Tracking on a Graph*

Songhwai Oh and Shankar Sastry
Department of Electrical Engineering and Computer Sciences
University of California, Berkeley, CA 94720
{sho,sastry}@eecs.berkeley.edu

***Abstract*— This paper considers the problem of tracking objects with sparsely located binary sensors. Tracking with a sensor network is a challenging task due to the inaccuracy of sensors and difficulties in sensor network localization. Based on the simplest sensor model, in which each sensor reports only a binary value indicating whether an object is present near the sensor or not, we present an *optimal distributed* tracking algorithm which does not require sensor network localization. The tracking problem is formulated as a hidden state estimation problem over the finite state space of sensors. Then a distributed tracking algorithm is derived from the Viterbi algorithm. We also describe provably good pruning strategies for scalability of the algorithm and show the conditions under which the algorithm is robust against false detections. The algorithm is also extended to handle non-disjoint sensing regions and to track multiple moving objects. Since the computation and storage of track information are done in a completely distributed manner, the method is robust against node failures and transmission failures. In addition, the use of binary sensors makes the proposed algorithm suitable for many sensor network applications.**

* Published in the Proc. of the Fourth International Conference on Information Processing in Sensor Networks (IPSN), 2005.

I. Introduction

In wireless ad-hoc sensor networks, many inexpensive and small sensor-rich devices are deployed to monitor and control our environment. It is envisioned that sensor networks will connect us to the physical world in a pervasive manner [1], [2], [3]. Each device, called a sensor node, has sensing, processing, and communication capabilities. Sensor nodes form a wireless ad-hoc network for communication. The limited supply of power and other constraints, such as manufacturing costs and limited package sizes, limit the capabilities of each sensor node. For example, a typical sensor node has short communication and sensing ranges, a limited amount of memory and limited computational power. However, the abundant number of spatially spread sensors will enable us to monitor changes in our environment accurately despite the inaccuracy of each sensor node.

Target tracking is a representative application of sensor networks as it exhibits different aspects of sensor networks such as event detection, sensor information fusion, communication, sensor management, and decision making. The applications of tracking using sensor networks include surveillance, search and rescue, disaster response system, pursuit evasion games [4], spatio-temporal data collection, and other location-based services [5]. However, the task of tracking with sensor networks is complicated by two problems: the inaccuracy of sensors and difficulties in sensor network localization.

Each sensor node has a limited supply of power and operates in the low SNR regime, leading to low detection probability and high false detection probabilities. The presence of false detections and missing observations complicates track initiation, maintenance, and termination. These important issues are ignored by many tracking algorithms designed for sensor networks. Instead of using an ideal sensor model which ignores the issues listed above, we take the minimalist approach and use the binary sensor model, in which each sensor reports only a binary value indicating whether an object is present near the sensor or not.

Tracking algorithms designed for sensor networks so far rely on sensor network localization [6], [7], [8], [9], [10]. Although many sensor network localization algorithms have been proposed and tested, it is still a challenging problem in practice [11], [12]. Furthermore, even if the exact distances between pairs of nodes are known, the localization in sparse networks is shown to be NP-hard [13]. Other alternatives, such as the manual deployment or the global positioning system (GPS), require expensive hardware and are not scalable. Since we cannot avoid localization errors, the localization-dependent tracking algorithms will frequently fail to track a moving object. In this paper, we develop a distributed tracking algorithm based on the formulation which is independent of sensor network localization. Hence, the performance of our algorithm is not affected by the localization error.

The continuous state-space, dynamic-based tracking algorithms, such as the Kalman filter and a particle filter, require reasonably good measurements of the states of a moving object. But, due to the inaccuracy of sensors and difficulties in sensor network localization, it is difficult to fulfill such requirements with currently available sensor networks. In addition, when an object moves with extremely complex dynamics or when we are tracking with sparsely located sensors, it is difficult to use a continuous state-space, dynamic-based tracking algorithm. For example, the task of tracking people indoor requires a complex dynamics model and the dynamic-based tracking algorithms cannot be easily applied. We address this issue by developing a tracking algorithm over the finite state space of sensors. Under this new formulation, we can track objects without requiring an ideal sensor model and sensor network localization.

In this paper, the tracking problem is formulated as a hidden state estimation problem in a hidden Markov model over

the finite state space of sensors. Then an optimal distributed tracking algorithm is derived from the Viterbi algorithm [14]. We then show provably good pruning strategies for scalability and the conditions under which the algorithm is robust against false detections. The algorithm is also extended to handle non-disjoint sensing regions and to track multiple moving objects. Since the computation and storage of track information are done in a completely distributed manner, the method is robust against node failures and transmission failures. In addition, the use of binary sensors makes the proposed algorithm suitable for many sensor network applications such as location-based services. The algorithm presented in this paper is suitable for indoor tracking problems where the movement of an object cannot be easily modeled and sensor network self-localization is difficult. The algorithm can be applied to outdoor tracking problems if the maximum degree of the passage connectivity graph of a sensor network (see Section II for its definition) is not too large. For arbitrary outdoor tracking scenarios, a hierarchical multiple-target tracking algorithm for sensor network is developed in [15].

The remainder of this paper is structured as follows. The problem of tracking on a graph is described in Section II. The optimal distributed tracking algorithm is presented in Section III and provably good pruning strategies for scalability is given in Section IV. We study the robustness of the algorithm in the presence of false detections in Section V. The algorithm is extended to handle non-disjoint sensing regions and to track multiple objects in Section VI and VII.

