{p}_i(x,s_i)=0$, for all i,x and s_i, in order to simplify the derivation. As already pointed out, for a given $x\in F$, $\widehat{p}(x,s)$ is not a continuous function of s and so neither is $P(x,\mathbf{s})$ in (4). Thus, we cannot interchange the order of differentiation and integration as in (7). A natural way to proceed is to separate the integration area F in (6) into the visibility polygon $V(s)$ and invisibility polygon $\overline{V}(s)$, and define

$$P_1(x,\mathbf{s})=1-\prod_{k=1,k\neq i}^{N}\left[1-\widehat{p}_k(x,s_k)\right]\left(1-p_i(x,s_i)\right)$$

$$P_2(x,\mathbf{s})=1-\prod_{k=1,k\neq i}^{N}\left[1-\widehat{p}_k(x,s_k)\right]\left(1-\widetilde{p}_i(x,s_i)\right)$$

so that

$$H(\mathbf{s})=\int_{V(s)} R(x)\,P_1(x,\mathbf{s})\,dx+\int_{\overline{V}(s)} R(x)\,P_2(x,\mathbf{s})\,dx. \tag{10}$$

Gradient Derivation: We will apply an extension of the Leibnitz rule [28] to evaluate $\partial H(\mathbf{s})/\partial s_x$, since in (10), both the integrand and the domain of integration are functions of s. For the first term in the right hand side of (10), we have

$$\frac{d}{ds_x}\int_{V(s)} R(x)\,P_1(x,\mathbf{s})\,dx=\int_{V(s)} R(x)\frac{\partial P_1(x,\mathbf{s})}{\partial s_x}dx+\int_{\partial V(s)} R(x)\,P_1(x,\mathbf{s})\left(u_x dx_y-u_y dx_x\right) \tag{11}$$

where (u_x,u_y) denotes the "velocity" vector at a boundary point $x=(x_x,x_y)$ of $V(s)$. The first term in the right hand side of (11) does not involve any variation of the integration domain and can be evaluated similarly to (8). To evaluate the second term, we need to introduce some more definitions related to the geometry of Fig. 2.

A reflex vertex v is an *anchor* of s if it is visible from s and $I(v,s)$ defined in (2) is not empty (see also [14]). Denote the anchors of s by $v_j, j=1,\ldots,Q(s)$, where $Q(s)$ is the number of anchors of s. An *impact point* of v_j, denoted by V_j, is the intersection of $I(v_j,s)$ and ∂F. Let $D_j=\|s-v_j\|$ and $d_j=\|V_j-v_j\|$. Define θ_j to be the angle formed by $s-v_j$ and the x-axis which satisfies $\theta_j\in[0,\pi/2]$.

For simplicity, we will abbreviate $I(v_j,s)$ by I_j in what follows. Observing that I_j is always on the boundary of $V(s)$ and $\overline{V}(s)$, we define n_j to be the unit normal vector on I_j which points to the interior of $V(s)$. Assuming that s is not on a reflex vertex, a polygonal inflection, or a bitangent (see definitions of these terms in [29]), on the boundary of the visibility polygon of s, only the points on I_j, $j=1,\ldots,Q(s)$, will have a nonzero

"velocity" when s is perturbed. Thus, for the second term in the right hand side of (11)

$$\int_{\partial V(s)} R(x) P_1(x,\mathbf{s})(u_x dx_y - u_y dx_x) = \sum_{j=1}^{Q(s)} \int_{I_j} R(x) P_1(x,\mathbf{s})(u_x dx_y - u_y dx_x). \quad (12)$$

Without loss of generality, we further assume that a point on I_j will only move along the x-axis under a small perturbation in s_x. Thus we have $u_y = 0$ and $u_x = -\|v_j - x\|/D_j$, for a point $x \in I_j$. Define

$$r = \begin{cases} \|v_j - x\| & \text{if integration is from } v_j \text{ to } V_j \\ \|V_j - x\| & \text{if integration is from } V_j \text{ to } v_j \end{cases}$$

and let dr be the new variable of integration in (12). Thus

$$u_x = \begin{cases} \frac{-r}{D_j} & \text{if integration is from } v_j \text{ to } V_j \\ \frac{-(d_j - r)}{D_j} & \text{if integration is from } V_j \text{ to } v_j \end{cases} \quad (13)$$

$$dx_y = -sgn(n_{jx}) \sin\theta_j \cdot dr. \quad (14)$$

Substituting (13) and (14) into (12) and using a change of variable ($r' = d_j - r$) reveals that the two cases in (13) give equivalent results. After evaluating the second term in (10) in a similar way and combining the results, we set $\rho_j(r) = (V_j - v_j)(r/d_j) + v_j$ and obtain

$$\frac{\partial H(\mathbf{s})}{\partial s_x} = \int_{V(s)} R(x) \frac{\partial P_1(x,\mathbf{s})}{\partial s_x} dx + \sum_{j=1}^{Q(s)} sgn(n_{jx}) \frac{\sin\theta_j}{D_j} \int_0^{d_j} R(\rho_j(r)) \Phi_{i,j}^N(r) p_i(\rho_j(r), s_i) r dr \quad (15)$$

where

$$\Phi_{i,j}^N(r) = \prod_{k=1, k\neq i}^{N} \left[1 - \widehat{p}_k(\rho_j(r), s_k)\right].$$

Proceeding in the exact same manner, we obtain a similar expression for $\partial H(\mathbf{s})/\partial s_y$. A more elaborate derivation of (15) using purely geometric arguments (instead of the Leibnitz rule) is also possible and can be found in [30].

