Cost-Efficient Barrier Coverage with a Hybrid Sensor Network under Practical Constraints

Xiaoyun Zhang, Mathew L. Wymore, and Daji Qiao
Iowa State University, Ames, IA 50011
{zxydut, mlwymore, daji}@iastate.edu

***Abstract*—Barrier coverage is a natural application of sensor networks in which sensors are deployed to detect intruders or protect crucial resources. In this paper, we consider a hybrid sensor network with a two-phase deployment, in which less-expensive static sensors are first randomly deployed in an area, and then more-expensive mobile sensors are deployed to fill coverage gaps. We use a probabilistic model to take into account the practical constraints of detection probability and false positives. We propose an iterative scheme that finds a sensor deployment strategy that minimizes the total sensor cost. Our scheme makes use of a graph transformation and includes speed-up strategies. We present simulation results that verify the correctness of the proposed scheme and demonstrate the effectiveness of the speed-up strategies.**

I. Introduction

Intruder detection and border surveillance are some of the many applications of sensor networks. In these applications, sensors are deployed in a monitored area to form a virtual barrier, or provide *barrier coverage*. For example, buried seismic sensors can be used to detect the vibrations of intruders traversing a national border [1].

To achieve barrier coverage, sensors can be manually deployed at desired positions, which is labor-intensive and may be dangerous in some scenarios, such as a battlefield. Alternatively, sensors can be dropped from a plane or helicopter, resulting in a random deployment that may have coverage gaps. If the sensors are mobile, they can relocate themselves to the desired positions to form a barrier. However, mobile sensors typically cost more, and an all-mobile sensor barrier would be expensive. A potential compromise is to deploy a hybrid network, with both static and mobile sensors, to achieve barrier coverage.

In our scenario, static sensors are first deployed, potentially leaving coverage gaps. Then, mobile sensors are deployed to fill the gaps and form a barrier. Our goal is to select a subset of static sensors to use in the barrier, and then determine the number of mobile sensors needed to fill any gaps. Our objective is to minimize the total cost of the active sensors, meaning those used to build the barrier. We leave the task of optimizing the actual sensor movement to other work, e.g. [2]–[4].

We consider *strong barrier coverage*, in which intruders may take any path to cross the monitored region. Unlike previous work such as [5], we employ a probabilistic model that allows us to consider the practical constraints of system detection probability and false alarm probability in our solution. However, this model also presents a challenge: the number of active sensors affects the decision threshold required to meet the false alarm probability constraint, which in turns affects the density of sensors required to meet the detection probability constraint.

Summarizing, this paper provides the following contributions:

- We define the min-cost strong barrier problem under a probabilistic model, and transform the barrier construction problem with probabilistic constraints to a graph problem.
- We propose an efficient iterative algorithm to solve the problem, including speed-up strategies that skip some iterations and prune the graph in each iteration.

The paper is organized as follows: Section II presents the probabilistic model and the problem statement. Section III describes the proposed scheme. Section IV presents results from our evaluation of the proposed scheme. Section V discusses related work. Finally, Section VI concludes the paper.

II. Model and Problem Statement

A. System Model

We consider a hybrid network of static and mobile sensors deployed to monitor a rectangular region with length L and width W, with the goal of detecting any intruders traversing the width of the region. We consider *strong barrier coverage*, meaning that an intruder, or *target*, may take any path to traverse the width of the region.

B. Sensing Model

We use a probabilistic sensing model, in which sensor readings are affected by randomly varying noise and sensor nodes use a decision threshold to determine if an intruder is present or not. Our model consists of a source model, a detection model, and a false alarm model.

1) Source Model: We assume either the target or its motion produces a physical signal, such as sound, electromagnetic waves, or vibrations. We assume the strength of the signal decays according to the power law, meaning that if the target is at point t, the signal strength at the location of sensor s_i is [6], [7]:

$$\omega_i(t) = \frac{\Omega}{1 + \big(d\,(s_i, t)\,\big)^{\alpha}}, \tag{1}$$

where Ω is the signal amplitude at the target, α is a known decay exponent, and $d(\cdot, \cdot)$ is the distance between two points.

2) Detection Model: In our detection model, we assume that background noise affects sensor readings. When a target is present at point t, a sensor s_i observes a signal $\mathbf{x}_i$ that depends on (1) and the background noise n, as follows:

$$\mathbf{x}_i = \omega_i(t) + n. \tag{2}$$

978-1-4673-8999-0/17/$31.00 ©2017 IEEE

When no target is present, $\mathbf{x}_i = n$. Let $F_N(n)$ denote the cumulative distribution function of noise, and assume that it is identical and independent for all sensors. We also assume that F_N is known by the base station.