Distributed Algorithm: The derivatives $\partial H(\mathbf{s})/\partial s_x$ and $\partial H(\mathbf{s})/\partial s_y$ can now be used in the motion control scheme (9) with the inclusion of a standard projection mechanism so that if a node points immediately into an edge of an obstacle or of the mission space boundary, we project $\partial H(\mathbf{s})/\partial s_i$ onto that edge, thus forcing the node to "slide" along the associated motion constraint. As in the case of no obstacles, $\partial H(\mathbf{s})/\partial s_i$ can be evaluated using only information locally available at node i. We can then rewrite (15) as

$$\frac{\partial H(\mathbf{s})}{\partial s_{x_i}} = \int_{V(s)\cap\Omega_i} R(x) \prod_{k\in\mathcal{B}_i} [1 - \widehat{p}_k(x, s_k)] \frac{dp_i(x, s_i)}{dd_i(x)} \frac{(s_i - x)_x}{d_i(x)} dx + \sum_{j\in\Gamma_i} sgn(n_{jx}) \frac{\sin\theta_j}{D_j} \int_0^{z_j} R(\rho_j(r)) \widehat{\Phi}_{i,j}^N(r) p_i(\rho_j(r), s_i) r dr \quad (16)$$

where $\Gamma_i = \{j : D_j < \delta,\ j = 1, \ldots, Q(s_i)\}$; $z_j = \min(d_j, \delta - D_j)$ and we define

$$\widehat{\Phi}_{i,j}^N(r) = \prod_{k\in\mathcal{B}_i} \left[1 - \widehat{p}_k(\rho_j(r), s_k)\right].$$

The computation of the integrals in (16) is quite involved. Thus, we resort to the same mission space discretization as in in [5], which reduces the evaluation to a worst-case computation of order $O(N_{\mathcal{B}} W^2 + |\Gamma_i| N_{\mathcal{B}} U)$ where $N_{\mathcal{B}} - 1$ is the number of neighbors of i, while W and U are controllable resolution parameters to discretize the surface integration and line integration, respectively.

Recall that in the derivation of (15) we assumed that the controllable node location s_i does not coincide with a reflex vertex, a polygonal inflection, or a bitangent, at which points $H(\mathbf{s})$ is generally not differentiable. To take these points into account, one can modify the standard gradient-based algorithm in (9) resorting to subgradient algorithms (e.g., [31]) or *bundle methods* [32] which aggregate the subgradient information in the past iterations. Whereas such nonsmooth optimization methods are invaluable in optimization problems where the points of nondifferentiability are hard to determine in advance, the geometric properties of the coverage problem greatly simplify this concern since it is easy to detect when s_i^k is in the vicinity of reflex vertices, polygonal inflection points, or bitangents. Moreover, the goal of a sensor network's mission does not stop with coverage and is constantly updated depending on new data sources found and on the parallel effort to optimize the data collection process. Thus, the overall problem goes beyond seeking a global optimum for the coverage component of the problem alone.

B. Modified Coverage Objective for Balanced Detection

As defined in (5), the coverage objective function aims at maximizing the joint event detection probability without considering the issue of maintaining a balance between a region which may be extremely well covered and others which may not. As a result, an optimal coverage solution may lead to a part of the mission space having a detection probability near 1, while other parts are covered with a disproportionately small detection probability. In order to address this issue, we introduce a modification to the objective function as follows:

$$H_M(\mathbf{s}) = \int_\Omega R(x) M(P(x,\mathbf{s})) dx \quad (17)$$

where $M(P) : [0,1] \to \mathbb{R}$ is a (possibly piecewise) differentiable concave non-decreasing function of the joint detection probability P. Clearly, $M(P)$ may be selected so that the same amount of marginal gain in $P(x,\mathbf{s})$ is assigned a higher reward at a lower value of P. Letting

$$\Psi = M(1 - \Phi_{i,j}^N(r)[1 - p_i(\rho_j(r))]) - M(1 - \Phi_{i,j}^N(r))$$

and using $M'(P)$ to denote the derivative of $M(P)$, the gradient of $H_M(\mathbf{s})$ is given by

$$\frac{\partial H_M(\mathbf{s})}{\partial s_x} = \int_{V(s)} R(x) M'(P_1(x,\mathbf{s})) \frac{\partial P_1(x,\mathbf{s})}{\partial s_x} dx \quad (18)$$

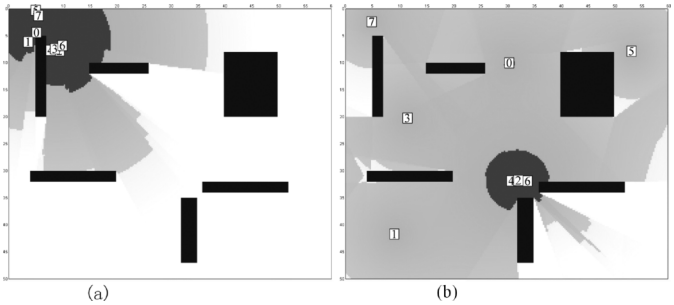

Fig. 3. Coverage control using the sensor model (1), $p_{0i} = 1$, $\lambda_i = 0.08$, 8 nodes deployed. (a) $H(\boldsymbol{s}) = 757.3$. (b) $H(\boldsymbol{s}) = 1832.6$.

$$+\sum_{j=1}^{Q(s)} sgn\left(n_{jx}\right)\frac{\sin\theta_j}{D_j}\int_0^{d_j} R\left(\rho_j\left(r\right)\right)\Psi r dr$$

with a similar expression for $\partial H_M\left(\mathbf{s}\right)/\partial s_y$. Comparing the second term on the right hand side of (18) and that of (15), we can see that $\Phi_{i,j}^N(r)p_i\left(\rho_j\left(r\right),s_i\right)$ is replaced by Ψ, which actually reduces to $\Phi_{i,j}^N(r)p_i\left(\rho_j\left(r\right),s_i\right)$ if we let $M\left(P\right)=P$. Obviously, using $\partial H_M\left(\mathbf{s}\right)/\partial s_i$ instead of $\partial H\left(\mathbf{s}\right)/\partial s_i$ to drive the motion control (9) will lead to different local optima. One can use $H(\mathbf{s})$ to guide the nodes to such a point and assess whether the coverage is sufficiently balanced. If not, switching to $H_M(\mathbf{s})$ induces the nodes to move to a different stationary configuration at which, interestingly, $H(\mathbf{s})$ may in fact be higher than before.

Fig. 3 shows snapshots of an optimal coverage deployment trajectory generated under (9) with the gradient evaluated using (15) in an interactive Java-based simulation environment which we have developed and may be found at *http://codes-color.bu.edu/simulators.html.* In this example, there are eight sensor nodes in a bounded mission space with uniform event density. The dark polygons represent obstacles which are totally opaque to the sensors. The numbered small white rectangles indicate the locations of the nodes, all initially starting at the upper-left corner. The mission space is color-coded from darker (purple) to lighter (white) as the detection probability decreases. Note that the equilibrium configuration includes three nodes (2, 4 and 6) which are very close to each other; local optimality prevents them from navigating into the lower right corner, which is only partially covered. If the modified objective function (17) is used with all other settings unchanged, the nodes spread more evenly as shown in Fig. 4(a). After an equilibrium is reached using (17), we change the objective function to the original one and obtain the coverage of Fig. 4(b) where the value of $H\left(\mathbf{s}\right)$ achieved is higher than that in Fig. 3.

Fig. 5 shows snapshots of an optimal coverage deployment trajectory in a maze-like environment, which is harder to cover.

In addition to extensive simulation experiments, we have conducted some tests of the coverage control algorithm in a laboratory setting with Khepera III mobile wireless robots and an overhead video camera. Fig. 6(a) shows the final node deployment in one of those tests. If we compare it with the simulation result obtained under a similar obstacle configuration (shown in Fig. 6(b)), the results are very close and both achieve a good coverage given the limited number of the nodes available.

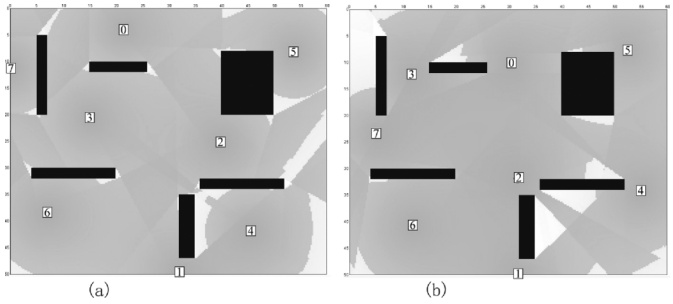

Fig. 4. (a) Modified coverage objective used. (b) Original objective used after reaching (a). (a) $H(\boldsymbol{s}) = 1775.7$. (b) $H(\boldsymbol{s}) = 1908.7$.

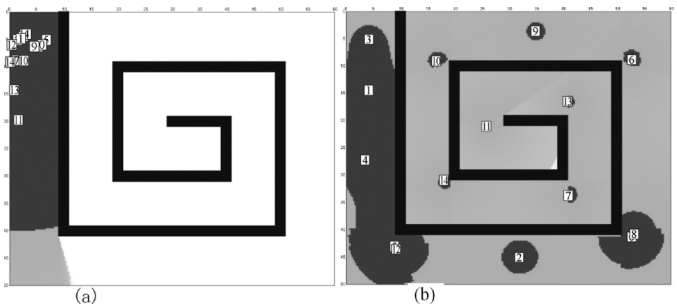

Fig. 5. Coverage control of maze with $\lambda_i = 0.05$, 15 nodes. (a) $H(\boldsymbol{s}) = 452.3$. (b) $H(\boldsymbol{s}) = 2332.0$.

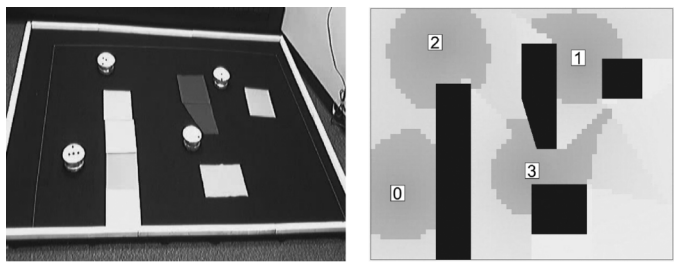

Fig. 6. (a) is the final node configuration in a test with four Khepera III robots. The color blocks represent obstacles and the mission boundary is marked with white lines. An overhead camera is used to to capture the robots' positions in this indoor laboratory setting. (b) is the simulation result in an environment similar to that of (a). The final node configurations in these two experiments are very close.

C. Limited Sensor Field of View (FOV)

So far, an omnidirectional sensing capacity model is assumed. In reality, many types of sensors (e.g., video cameras) only have a limited field of view (FOV). We model this limited FOV with a *sensing cone*, which is defined by the sensor location s, a left edge $\overline{E}_L$ (a ray starting from s and extending to infinity) and a right edge $\overline{E}_R$. The angle formed by the left (right) edge of the sensing cone and the positive direction of the x axis is denoted by θ_L (θ_R). Define $e_L = \arg\min_{q\in\{\overline{E}_L\cap\partial F\}}\|q-s\|$ (the point where $\overline{E}_L$ first hits an obstacle or boundary) and $e_R = \arg\min_{q\in\{\overline{E}_R\cap\partial F\}}\|q-s\|$. Let E_L (E_R) denote the line segment between s and e_L (e_R), i.e., $E_L = \{q|q=\lambda e_L+(1-\lambda)s, \lambda\in[0,1]\}$. In addition, for simplicity, we assume that all sensors have the same and fixed FOV (fixed angle $\angle e_L s e_R$).

We now add θ_L to each sensor node's state vector as a new decision variable and redefine $\mathbf{s} = ((s_1,\theta_{L1}),\ldots,(s_N,\theta_{LN}))$. Given (s,θ_L) for a typical node, its sensing cone, θ_R, e_L, e_R, E_L and E_R can all be determined. Denoting a node's sensing cone by $C(s,\theta_L)$, we have a new sensing model

$$\widehat{p}_i(x,s_i,\theta_{Li}) = \begin{cases} p_i(x,s_i) & \text{if } x \in V(s_i) \cap C(s_i,\theta_{Li}) \\ 0 & \text{otherwise} \end{cases} \tag{19}$$

E_L and E_R introduce discontinuities to the derivative similar to those introduced by $I(v,s)$ defined in (2) where v is an anchor of s. Thus we can extend the result in (15), and get the new partial derivative for s_x

$$\begin{aligned}\frac{\partial H(\mathbf{s})}{\partial s_{x_i}} &= \int_{V(s_i)\cap C_i} R(x)\frac{\partial P_1(x,\mathbf{s})}{\partial s_{x_i}}dx \\ &+ \sum_{j=1}^{\widehat{Q}_i} sgn(n_{jx})\frac{\sin\theta_j}{D_j}\int_0^{d_j} R(\rho_j(r))\Phi_{i,j}^N(r)p_i(\rho_j(r),s_i)rdr \\ &- \int_{E_{Li}} R(\rho_j(r))\Phi_{i,j}^N(r)p_i(\rho_j(r),s_i)\sin\theta_L rdr \\ &+ \int_{E_{Ri}} R(\rho_j(r))\Phi_{i,j}^N(r)p_i(\rho_j(r),s_i)\sin\theta_R rdr\end{aligned}$$

with a similar expression for $\partial H(\mathbf{s})/\partial s_{y_i}$, where $C_i = C(s_i,\theta_{Li})$ and $\widehat{Q}(s_i)$ is the number of anchors of s_i located in C_i. The derivative with respect to the angle of the node's FOV is obtained as

$$\begin{aligned}\frac{\partial H(\mathbf{s})}{\partial \theta_{L_i}} &= \int_{E_{L_i}} R(\rho_j(r))\Phi_{i,j}^N(r)p_i(\rho_j(r),s_i)rdr \\ &- \int_{E_{R_i}} R(\rho_j(r))\Phi_{i,j}^N(r)p_i(\rho_j(r),s_i)rdr.\end{aligned} \tag{20}$$