To detect a target, sensors set a decision threshold T. When a sensed reading $\mathbf{x}_i$ exceeds T, the sensor generates an alarm to report the presence of a target. Therefore, given T, the probability that sensor s_i detects a target at point t is:

$$P_d(s_i, t) = 1 - F_N\big(T - \omega_i(t)\big). \tag{3}$$

Given a traversing path φ, we define $P_D(\varphi)$ as the maximum probability of detection, by any active sensor, for any point along the path. In other words, $P_D(\varphi)$ is the detection probability of the most well-monitored point in path φ. If we use S_A to denote the set of active sensors, then:

$$P_D(\varphi) = \max_{t\in\varphi} \max_{s_i\in S_A} P_d(s_i, t). \tag{4}$$

Strong barrier coverage assumes that the target may take any traversing path φ. Thus, the worst-case probability of detecting any given intruder is $P_D(\varphi)$ of the least-monitored φ. We call this worst-case probability the *system detection probability*, P_D, and define it as follows:

$$P_D = \min_{\varphi} P_D(\varphi) = \min_{\varphi} \max_{t\in\varphi} \max_{s_i\in S_A} P_d(s_i, t). \tag{5}$$

3) False Alarm Model: Due to excessive noise, a sensor may generate an alarm and report the presence of a target when no target is present. This type of alarm is called a *false alarm*. The probability of false alarms should be bounded in order to avoid burdening the end user. The probability of a particular sensor generating a false alarm is:

$$P_f = 1 - F_N(T). \tag{6}$$

We then define the *system false alarm probability* P_F as the probability that any sensor produces a false alarm, as follows:

$$P_F = 1 - (1 - P_f)^{|S_A|}, \tag{7}$$

where $|S_A|$ is the total number of active sensors. This definition of P_F is consistent with system false alarm probability in [6] and [8] and network false alarm rate in [7].

C. Problem Statement

We define *strong ($P_D^{\min}$, $P_F^{\max}$)-barrier coverage* as a barrier coverage that requires $P_D \geq P_D^{\min}$ and $P_F \leq P_F^{\max}$ for any traversing path through the region. In this paper, we consider a two-phase deployment strategy to achieve this coverage. First, N_s^{total} static sensors are randomly deployed in the monitored region, potentially leaving some coverage gaps. Then, mobile sensors are deployed to fill the coverage gaps between static sensors, ultimately forming a barrier. Mobile sensors usually are more expensive than static sensors, and we use ν to represent the mobile-to-static sensor cost ratio and we assume $\nu \geq 1$.

The goal is to minimize the overall cost of sensors used to achieve strong ($P_D^{\min}$, $P_F^{\max}$)-barrier coverage. Any static sensors not chosen to be active in the barrier are left in the monitored region, and they can participate future barrier constructions and are therefore not part of the barrier cost. Let N_s be the number of static sensors selected to be active, and N_m be the number of mobile sensors needed to fill the coverage gaps between active static sensors. Formally, our problem is to minimize $(\nu N_m + N_s)$, subject to $P_D \geq P_D^{\min}$, $P_F \leq P_F^{\max}$, and $N_s \leq N_s^{\text{total}}$.

III. Proposed Scheme

A. Overview

To solve our min-cost barrier coverage problem, we propose a scheme that iterates over the assumed number of active sensors, N_A. The basic idea is to first assume a value for N_A, which is used to set the decision threshold T and calculate the sensing radius R_s. Then, we check whether the minimum cost for strong ($P_D^{\min}$, $P_F^{\max}$)-barrier coverage can be achieved with N_A sensors. If not, we update our assumption for N_A and iterate.

The scheme is composed of four modules, shown in Fig. 1. The initialization module determines the N_A value for the iteration and accelerates the scheme by skipping N_A values that will not produce a valid solution, meaning that $P_D^{\min}$ and $P_F^{\max}$ will not be satisfied. The initialization module also outputs the sensing radius corresponding to the value of N_A. The mapping module transforms the original problem into a graph problem by generating, updating, and pruning a weighted graph G, which is used in the next two modules. The min-cost algorithm finds the cost lower bound C_l by identifying the min-cost set of sensors, denoted by S_c. The hop-restricted algorithm finds the cost upper bound C_u by identifying the best strategy with exactly N_A sensors, denoted by S_h. S_h^* represents the best feasible solution if the current N_A assumption is correct, and it is stored as a potential output. The number of mobile sensors in S_h^* is used to update N_m^u, the upper bound on the number of mobile sensors for future iterations. N_m^u is provided to the mapping module in order to prune G in each iteration. If the upper and lower bounds on cost meet (i.e., $C_l \geq C_u$), the scheme terminates and outputs a set of active sensors S_{final} as the final optimal deployment strategy. Otherwise, N_A is updated and iterations continue. Next, we introduce each module in detail.

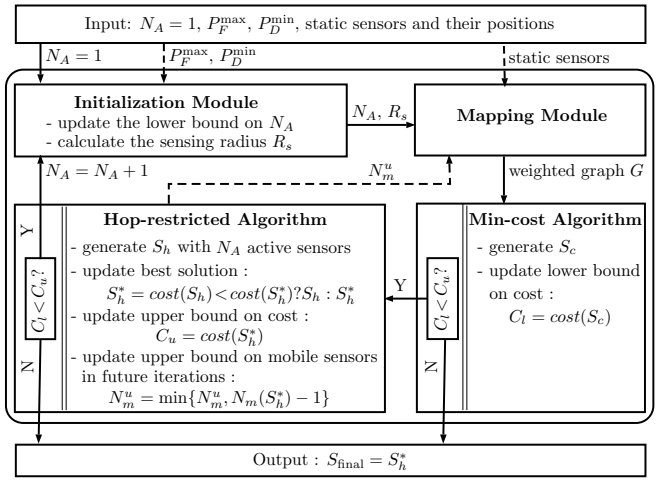


Fig. 1. Overview of proposed scheme. The solid lines indicate iteration flow and the dashed lines indicate parameter flow.

B. Initialization Module

The initialization module, shown in Fig. 2, initializes each iteration. It determines the assumption of N_A to use for the iteration. The initialization module accelerates the scheme by skipping N_A values which will not produce a valid solution. The module takes a tentative N_A as input, and outputs the next value of N_A that has a valid solution.

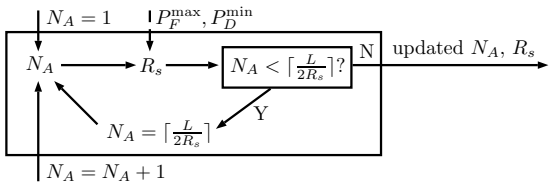


Fig. 2. Initialization module.

As shown in Fig. 2, when entering this module, N_A is set to one for the first iteration, and $N_A + 1$ in subsequent iterations. Given N_A, $P_D^{\min}$, $P_F^{\max}$ and other sensing model-related parameters, this module performs the following steps.

1) Calculate the sensing threshold T. A sensed reading greater than T shall trigger an alarm. From (6) and (7), to satisfy $P_F \leq P_F^{\max}$, T is calculated as:
$$T \leq F_N^{-1}\left((1 - P_F^{\max})^{1/N_A}\right). \quad (8)$$
To maximize the coverage region, we choose the maximum value for T.
2) Calculate the sensing radius R_s. Intruders within this radius of a sensor shall be detected by that sensor with probability $P_D^{\min}$. Combining (1) and (3), we obtain:
$$R_s = \left(\frac{\Omega}{T - F_N^{-1}(1 - P_D^{\min})} - 1\right)^{1/\alpha}. \quad (9)$$
3) Compute the minimum number of sensors required to achieve barrier coverage, $\lceil \frac{L}{2R_s} \rceil$.
4) If $\lceil \frac{L}{2R_s} \rceil$ is larger than N_A, then $N_A = \lceil \frac{L}{2R_s} \rceil$ and repeat the above steps; otherwise, the module outputs the current N_A.

The first three steps may need to be repeated because the value of N_A affects T, which affects R_s. When $\lceil \frac{L}{2R_s} \rceil > N_A$, the potential values of N_A between N_A and $\lceil \frac{L}{2R_s} \rceil$ are skipped because they will not lead to valid solutions, as proved in Theorem 1.

Theorem 1. *If $\lceil \frac{L}{2R_s} \rceil > N_A$, then for any $N'_A \in [N_A, \lceil \frac{L}{2R_s} \rceil)$, all deployment strategies with the assumption of N'_A active sensors would yield a P_F larger than $P_F^{\max}$, making them invalid solutions.*

Proof: Since $N'_A \geq N_A$, we have $R'_s \leq R_s$, because R_s is a decreasing function of N_A according to (9). This leads to $\lceil \frac{L}{2R'_s} \rceil \geq \lceil \frac{L}{2R_s} \rceil$. Let T' denote the sensor decision threshold and S'_A denote the set of active sensors in any deployment strategy corresponding to N'_A. Since $|S'_A| \geq \lceil \frac{L}{2R'_s} \rceil$, we have

$$\begin{aligned} P_F &= 1 - F_N(T')^{|S'_A|} \geq 1 - F_N(T')^{\lceil \frac{L}{2R'_s} \rceil} \\ &\geq 1 - F_N(T')^{\lceil \frac{L}{2R_s} \rceil} > 1 - F_N(T')^{N'_A} = P_F^{\max}. \end{aligned}$$ ∎

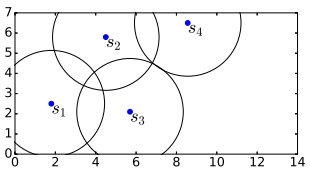
As an example of how the initialization module works, suppose we want to build a barrier in an area of length $L = 14$ m. The signal amplitude emitted by the target, Ω, is 30 mW and the standard deviation of the environmental noise is $\sigma = 1$ mW. Given $P_F^{\max} = 0.05$, $P_D^{\min} = 0.95$, and $N_A = 1$, we obtain $R_s = 2.85$ m in Step 2 and $\lceil \frac{L}{2R_s} \rceil = 3$ in Step 3. Since $\lceil \frac{L}{2R_s} \rceil > N_A$, we return to Step 1 with $N_A = 3$. In the following steps, we obtain $R_s = 2.64$ m and $\lceil \frac{L}{2R_s} \rceil = 3$. Now $\lceil \frac{L}{2R_s} \rceil = N_A$, so we output $N_A = 3$ and $R_s = 2.64$ m.

C. Mapping Module

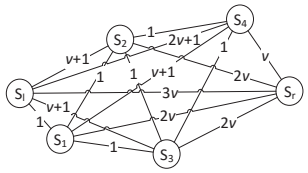
The mapping module maps the sensor network to an undirected weighted graph G to allow our scheme to use graph-based algorithms for optimization. G is initialized during the first iteration, and updated and pruned during the following iterations. Initially, G is constructed as follows.

- Vertices: There is a vertex for each static sensor. Two additional vertices, s_l and s_r, represent the left and right boundaries of the region, respectively.
- Edges: There is an edge between any two vertices.
- Weight: The weight of an edge is the cost of mobile sensors required to fill the gap between them, plus the cost of the static sensor at one end of the edge. In detail, the weights are listed below. R_s is the coverage radius corresponding to the N_A assumed for this iteration.
  - Between two physical, static sensors s_i and s_j:
  $$w_{i,j} = \lceil \frac{\max\{\text{dist}(s_i, s_j) - 2R_s, 0\}}{2R_s} \rceil * \nu + 1. \quad (10)$$
  - Between a physical sensor s_i and a boundary:
  $$w_{s_l,i} = \lceil \frac{\max\{x_i - R_s, 0\}}{2R_s} \rceil * \nu + 1. \quad (11)$$
  $$w_{i,s_r} = \lceil \frac{\max\{L - x_i - R_s, 0\}}{2R_s} \rceil * \nu. \quad (12)$$
  - Between the left and right boundaries:
  $$w_{s_l,s_r} = \lceil \frac{L}{2R_s} \rceil * \nu. \quad (13)$$

Continuing with the example from the previous section, let the input to the mapping module be $N_A = 3$ and $R_s = 2.64$ m, with the static sensors shown in Fig. 3(a). The graph G is then constructed as shown in Fig. 3(b).