Observe that in (20) if only the rotation direction of a node's sensor is needed, we can simply compare the length of E_R and E_L and let the sensor turn toward the longer side. Similar to (9), at each iteration, θ_{L_i} is updated according to

$$\theta_{L_i}^{k+1} = \theta_{L_i}^k + \zeta_k\frac{\partial H(\mathbf{s})}{\partial \theta_{L_i}} \tag{21}$$

where ζ_k is the step size and $\theta_{L_i}^k$ denotes the value of θ_{L_i} at the kth iteration. As in (9), the convergence of (21) requires an appropriately selected step size sequence. As already pointed out, convergence is not always desirable if the mission space is not stationary and one needs to track changes in the density function or the addition/removal of sensor nodes. One may trade off oscillatory behavior around local optima for the benefit of reacting to such changes. It is also obvious that the addition of (21) to (9) increases the possibility of local optima in solving the coverage problem.

An example of coverage control with sensor nodes that have a limited FOV is shown in Fig. 7 where four nodes have a 90° FOV under two different initial starting points. Observe that the final node configuration could be very different depending on initial conditions.

D. Connectivity Preservation

We have thus far assumed that nodes exchange information over a connected communication network. However, when nodes are mobile and wireless, this assumption is often violated unless there is a mechanism to explicitly preserve connectivity. For example, in coverage control missions, sensor nodes tend to spread out to cover remote regions; this increases hop distance and often disconnects critical links. In this section, we address the issue of connectivity preservation which is essential in the context of our overall sensor network framework in Fig. 1. In particular, we seek to ensure the following two requirements: *(i)* The distance between two communicating nodes is less than a certain threshold C. This is to maintain link quality (i.e., a sufficiently high SNR). We define the following Boolean variable to indicate whether two nodes i and j satisfy this requirement (henceforth s_i will be interpreted as node i's 2-D location):

$$c_1(s_i,s_j) = \begin{cases} 1, & \|s_i - s_j\| \le C \\ 0, & \text{otherwise} \end{cases}$$

(ii) In the presence of obstacles [15], line-of-sight between two communicating nodes is maintained. We define the Boolean variable

$$c_2(s_i,s_j) = \begin{cases} 1, & as_i + (1-\alpha)s_j \in F \text{ for all } \alpha \in [0,1] \\ 0, & \text{otherwise} \end{cases}$$

where F denotes the free portion of the mission space unoccupied by obstacles. Links satisfying *(i)* and *(ii)* are termed *strong* links and we define $c(s_i,s_j) = c_1(s_i,s_j)\cdot c_2(s_i,s_j)$ so that i and j form a *strong* link if and only if $c(s_i,s_j)=1$.

Let us represent the network of mobile nodes by a graph $\mathcal{G}(\mathbf{s}) = (\mathcal{N},\mathcal{E}(\mathbf{s}))$, where $\mathcal{N} = \{0,1,\ldots,N\}$ is the set of node indices including the base station denoted by 0, and $\mathcal{E}(\mathbf{s}) = \{(i,j) : i,j \in \mathcal{N}, i \neq j, c(s_i,s_j) = 1\}$, which is the set of all *strong* links. Over $\mathcal{G}(\mathbf{s})$, let $\overline{\Pi}_i$ be the set of all possible loop-free paths from node i to 0. If $\overline{\Pi}_i \neq \emptyset$ for all $i \in \{1,\ldots,N\}$, the graph $\mathcal{G}(\mathbf{s})$ is *connected*. Our goal is to preserve this property.

We assume that the network operates with a distributed routing algorithm in parallel with our distributed optimization algorithm. The routing algorithm proactively generates and maintains a set of paths, denoted by $\widehat{\Pi}_i$, for each node i to forward data to 0. Note that the routing algorithm is not restricted to use only links in $\mathcal{E}(\mathbf{s})$. Let $\Pi_i = \overline{\Pi}_i \cap \widehat{\Pi}_i$ be the set of paths in $\widehat{\Pi}_i$ which contains only *strong* links. Denote the kth path in Π_i by $\pi_{i,k}$, represented by a *sequence*

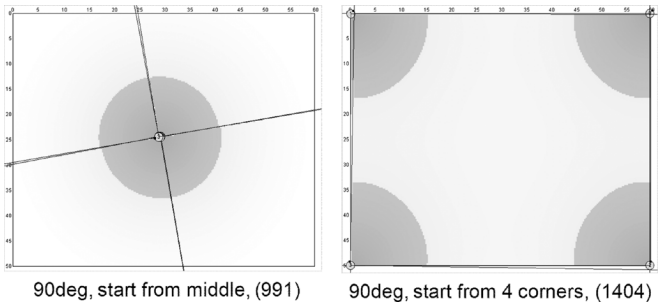

Fig. 7. Coverage with sensor nodes having different FOV size. Objective function values after convergence are shown in parentheses. Lines extending from sensor nodes indicate edges of their sensing cone. Green indicates good coverage while yellow or white indicate poor coverage.

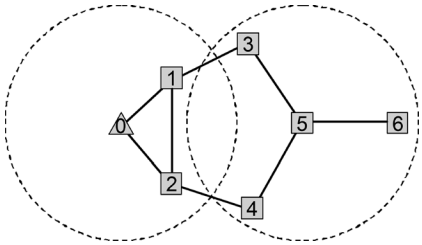

Fig. 8. Base station is labeled 0. Desired communication range is marked for node 0 and 5. *Strong* links are indicated by lines between nodes.