(a) Sensor deployment. The circles are the coverage regions of the static sensors with $N_A = 3$.



(b) Weighted graph G. s_l and s_r represent the left and right boundaries. Edges are labeled with weights.

Fig. 3. The graph initially constructed by the mapping module.

We have two observations regarding G. First, any path from s_l to s_r in G represents a possible deployment strategy, and

the sum of the weights of the edges in the path is the cost of the corresponding strategy. However, if this corresponding strategy requires more than N_A active sensors, then it is not necessarily valid with respect to $P_F^{\max}$ and $P_D^{\min}$.

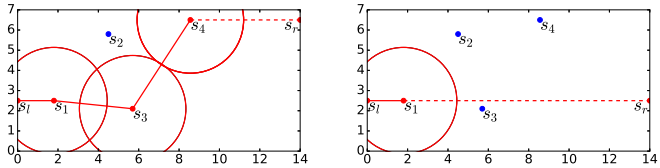
For instance, in Fig. 3(b), $\{s_l, s_1, s_2, s_3, s_4, s_r\}$ is a possible deployment strategy that uses one mobile sensor between s_4 and s_r. But this strategy uses five sensors in total, which violates the assumption of $N_A = 3$, so this strategy may not be a valid solution. As another example, $\{s_l, s_1, s_r\}$ is a valid strategy with three sensors and a cost of $1 + 2\nu$; it uses two mobile sensors between s_1 and s_r.

The second observation is that G initially is constructed as a fully connected graph. This is necessary because, in the optimal solution, mobile sensors may be deployed to fill the gap between *any* pair of static sensors, and not necessarily the smallest gap between two static sensor clusters. Although a barrier built by only filling the shortest gaps between static sensor clusters may use fewer mobile sensors, it also may use many more static sensors, possibly at a higher cost. To increase efficiency, our scheme prunes edges from the graph in future iterations. This pruning process will be discussed in Section III-F.

D. Min-cost Algorithm to Update Cost Lower Bound

The goal of the min-cost algorithm is to find the solution that minimizes the active sensor cost. We achieve this by finding a minimum cost path from s_l to s_r on G using Dijkstra's algorithm. The output of this algorithm is a set of active sensor nodes, S_c, that correspond to the vertices in the path, plus any mobile sensors that need to be added.

As an example, Fig. 4(a) shows the min-cost path S_c from the example in Fig. 3(b) when $\nu = 3$. This path is $\{s_l, s_1, s_3, s_4, s_r\}$. Three static sensors, s_1, s_3, and s_4, are used. One mobile sensor is required to fill the gap between s_4 and the right boundary s_r. The total cost of S_c is $cost(S_c) = 6$.



(a) Output of the min-cost algorithm when $N_A = 3$ and $\nu = 3$.

(b) Output of the hop-restricted algorithm when $N_A = 3$ and $\nu = 3$.

Fig. 4. Example outputs of the graph algorithms. An edge between two sensors means that both sensors are active. A solid edge means their coverage regions overlap, requiring no mobile sensors. A dashed edge means mobile sensors are needed to fill the coverage gap between the sensors. In (a), one mobile sensor is required between s_4 and s_r, and in (b), two are required between s_1 and s_r.

S_c is the min-cost solution on G, but if $|S_c| > N_A$, the $P_F^{\max}$ requirement may not be satisfied by this solution. Therefore, S_c may be an invalid solution. However, even in this case, $cost(S_c)$ can still serve as a valid lower bound on the cost for future iterations. This is because, in future iterations, (1) the assumed number of active sensors increases, the sensor coverage radius decreases, and the edge weights on G increase; (2) the edge set of G is smaller because it is pruned in each iteration. Thus, any solution found in future versions of G would yield a higher cost than that of the current S_c.

Accordingly, the lower cost bound C_l is updated to $cost(S_c)$. The updated C_l is then compared to the cost upper bound, C_u. If $C_l < C_u$, a better solution may be found in future iterations, so the scheme continues to the hop-restricted algorithm. Otherwise, the scheme terminates and outputs the best valid solution yet found (from the hop-restricted algorithm in a previous iteration). C_u is initialized to infinity, so in the example in Fig. 4(a), $C_l = cost(S_c) = 6 < C_u = \infty$ and we continue to the hop-restricted algorithm.

E. Hop-restricted Algorithm to Update Cost Upper Bound

The hop-restricted algorithm identifies the best path S_h from s_l to s_r that has exactly N_A physical sensors. This restriction ensures that S_h is a valid solution, distinguishing the hop-restricted algorithm from the min-cost algorithm in the previous module, which does not restrict hops and thus may not find a valid solution. The hop-restricted algorithm tracks the best valid solution found so far, and updates the cost upper bound C_u, terminating the scheme if the termination criterion is met. This algorithm also outputs the upper bound on the number of mobile sensors, N_m^u, which is used for graph pruning in the next iteration.

1) Hop-restricted Path S_h and Cost Upper Bound C_u: S_h is obtained by running a dynamic programming-based algorithm on G. For the algorithm, two sets of weights are needed:

- $w_{i,j}$: the cost of active sensors to fill the gap between s_i and s_j, as defined in Section III-C, and
- $w_{i,j}^t$: the number of active sensors to fill the gap between s_i and s_j, obtained by setting the cost ratio ν in $w_{i,j}$ to one.

Let c_i^k denote the minimum cost of the path from s_l to s_i with k physical sensors. For $k \geq 1$,

$$c_i^k = \min_{j \in \Gamma_i} \{c_j^{k - w_{i,j}^t} + w_{i,j}\}, \quad \text{for } i = [1, 2, ..., s_r], \tag{14}$$

where Γ_i is s_i's neighborhood set. For $k \leq 0$, we define:

$$c_{s_l}^0 = 0, \tag{15}$$

$$c_i^0 = \infty, \quad \text{for } s_i \neq s_l, \tag{16}$$

$$c_i^k = \infty, \quad \text{for any } i \text{ if } k < 0. \tag{17}$$

The hop-restricted algorithm iterates over k and terminates when $c_{s_r}^{N_A}$ is obtained. S_h is then the path that reaches s_r with cost $c_{s_r}^{N_A}$. We define S_h^* as the best solution found so far. If $cost(S_h)$ is less than $cost(S_h^*)$, we store S_h as S_h^* and update the cost upper bound C_u to $cost(S_h^*)$.

Fig. 4(b) shows the hop-restricted path S_h for the example in Fig. 3(b). With $N_A = 3$ and $\nu = 3$, S_h is $\{s_l, s_1, s_r\}$. One static sensor, s_1, is used. Two mobile sensors are needed to fill the gap between s_1 and the right boundary. The cost of S_h is 7, which is less than $cost(S_h^*) = \infty$ (as no S_h^* has previously been stored). S_h is then stored as S_h^*, and C_u is updated to 7.

Once C_u is updated, it is compared with C_l. If $C_l < C_u$, the scheme continues to the next iteration, with $N_A = N_A + 1$ as the input to the initialization module. Otherwise, the optimal solution has been found, and the scheme terminates and

outputs S_h^*. In the above example, after executing the hop-restricted algorithm with $N_A = 3$, $C_l = 6 < C_u = 7$. Therefore, we continue with the next iteration and pass $N_A = 4$ to the initialization module.

2) Upper Bound on the Number of Mobile Sensors N_m^u: Another output of the hop-restricted algorithm is N_m^u, the upper bound of the number of mobile sensors for any solution in future iterations that has a lower cost. N_m^u is set as follows:

$$N_m^u = \min\{N_m^u, N_m^* - 1\}, \tag{18}$$

where N_m^* is the number of mobile sensors in S_h^*, shown as $N_m(S_h^*)$ in Fig. 1. For a solution S_h in any future iteration, we have

THEOREM 2. *$cost(S_h) > cost(S_h^*)$ if $N_m \geq N_m^*$.*

Proof: Since N_A increases as the scheme iterates, we have $|S_h| > |S_h^*|$, where $|S_h|$ and $|S_h^*|$ are the number of active sensors on S_h and S_h^*, respectively. Then, we have

$$N_m + N_s > N_m^* + N_s^* \Longrightarrow N_s > N_m^* + N_s^* - N_m.$$

Further,

$$\begin{aligned} cost(S_h) = \nu N_m + N_s &> \nu N_m + N_m^* + N_s^* - N_m \\ &= (\nu - 1)N_m + N_m^* + N_s^* \\ &\geq (\nu - 1)N_m^* + N_m^* + N_s^* \\ &= \nu N_m^* + N_s^* = cost(S_h^*). \end{aligned}$$ ■

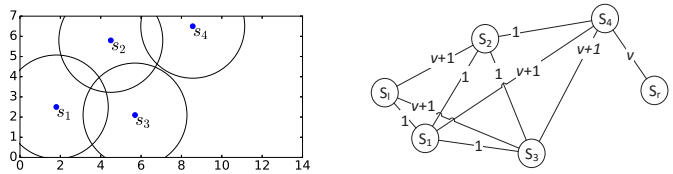
Theorem 2 shows that a solution S_h in future iterations with N_m^* or more sensors will have a higher cost than the current best solution S_h^*. Therefore, $N_m^* - 1$ is the upper bound of the number of mobile sensors in any solution in future iterations that has a lower cost.

F. Mapping Module Revisited

In the mapping module, after the first iteration, G only needs to be updated and pruned. G's vertex set remains the same for all iterations, while G's edge weights are updated and the edge set is pruned, using the following procedure:

1) Update: With the updated N_A and R_s in a new iteration, the number of mobile sensors needed to fill the gaps between static sensors is recalculated, and the edge weights of G are updated correspondingly.
2) Prune: After updating the weights of G, the edges of G that need more than N_m^u mobile sensors to fill the coverage gap are pruned.

Returning to the example, the solution shown in Fig. 4(b) was stored as S_h^* and N_m^u was updated to $N_m^u = N_m^* - 1 = 1$. Fig. 5 shows the sensor deployment and graph of this example in the following iteration, with $N_A = 4$. Comparing with Fig. 3, we can see that the weights of some edges have been updated. For example, a coverage gap has appeared between s_3 and s_4 due to the decreased R_s, and the weight of the edge has increased by ν to reflect that a mobile sensor is now required. Additionally, several edges have been pruned, such as the edge from s_3 to s_r.