(ordered set) which contains all the nodes on this path from i and 0 (including i and 0) ordered by their hop counts. If we use $\pi_{i,k}^{j}$ to denote the jth element in $\pi_{i,k}$, then node $\pi_{i,k}^{j}$ is $j-1$ hops away from node i on the path $\pi_{i,k}$. We require that the path be loop-free, thus $\pi_{i,k}$ cannot contain duplicate elements, i.e., $\pi_{i,k}^{j} \neq \pi_{i,k}^{l}$, if $j \neq l$. Since $\overline{\Pi}_i$ only uses links in $\mathcal{E}(\mathbf{s})$, $c(\pi_{i,k}^{j}, \pi_{i,k}^{j+1}) = 1$, for all i, j, k. For example, in Fig. 8, $\overline{\Pi}_5 = \{\{5,4,2,0\}, \{5,3,1,0\}, \{5,3,1,2,0\}, \{5,4,2,1,0\}\}$ and depending on the result of the routing algorithm, Π_5 is some subset of $\overline{\Pi}_5$. The routing algorithm also maintains at node i its *upstream* (further from the base station 0) node set, denoted by $\mathcal{U}_i = \cup_{j,k} \omega_i(\pi_{j,k})$ where

$$\omega_i(\pi_{j,k}) = \begin{cases} \pi_{j,k}^{l-1}, & \text{if } i \in \pi_{j,k}, i \neq j \text{ and } i = \pi_{j,k}^{l} \\ \emptyset, & \text{else} \end{cases}.$$

In addition, it maintains the *downstream* node set $\mathcal{D}_i = \{j : i \in \mathcal{U}_j, j \in \{0, \ldots, N\}\}$.

Before proceeding, we also define a projection of $x \in \mathbb{R}^2$ on a set $\mathcal{S} \subset \mathbb{R}^2$ as $P_{\mathcal{S}}(x) = \arg\min_{y \in \mathcal{S}} \|x - y\|$. Next, let $\mathcal{X}(s) = \{x : x \in \mathbb{R}^2, c(x,s) = 1\}$ be the region where a *strong* link with s can be established. Finally, for simplicity, henceforth s_j stands for $s_j(k)$ unless expressly specified otherwise, where k indexes state update times t_k.

Algorithm 1:

When node i makes a location update at t_k, using (9), it takes the following steps:

1) Using (9), generate a candidate of the next location: $\hat{s}_i$.
2) If for all $j \in \mathcal{D}_i$, $c(\hat{s}_i, s_j) = 0$, go to Step 3; else, go to Step 5.
3) If there exists a node $j \neq i$ such that $s_j \in \mathcal{X}(\hat{s}_i)$ and there exists a path $\pi_{j,l} \in \Pi_j$, such that $i \notin \pi_{j,l}$, go to Step 5; else, go to Step 4.
4) Select some $j \in \mathcal{D}_i$ and project $\hat{s}_i$ on $\mathcal{X}(s_j)$. Redefine the result $P_{\mathcal{X}(s_j)}(\hat{s}_i)$ as $\hat{s}_i$.
5) If for all $j \in \mathcal{U}_i$, $c(\hat{s}_i, s_j) = 1$, go to Step 7; else, go to Step 6.
6) If for all $j \in \mathcal{U}_i$ such that $c(\hat{s}_i, s_j) = 0$ there exists a path $\pi_{j,l} \in \Pi_j$, such that either $i \notin \pi_{j,l}$ or $i \in \pi_{j,l}$ and $c(\hat{s}_i, s_r) = 1$ with $r = \omega_i(\pi_{j,l})$, go to Step 7; else $s_i(k+1) = s_i(k)$ and skip Step 7.
7) $s_i(k+1) = \hat{s}_i$.

It should be pointed out that in Steps 3 and 5, node i has to communicate with nodes in $\mathcal{X}(\hat{s}_i)$ and $\mathcal{U}_i$ respectively. In Step 4, when the set $\mathcal{X}(s_j)$ is a disk, the projection is straightforward. When $\mathcal{X}(s_j)$ is nonconvex due to obstacles, the projection involves finding the closest point on some line segments and some circular arcs to $\hat{s}_i$.

To show the validity of this algorithm, we define some extra notation. When $i \in \pi$, we can divide path π into two sub-paths separated by i. Let $\beta_i(\pi)$ denote the first sub-path, which does not contain i, and $\gamma_i(\pi)$ denote the second sub-path, which contains i. We use the binary operator $+$ to concatenate two paths into a new path. If $\pi_a = \{\pi_a^1, \ldots, \pi_a^m\}$ and $\pi_b = \{\pi_b^1, \ldots, \pi_b^n\}$, then $\pi_a + \pi_b = \{\pi_a^1, \ldots, \pi_a^m, \pi_b^1, \ldots, \pi_b^n\}$.

Theorem 1: Assuming only one node performs a state update at any given time and $\Pi_j \neq \emptyset$ for all $j \in \{1, \ldots, N\}$ before the state update, an iteration of Algorithm 1 preserves the connectivity of $\mathcal{G}(\mathbf{s})$.

Proof: Let us assume that node i updates at t_k using Algorithm 1 and $\Pi_j(k) \neq \emptyset$ for all $j \in \{1, \ldots, N\}$. We will prove that $\overline{\Pi}_j(k+1) \neq \emptyset$ for all $j \in \{1, \ldots, N\}$.

We first rewrite the node set so that $\{1, \ldots, N\} = \mathcal{N}_1 \cup \mathcal{N}_2 \cup \{i\}$ where $j \in \mathcal{N}_1$ if there exists a path $\pi_{j,l} \in \Pi_j(k)$ such that $i \notin \pi_{j,l}$ and $j \in \mathcal{N}_2$ if $j \neq i$ and there exists a path $\pi_{j,l} \in \Pi_j(k)$ such that $i \in \pi_{j,l}$. We will prove the desired result for each subset.

By assumption, only s_i is modified. Therefore, a path π is unaffected by an algorithm iteration if $i \notin \pi$. For any $j \in \mathcal{N}_1$, there is a path $\pi_{j,l}$ such that $i \notin \pi_{j,l}$ thus $\overline{\Pi}_j(k+1) \neq \emptyset$ because $\pi_{j,l} \in \overline{\Pi}_j(k+1)$.

Next we show that Steps 2–4 of Algorithm 1 guarantee that if $s_i(k+1) = \hat{s}_i$ is executed at Step 7, $\overline{\Pi}_i(k+1)$ contains at least one path, denoted by π_b. Note that from Step 2, there are three possible control paths which lead to Step 5. When the path is $(2 \to 5)$ or $(2 \to 3 \to 4 \to 5)$, there exists a node $j \in \mathcal{D}_i$ such that $c(\hat{s}_i, s_j) = 1$. Then, by the definition of $\mathcal{D}_i$ and $\mathcal{U}_i$, there is a loop-free path π_a with the form $\pi_a = \{\ldots, i, j, \ldots, 0\}$ and we can set $\pi_b = \gamma_i(\pi_a)$. When the path is $(2 \to 3 \to 5)$, there exists a node j such that $c(\hat{s}_i, s_j) = 1$, $\pi_{j,l} \in \Pi_j(k)$ and $i \notin \pi_{j,l}$. Thus, $\pi_b = \{i\} + \pi_{j,l}$. By now, we have shown the existence of π_b and it will not be affected by later algorithm steps since $\hat{s}_i$ is not modified in them.