(a) Sensor deployment. R_s is calculated with $N_A = 4$.

(b) Pruned weighted graph with $N_m^u = 1$.

Fig. 5. The graph updated and pruned by the mapping module.

G. Terminating Condition

The scheme terminates when the upper and lower bounds for the cost meet or cross, meaning $C_l \geq C_u$. This can occur in either the min-cost algorithm or the hop-restricted algorithm. In our example, the min-cost algorithm is run on the newly pruned G in Fig. 5(b). The min-cost path in G is $\{s_l, s_1, s_2, s_4, s_r\}$, shown in Fig. 6. Its cost is 6, so C_l is updated to 6. Since C_l is still less than C_u, the hop-restricted algorithm is run on G, producing the same path. This path is saved as S_h^*. The cost upper bound C_u is then updated to 6 and now equals C_l, so the scheme terminates and outputs $S_{\text{final}} = S_h^* = \{s_l, s_1, s_2, s_4, s_r\}$. This solution consists of three static sensors (s_1, s_2, and s_4) and one mobile sensor to fill the gap between s_4 and the right boundary.

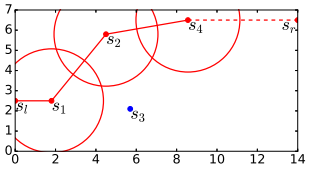


Fig. 6. Output of the min-cost algorithm and hop-restricted algorithm when $N_A = 4$ and $\nu = 3$. The output consists of three static sensors (s_1, s_2, and s_4) and one mobile sensor to fill the gap between s_4 and the right boundary.

H. Complexity Analysis

Now let us do a complexity analysis. The total number of iterations will not be more than $|S_{\text{final}}|$, since we skip some N_A values in the initialization module. In an iteration with the assumed number of sensors as N_A, the min-cost algorithm has a worst-case complexity of $O((N_s^{\text{total}})^2)$, the hop-restricted algorithm has a worst-case complexity of $O(N_A|E|) = O(N_A(N_s^{\text{total}})^2)$ where $|E|$ is the number of edges on the weighted graph G. Sum up all the iterations, the worst-case complexity should be $O(|S_{\text{final}}|^2(N_s^{\text{total}})^2)$. In practice, the scheme performs far more better than the worst-case due to the skipping of N_A and the graph pruning strategy which is verified by simulation.

IV. EVALUATION

We evaluate the performance of the proposed scheme by varying the mobile-to-static sensor cost ratio (ν), the total number of deployed static sensors (N_s^{total}), and the $P_F^{\max}$ and $P_D^{\min}$ parameters. We also demonstrate the effectiveness of N_A skipping and the graph pruning strategy. In our simulations, sensors are randomly deployed in a 100×10 m rectangular region. The default simulation parameters are shown in Table I, which are chosen according to the real-world data mentioned in [6]. All the simulation results are an average of 50 experiments.

TABLE I
DEFAULT SIMULATION PARAMETERS

Parameter	Meaning	Default Value
$P_D^{\min}$	Minimum system detection probability	0.95
$P_F^{\max}$	Maximum system false alarm probability	0.05
Ω	Source signal strength	30 mW
α	Source signal decay exponent	2
F_N	CDF of noise distribution	Gaussian
μ	Noise mean	0 mW
σ	Noise standard deviation	1 mW
ν	Mobile-to-static sensor cost ratio	5
N_s^{total}	Total number of deployed static sensors	100

A. Effect of Cost Ratio

Fig. 7 demonstrates the effect of cost ratio on the number of active mobile and static sensors. When the cost ratio is one, meaning mobile sensors and static sensors have the same cost, the scheme only uses mobile sensors. This is because, in this case, a barrier between the left and right boundaries can be formed at minimum cost with a horizontal line of mobile sensors. At higher cost ratios, meaning more expensive mobile sensors, the scheme favors static sensors over mobile sensors. Additionally, as the cost ratio increases, the total number of active sensors also increases. This is because the scheme must seek out solutions that are more indirect and winding, requiring more static sensors, in order to reduce coverage gaps and the number of mobile sensors.

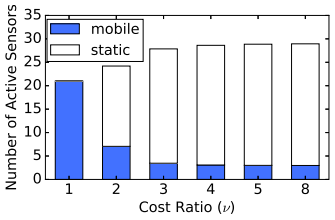
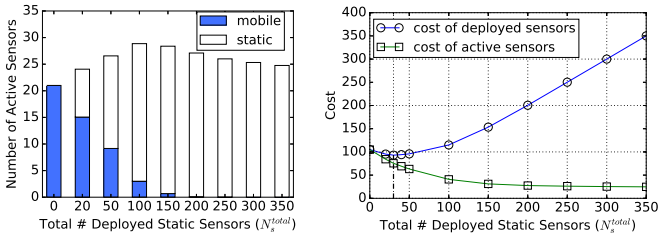


Fig. 7. Effect of cost ratio.

B. Effect of the Total Number of Deployed Static Sensors

Fig. 8(a) shows the effect of the total number of deployed static sensors on the number of active mobile and static sensors. As more static sensors are deployed, fewer mobile sensors are utilized, because the static sensor deployment is better able to form a barrier at less cost on its own. When the number of deployed static sensors reaches a threshold (in this case 200), mobile sensors are no longer needed. Above this threshold, deploying more static sensors slowly reduces the total number of active sensors required. This is because more static sensors leads to more possible strategies that satisfy the barrier coverage requirements, and some of these additional strategies may use fewer active static sensors.



(a) Number of active sensors vs. total number of deployed static sensors. (b) Sensor cost vs. total number of deployed static sensors.

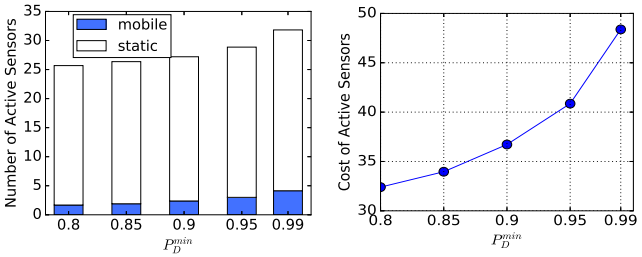
Fig. 8. Effects of the total number of deployed static sensors.

Fig. 8(b) shows sensor costs as the total number of deployed static sensors increases. Two costs are evaluated. "Cost of active sensors" is the total cost of the active sensors, both mobile and static. "Cost of deployed sensors" is the total cost of all deployed sensors, both active and inactive. Note that the mobile sensors are deployed as needed and hence all of them are active.

In Fig. 8(b), the cost of deployed sensors first decreases and then increases. In the scenario demonstrated in Fig. 8(b), the minimum cost is reached on average when deploying 30 static sensors. When the number of deployed static sensors is below 30, more mobile sensors must be used, increasing the cost. When the number of deployed static sensors is higher than 30, the increased cost of deployed static sensors outweighs the decreased number of mobile sensors. Overall, we observe that the total number of deployed static sensors should be carefully chosen to minimize the total cost. On the other hand, the cost of active sensors, which is minimized by the proposed scheme, strictly decreases with the number of deployed static sensors. This aligns with the results in Fig. 8(a).

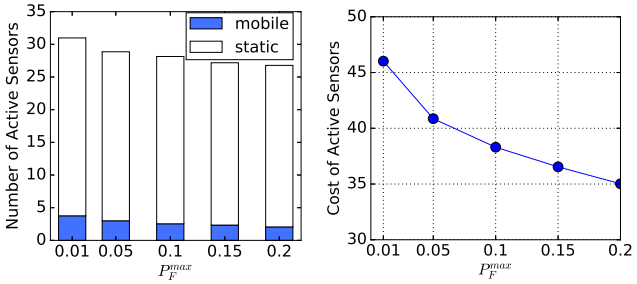
C. Effects of $P_D^{\min}$ and $P_F^{\max}$

Fig. 9 shows the number of active sensors and their costs with different values of $P_D^{\min}$. The higher $P_D^{\min}$ is, the more sensors are needed to reach the required coverage level. The cost increases correspondingly.



(a) # active sensors vs. $P_D^{\min}$. (b) Cost of active sensors vs. $P_D^{\min}$.

Fig. 9. Effect of $P_D^{\min}$.



(a) # active sensors vs. $P_F^{\max}$. (b) Cost of active sensors vs. $P_F^{\max}$.

Fig. 10. Effect of $P_F^{\max}$.

Fig. 10 shows the number of active sensors and their costs with different values of $P_F^{\max}$. The higher $P_F^{\max}$ is, the fewer mobile and static sensors are needed. This is intuitive, as increasing $P_F^{\max}$ relaxes the constraint.

D. Effectiveness of N_A Skipping and Graph Pruning

Table II shows the number of iterations of different setups, compared with the number of sensors in the final solution $|S_{\text{final}}|$. As we can see, the number of iterations is less than $|S_{\text{final}}|$ since we skip some N_A values in the initialization

module. The simulation results demonstrate the effectiveness of the skipping strategy.

TABLE II
NUMBER OF ITERATIONS WITH DIFFERENT L AND N_s^{TOTAL} ($W = 10$ M)

	$L = 100$ m			$L = 250$ m			$L = 500$ m		
N_s^{total}	50	100	200	50	100	200	50	100	200
$\lvert S_{\text{final}} \rvert$	26.6	28.9	27.1	61.3	66.2	73.7	118.3	124.7	135
# Iterations	8.7	10.9	9.1	10.4	15.3	22.8	10.3	16.9	27.1

Fig. 11 shows the number of edges in G throughout the iterations of the scheme, a measure of the effectiveness of the pruning process. Results are shown for 50 and 200 deployed static sensors. The number of edges is normalized to the number of edges in the initial, fully-connected graph. As the scheme iterates, the edges in G are gradually pruned. When the number of deployed static sensors is higher, more edges are pruned because fewer mobile sensors are used in the solutions, providing a tighter upper bound on the number of mobile sensors that can be included in an edge. Overall, the graph pruning process is shown to be effective. This helps expedite the scheme by reducing the computational complexity as the scheme runs.

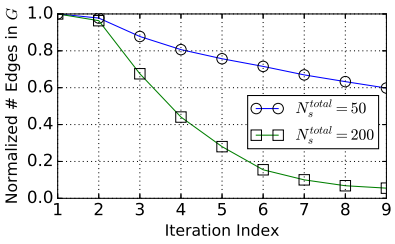


Fig. 11. Number of edges in graph G, normalized to the initial, fully-connected graph, vs. iteration.