Steps 5–6 of Algorithm 1 guarantee that for $j \in \mathcal{N}_2$, $\overline{\Pi}_j(k+1)$ contains at least one path, denoted by π_d. Let $\pi_{j,l}$ be a path in $\Pi_j(k)$ such that $i \in \pi_{j,l}$ and $\pi_c = \beta_i(\pi_{j,l})$. The last element in π_c is denoted by q and note that $q \in \mathcal{U}_i$. From step 5, there are three possible control paths to reach the end of the algorithm. First, when the path is $(5 \to 7)$, we have $c(\hat{s}_i, s_q) = 1$, and $\pi_d = \pi_c + \pi_b$. Second, when the path is $(5 \to 6 \to 7)$, if we have $c(\hat{s}_i, s_q) = 1$, then $\pi_d = \pi_c + \pi_b$. If $c(\hat{s}_i, s_q) = 0$, we have two cases. If there exists a path $\pi_{q,l} \in \Pi_q(k)$, such that $i \notin \pi_{q,l}$, then $\pi_d = \pi_c + \pi_{q,l}$. If there exists a path $\pi_{q,l} \in \Pi_q(k)$, such that $i \in \pi_{q,l}$ and $c(\hat{s}_i, s_{\omega_i(\pi_{q,l})}) = 1$, then $\pi_d = \pi_c + \beta_i(\pi_{q,l}) + \pi_b$. Finally, when the path is $(5 \to 6)$, $s_i(k+1) = s_i(k)$ and no change is made to $\mathbf{s}$. Then, we still have $\overline{\Pi}_j(k+1) = \Pi_j(k) \neq \emptyset$ for all $j \in \{1, \ldots, N\}$.

Now we can conclude that under all possible control paths, $\overline{\Pi}_j(k+1) \neq \emptyset$ for all $j \in \{1, \ldots, N\}$, which implies $\mathcal{G}(\mathbf{s}(k+1))$ is connected. ■

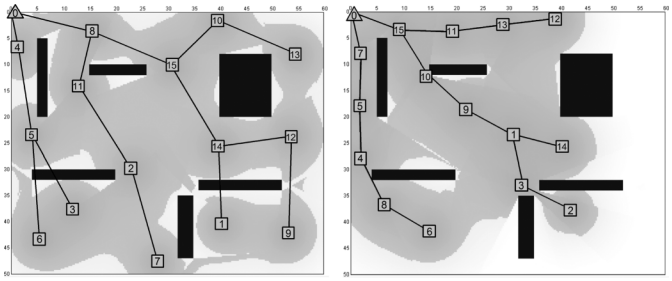

Fig. 9. (a) shows the final node deployment when applying the original coverage control algorithm. The lines between nodes indicate communication links. The objective function reaches a value $H(\mathbf{s}) = 1642.1$. In (b), Algorithm 1 with $C = 10$ is applied to preserve connectivity and $H(\mathbf{s}) = 1449.4$.

Fault Recovery: When two or more nodes update their states at the same time, Algorithm 1 does not guarantee connectivity, thus a fault recovery mechanism is required. Connectivity loss may also be caused by a bad initial condition, excessive communication delays, or node failure. When a node detects that the base station is unreachable, a connection can be re-established by navigating back toward the base station (a known position). As soon as that node establishes a communication link with the base station or any other node with a path to the base station, it can use Algorithm 1 to maintain connectivity.

In Fig. 9, we compare the results of coverage control with and without connectivity preservation. The base station is located at the upper-left corner and all nodes initially start from that location. Blue rectangles represent obstacles and the navigable areas are color-coded to indicate the quality of coverage as described earlier. Fig. 9(a) shows the final node deployment after applying coverage control without connectivity preservation. Fig. 9(b) shows the result of applying Algorithm 1 to the coverage control problem. Here, all communication links are under the desired range limit $C = 10$ and line-of-sight constraints are satisfied (unlike in (a)). As a price for guaranteed connectivity, the final objective function value reached is lower when Algorithm 1 is applied.

III. Data Collection From Detected Data Sources

In our work thus far, we have focused on how to deploy sensors in order to maximize the probability of detecting future events, whose spatial distribution is given by the event density function $R(\cdot)$. Once such an event is detected at a certain location by a sensor node, it becomes a known data source and one can expect that some sensor node locations will be reconfigured to extract additional information from the newly found data source. Inevitably, the reconfiguration will disturb a motion controller (9) aiming at maximizing coverage. However, this should not substantially degrade the coverage component of the mission (e.g., leave security loopholes for future intruders in a surveillance task). Thus, the sensor network's overall objective now becomes to collect information from all known data sources while still maintaining adequate coverage to detect future events. In this section, we discuss how this objective can be achieved by modifying the optimal coverage problem to incorporate a data collection objective.

Similar to a sensing model necessary for formulating the coverage problem, we now need a data collection model. Let Λ_t be a time-varying discrete set which contains all known data source locations in Ω at time t. For each $u \in \Lambda_t$, we define $S(u) \in [0,1]$ to be the "source value" associated with the data source at u. This quantity measures the relative importance of different data sources to the sensor nodes and the exact form of $S(u)$ is application-dependent. In the rest of this section, we will assume that Λ_t and $S(u)$, $u \in \Lambda_t$, are all known. In the next section, we will return to the data source detection process, which ultimately gives us an estimate of Λ_t and we will present a possible form of $S(u)$, which takes sensor perception into consideration.

Next we define $f_i(u, s_i) : \Omega \times \Omega \to [0,1]$ to be a data collection *quality* function for node i. We assume that the quality of data collection deteriorates with distance, thus $f_i(u, s_i)$ should be a non-increasing function of $\|u - s_i\|$. Notice that $f_i(u, s_i)$ could also be affected by factors such as obstacle occlusion and limited sensor field-of-view. The effects of these factors can be modelled by following the same approach as in Section II and will not be elaborated here.

Since multiple nodes can collect data from a single source simultaneously, we also need to define a joint data collection function $F(u, \mathbf{s})$ to represent the total data collection quality at x. Similar to (4) we define

$$F(u, \mathbf{s}) = 1 - \prod_{i=1}^{N} [1 - f_i(u, s_i)]. \tag{22}$$

With Λ_t and $F(u, \mathbf{s})$ defined, we modify the optimization problem (5) to incorporate rewards from collecting data as follows:

$$H(\mathbf{s}, t) = \int_{\Omega} R(x) P(x, \mathbf{s}) \, dx + \beta \sum_{u \in \Lambda_t} S(u) F(u, \mathbf{s}). \tag{23}$$

The positive coefficient β is used to trade off the two competing goals in $H(\mathbf{s}, t)$. For larger β, the sensor network will shift more of its attention from coverage to data collection, and vice versa. Notice that the objective function is time-varying because Λ_t changes over time. The distributed motion control of node i in (9) is now driven by

$$\frac{\partial H(\mathbf{s}, t)}{\partial s_i} = \frac{\partial}{\partial s_i} \int_{\Omega} R(x) P(x, \mathbf{s}) \, dx + \beta \sum_{u \in \Lambda_t} S(u) \frac{\partial F(u, \mathbf{s})}{\partial s_i}. \tag{24}$$

To evaluate (24), the sensor nodes need a mechanism to reliably estimate Λ_t. Such a mechanism is discussed in [25], which utilizes a recursive Bayes filter to integrate successive sensor observations into an occupancy grid map. A simulation example that illustrates the operation of a sensor network under the combined effect of all three tasks of Fig. 1 is shown in Fig. 10. In this example, the data source detection mechanism proposed in [25] is in effect and data collection is traded off against coverage once a mobile data source is detected. In subfigure 1 of Fig. 10, no data source exists and the nodes perform coverage control. In subfigure 2, a data source (purple circle) appears and is detected by the nearest node 0. When the data source moves across the cluttered space, it is always closely monitored by one or two sensor nodes, while at the same time the remaining nodes maintain adequate coverage.

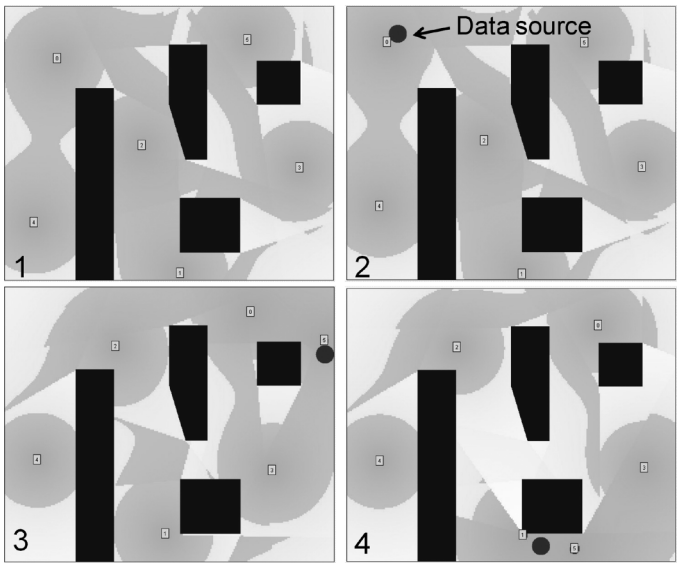

Fig. 10. Data source moves across the mission space and the sensor nodes adjust their locations to trade off between coverage and continuous monitoring of the data source.

IV. Conclusion

We have presented a framework intended to provide an end-to-end solution to a wireless sensor network mission whose objective is to control the locations of mobile nodes so as to maximize the probability of detecting randomly occurring events in a mission space and to extract information from data sources, when detected, with maximal effectiveness. We have identified and developed models for three interacting components required in the execution of such missions: coverage control, data source detection, and data collection. The *coverage control* and *data collection* components are solved jointly as an optimization problem trading off two objectives. The *data source detection* procedure proposed in [25] follows a Bayes estimation approach in order to mitigate the effect of detection errors in the process. We have also discussed the interactions among these three components leading to an overall mission execution control system and addressed related node communication issues.

Future work in coverage control includes refining our model of the sensing cone under limited field of vision, so that the sensing capability varies according to the angular distance from the central axis. In addition, as mentioned in the introduction, the optimization problem we solve needs to incorporate the energy consumed for both communication and motion. In the data detection component of the overall system, we intend to explore tracking approaches for data sources which depend on specific application settings. Finally, we are studying how the event-driven communication mechanism used in [33] to solve the coverage control problem without any synchronization limitation can be used in the the modified optimization problem that combines coverage and data collection quality. A prevailing question in the gradient-based optimization framework is the presence of multiple local optima. As pointed out in Section IIB, modifying the objective function to compensate for imbalanced joint detection probabilities over the mission space can partially address this problem and suggests the possibility for a systematic approach to seek a global optimum or at least improved performance.