V. RELATED WORK

Early works on barrier coverage assumed a simple coverage model, but later works have proposed probabilistic coverage models to better capture sensing behavior. In [6] and [9], the authors analyzed the coverage region of sensors and studied the area coverage problem under a probabilistic model. Yang [8] proposed a scheme to achieve weak barrier coverage with a constraint on system false alarm probability under a probabilistic model. All of these papers agree that the coverage region under the probabilistic model must consider the constraint on false alarm probability. This brings new challenges to coverage schemes that utilize probabilistic models.

Hybrid networks have been used in many sensor network applications. In [10] and [11], the authors studied the tradeoff of deploying static and mobile sensors, and proposed schemes to achieve area coverage with a hybrid network. For barrier coverage, Wang [5] considered how to achieve k-barrier coverage with the minimum number of mobile directional sensors filling the gaps between static ones. This problem is similar to the problem studied in this paper, but instead of k-barrier coverage under the disk model, we consider strong barrier coverage under a probabilistic model with constraints on system detection probability and false alarm probability. Additionally, we consider a more general sensor cost instead of minimizing only the number of mobile sensors. In [12], Kim studied the problem of achieving max-lifetime barrier coverage by scheduling mobile sensors to move around barriers composed of static sensors, focusing on the scheduling problem with a fixed number of mobile sensors. Xu [13] investigated the problem of allocating mobile sensors to fortify weak points in a barrier, which intruders may have a higher probability of visiting. The objective was to provide a minimum level of coverage, instead of minimizing the cost.

Many papers focus on how to achieve barrier coverage with only mobile sensors [2], [3], [14], [15], instead of a hybrid network. These papers propose schemes to relocate mobile sensors to construct a barrier with various objectives, such as minimizing the maximum moving distance of sensors or minimizing the sum of moving distances of sensors. Our work complements these works; we focus on determining where to form the barrier, while these works generally assume the barrier location is known and focus on how to move mobile sensors to the barrier.

VI. CONCLUSION

In this paper, we proposed a scheme to solve the min-cost barrier coverage problem in a hybrid sensor network with static and mobile sensors. Our scheme takes into account the practical constraints of system detection probability and false alarm probability. Simulation results show the proposed scheme can effectively choose the deployment strategy that achieves strong barrier coverage at a minimum cost.

REFERENCES

[1] Z. Sun, P. Wang, M. C. Vuran, M. A. Al-Rodhaan, A. M. Al-Dhelaan, and I. F. Akyildiz, "Bordersense: Border patrol through advanced wireless sensor networks," *Ad Hoc Networks*, vol. 9, no. 3, pp. 468–477, 2011.

[2] A. Saipulla, B. Liu, G. Xing, X. Fu, and J. Wang, "Barrier coverage with sensors of limited mobility," in *Proc. ACM MobiHoc*, 2010.

[3] S. Li and H. Shen, "Minimizing the maximum sensor movement for barrier coverage in the plane," in *Proc. IEEE INFOCOM*, 2015.

[4] X. Zhang, M. L. Wymore, and D. Qiao, "Optimized barrier location for barrier coverage in mobile sensor networks," in *Proc. IEEE WCNC*, 2015.

[5] Z. Wang, J. Liao, Q. Cao, H. Qi, and Z. Wang, "Achieving k-barrier coverage in hybrid directional sensor networks," *IEEE Transactions on Mobile Computing*, vol. 13, no. 7, pp. 1443–1455, 2014.

[6] G. Xing, R. Tan, B. Liu, J. Wang, X. Jia, and C.-W. Yi, "Data fusion improves the coverage of wireless sensor networks," in *Proc. ACM MobiCom*, 2009.

[7] R. Tan, G. Xing, J. Wang, and B. Liu, "Performance analysis of real-time detection in fusion-based sensor networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, no. 9, 2011.

[8] G. Yang and D. Qiao, "Barrier information coverage with wireless sensors," in *Proc. IEEE INFOCOM*, 2009.

[9] W. Wang, V. Srinivasan, K.-C. Chua, and B. Wang, "Energy-efficient coverage for target detection in wireless sensor networks," in *Proc. IEEE IPSN*, 2007.

[10] W. W. V. Srinivasan and K.-C. Chua, "Trade-offs between mobility and density for coverage in wireless sensor networks," in *ACM MobiCom*, 2007.

[11] G. Wang, G. Cao, P. Berman, and T. F. La Porta, "Bidding protocols for deploying mobile sensors," *IEEE Transactions on Mobile Computing*, vol. 6, no. 5, pp. 563–576, 2007.

[12] D. Kim, W. Wang, J. Son, W. Wu, W. Lee, and A. O. Tokuta, "Maximum lifetime combined barrier-coverage of weak static sensors and strong mobile sensors," *To be published in IEEE Transactions on Mobile Computing*, 2016.

[13] B. Xu, D. Kim, D. Li, J. Lee, H. Jiang, and A. O. Tokuta, "Fortifying barrier-coverage of wireless sensor network with mobile sensor nodes," in *International Conference on Wireless Algorithms, Systems, and Applications*, 2014.

[14] J. Czyzowicz, E. Kranakis, D. Krizanc, I. Lambadaris, L. Narayanan, J. Opatrny, L. Stacho, J. Urrutia, and M. Yazdani, "On minimizing the maximum sensor movement for barrier coverage of a line segment," in *Ad-Hoc, Mobile and Wireless Networks*, 2009.

[15] ——, "On minimizing the sum of sensor movements for barrier coverage of a line segment," in *Ad-Hoc, Mobile and Wireless Networks*, 2010.