References

[1] S. Meguerdichian, F. Koushanfar, M. Potkonjak, and M. B. Srivastava, "Coverage problems in wireless ad-hoc sensor networks," in *Proc. IEEE INFOCOM*, 2001, pp. 1380–1387.
[2] J. Cortes, S. Martinez, T. Karatas, and F. Bullo, "Coverage control for mobile sensing networks," *IEEE Trans. Robot. Autom.*, vol. 20, no. 2, pp. 243–255, Apr. 2004.
[3] L. Mihaylova, T. Lefebvre, H. Bruyninckx, and K. Gadeyne, "Active sensing for robotics-A survey," in *Proc. 5th Int. Conf. Numer. Methods Appl.*, Borovets, Bulgaria, 2002, pp. 316–324.
[4] W. Li and C. G. Cassandras, "Distributed cooperative coverage control of sensor networks," in *Proc. 44th IEEE Conf. Decision Control*, 2005, pp. 2542–2547.
[5] C. G. Cassandras and W. Li, "Sensor networks and cooperative control," *Eur. J. Control*, vol. 11, no. 4-5, pp. 436–463, 2005.
[6] Z. Drezner and H. Hamacher, *Facility Location Application and Theory*. Berlin, Germany: Springer-Verlag, 2002.
[7] Q. Du, V. Faber, and M. Gunzburger, "Centroidal Voronoi tessellations: Applications and algorithms," *SIAM Rev.*, vol. 41, no. 4, pp. 637–676, 1999.
[8] S. Lloyd, "Least squares quantization in PCM," *IEEE Trans. Inform. Theory*, vol. IT-28, no. 2, pp. 129–137, Mar. 1982.
[9] A. Kwok and S. Martínez, "Energy-balancing cooperative strategies for sensor deployment," in *Proc. 46th IEEE Conf. Decision Control*, 2007, pp. 6136–6141.
[10] J. Cortés, S. Martínez, and F. Bullo, "Spatially-distributed coverage optimization and control with limited-range interactions," *ESAIM. Control, Optim. Calculus Var.*, vol. 11, no. 4, pp. 691–719, 2005.
[11] Y. Zou and K. Chakrabarty, "Sensor deployment and target localization based on virtual forces," in *Proc. IEEE INFOCOM*, 2003, pp. 1293–1303.
[12] S. Poduri and G. Sukhatme, "Constrained coverage for mobile sensor networks," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2004, pp. 165–171.
[13] J. O'Rourke, *Art Gallery Theorems and Algorithms*. New York: Oxford Univ. Press, 1987.
[14] A. Ganguli, J. Cortés, and F. Bullo, "Maximizing visibility in nonconvex polygons: Nonsmooth analysis and gradient algorithm design," *SIAM J. Control Optim.*, vol. 45, pp. 1657–1679, 2006.
[15] M. Zhong and C. G. Cassandras, "Distributed coverage control in sensor networks environments with polygonal obstacles," in *Proc. 17th IFAC World Congress*, 2008, pp. 4162–4167.
[16] P. Yang, R. Freeman, G. Gordon, K. Lynch, S. Srinivasa, and R. Sukthankar, "Decentralized estimation and control of graph connectivity in mobile sensor networks," in *Proc. Amer. Control Conf.*, 2008, pp. 2678–2683.
[17] M. Zavlanos and G. Pappas, "Distributed connectivity control of mobile networks," *IEEE Trans. Robotics*, vol. 24, no. 6, pp. 1416–1428, Jun. 2008.
[18] M. Zhong and C. G. Cassandras, "Asynchronous distributed optimization with minimal communication and connectivity preservation," in *Proc. 48th IEEE Conf. Decision Control*, 2009, pp. 5396–5401.
[19] E. Royer and C.-K. Toh, "A review of current routing protocols for ad hoc mobile wireless networks," *IEEE Personal Commun.*, vol. 6, no. 2, pp. 46–55, 1999.
[20] J. Al-Karaki and A. Kamal, "Routing techniques in wireless sensor networks: A survey," *IEEE Wireless Commun.*, vol. 11, no. 6, pp. 6–28, Dec. 2004.
[21] W. Li and C. G. Cassandras, "A cooperative receding horizon controller for multi-vehicle uncertain environments," *IEEE Trans. Autom. Control*, vol. 51, no. 2, pp. 242–257, Feb. 2006.
[22] L. Matthies and A. Elfes, "Integration of sonar and stereo range data using a grid-based representation," in *Proc. IEEE Int. Conf. Robot. Autom.*, 1988, pp. 727–733.
[23] K. Konolige, "Improved occupancy grids for map building," *Auton. Robots*, vol. 4, pp. 351–367, 1997.
[24] M. Tay, K. Mekhnacha, M. Yguel, C. Coue, C. Pradalier, C. Laugier, T. Fraichard, and P. Bessiere, "The bayesian occupation filter," in *Probabilistic Reasoning and Decision Making in Sensory-Motor Systems*, P. Bessiere, C. Laugier, and R. Siegwart, Eds. New York: Springer, 2008.
[25] M. Zhong and C. G. Cassandras, "Distributed coverage control and data collection with mobile sensor networks," in *Proc. 49th IEEE Conf. Decision Control*, 2010, pp. 5604–5609.
[26] T. Asano, S. K. Ghosh, and T. C. Shermer, "Visibility in the plane," in *Handbook of Computational Geometry*, J. R. Sack and J. Urrutia, Eds. Amsterdam, The Netherlands: North-Holland, 2000, pp. 829–876.
[27] D. P. Bertsekas, *Nonlinear Programming*. Belmont, MA: Athena Scientific, 1995.

[28] H. Flanders, "Differentiation under the integral sign," *Amer. Math. Monthly*, vol. 80, no. 6, pp. 615–627, 1973.
[29] S. M. LaValle and J. E. Hinrichsen, "Visibility-based pursuit-evasion: The case of curved environments," *IEEE Trans. Robot. Autom.*, vol. 17, no. 2, pp. 196–206, Apr. 2001.
[30] M. Zhong and C. G. Cassandras, "Converage Control in Environments with Polygonal Obstacles," Boston Univ., Tech. Rep., 2007.
[31] N. Z. Shor, *Minimization Methods for Non-Differentiable Functions*. Berlin, Germany: Springer-Verlag, 1985.
[32] K. C. Kiwiel, *Methods of Descent for Nondifferentiable Optimization*. Berlin, Germany: Springer-Verlag, 1985.
[33] M. Zhong and C. G. Cassandras, "Asynchronous distributed optimization with event-driven communication," *IEEE Trans. Autom. Control*, vol. 55, no. 12, pp. 2735–2750, Dec. 2010.

Minyi Zhong (M'10) received the B.E. degree in electrical and information engineering from Shanghai Jiaotong University, Shanghai, China, in 2005, and the M.S. and Ph.D degree from Boston University, Boston, MA, in 2009 and 2011, respectively.

He is currently working at Microsoft Corporation, Redmond, WA. His research interests include distributed optimization and cooperative control, with applications to communication networks, sensor networks, manufacturing systems, intelligent vehicle systems and robotics.

Christos G. Cassandras (F'96) received the B.S. degree from Yale University, New Haven, CT, in 1977, the M.S.E.E degree from Stanford University, Stanford, CA, in 1978, and the S.M. and Ph.D. degrees from Harvard University, Cambridge, MA, in 1979 and 1982, respectively.

From 1982 to 1984, he was with ITP Boston, Inc. where he worked on the design of automated manufacturing systems. From 1984 to 1996, he was a Faculty Member at the Department of Electrical and Computer Engineering, University of Massachusetts, Amherst. Currently, he is Head of the Division of Systems Engineering and Professor of Electrical and Computer Engineering at Boston University, Boston, MA and a founding member of the Center for Information and Systems Engineering (CISE). He has published over 300 papers in these areas, and five books. He specializes in the areas of discrete event and hybrid systems, stochastic optimization, and computer simulation, with applications to computer networks, sensor networks, manufacturing systems, transportation systems, and command/control systems.

Dr. Cassandras received several awards, including the Distinguished Member Award of the IEEE Control Systems Society (2006), the 1999 Harold Chestnut Prize, and a 1991 Lilly Fellowship. He is a a Fellow of the IFAC and a member of Phi Beta Kappa and Tau Beta Pi. He was Editor-in-Chief of the IEEE TRANSACTIONS ON AUTOMATIC CONTROL from 1998 through 2009 and has served on several editorial boards and as Guest Editor for various journals. He is the 2011 President-Elect of the IEEE Control Systems Society and also serves on the Society's Board of Governors.