II. Problem Formulation

Suppose there are N sensor nodes, $s_1, \ldots, s_N$. We denote the sensing region of sensor node s_i by C_i and assume that the sensing region of each sensor node is disjoint from the sensing regions of other sensor nodes. A sensor node s_i is *passage connected* to a sensor node s_j if there is a path from the sensing region of s_i to the sensing region of s_j such that an object can traverse. A sensor node s_i is *direct passage connected* to a sensor node s_j if s_i is passage connected to s_j and there is a path from s_i to s_j which does not intersect sensing regions of the remaining sensors. We assume that the passage connectivity (direct passage connectivity) is symmetric, *i.e.*, if s_i is passage connected (direct passage connected) to s_j, then s_j is passage connected (direct passage connected) to s_i. Note that when there is no confusion, we will denote sensor s_i by its index i.

Definition 1: A *passage connectivity graph* of sensor nodes $s_1, \ldots, s_N$ is a graph $G = (V, E)$, where $V = \{1, \ldots, N\}$ and $(u, v) \in E$ if and only if u and v are direct passage connected.

Let $G = (V, E)$ be the passage connectivity graph of sensor nodes $s_1, \ldots, s_N$. An example of a passage connectivity graph is shown in Figure 1. Notice that since $u \in V$ is direct passage connected to u, $(u, u) \in E$. Throughout this paper, without loss of generality, we assume that G is a connected graph, since, if G is disconnected, we can consider each partition of G separately. Let $X_t \in \{1, \ldots, N\}$ be the location of an object on G at time t for $t \in \{1, \ldots, T\}$, where T is the time horizon. An object appears at $i \in V$ with the initial state probability $\pi_i = P(X_1 = i)$. If $X_t = i$, an object is located in C_i at time t.

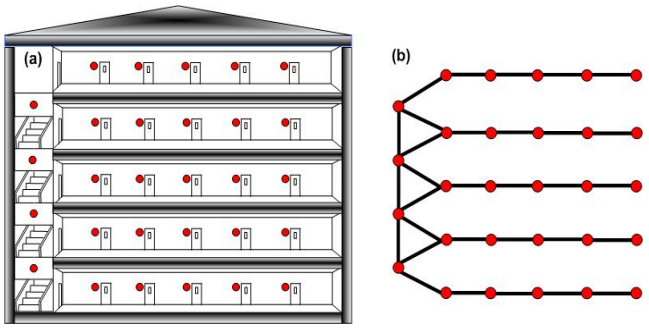


Fig. 1. (a) An example of indoor sensor network deployment (solid circles are sensors). (b) The passage connectivity graph of the sensor network shown in (a).

Let $\text{NB}_i = \{j \in V : (i, j) \in E\}$ be the neighborhood of i. We assume that the evolution of an object on G is Markovian such that $P(X_{t+1} = j | X_t = i, X_{t-1}, \ldots, X_1) = P(X_{t+1} = j | X_t = i) := p_{ij}$ for all t. Let $P_{\text{t}} = [p_{ij}]$ be the transition probability matrix. For each i, $\sum_{j \in \text{NB}_i} p_{ij} = 1$ and $p_{ij} = 0$ for $j \notin \text{NB}_i$. Notice that our framework can handle more complex dynamics of a moving object by using a k-th order Markov chain, for $k \geq 2$. For each sensor node s_i, if an object is in C_i, it is detected with probability η_i. Let $Y_t^i \in \{0, 1\}$ be the observation made by s_i at time t. When an object is in C_i at time t, the object is detected, $Y_t^i = 1$, with probability η_i, and the object is not detected, $Y_t^i = 0$, with probability $1 - \eta_i$. For now we assume there are no false detections. We show the robustness of our algorithm in the presence of false detections in Section V. The model parameters, π, P_{t}, and η, can be learned from historic data using Baum-Welch method [14]. So we assume that they are known in this paper. We also assume that node u can communicate with node v reliably in timely manner if $(u, v) \in E$.

Since the sensing regions are disjoint, we can assume that each sensor makes an independent observation at each time. Let $Y_t = (Y_t^1, \ldots, Y_t^N)^T$, $Y_{1:T} = \{Y_1, \ldots, Y_T\}$ and $X_{1:T} = \{X_1, \ldots, X_T\}$. The joint distribution is

$$P(X_{1:T}, Y_{1:T}) = P(X_1) \prod_{t=2}^{T} P(X_t | X_{t-1}) \prod_{t=1}^{T} P(Y_t | X_t), \quad (1)$$

where

$$P(Y_t | X_t) = (\eta_{X_t})^{Y_t^{X_t}} (1 - \eta_{X_t})^{1 - Y_t^{X_t}}. \quad (2)$$

Now the tracking problem on a graph is to find the most probable trajectory of a moving object on the graph given observations $y_{1:T}$, *i.e.*, finding the maximum a posteriori

(MAP) solution $x^*_{1:T}$:

$$\begin{aligned} x^*_{1:T} &= \arg\max_{x_{1:T}} P(x_{1:T}|y_{1:T}) \\ &= \arg\max_{x_{1:T}} \frac{P(y_{1:T}|x_{1:T})P(x_{1:T})}{P(y_{1:T})} \\ &= \arg\max_{x_{1:T}} P(y_{1:T}|x_{1:T})P(x_{1:T}) \\ &= \arg\max_{x_{1:T}} P(x_{1:T}, y_{1:T}). \end{aligned} \quad (3)$$

The Bayes rule is used for the second equality and the normalization constant $P(y_{1:T})$ is dropped in the third equality since it does not change the maximization problem.

III. Optimal Distributed Tracking Algorithm

The most probable sequence of hidden states $x^*_{1:T}$ given observations $y_{1:T}$ can be efficiently found by the Viterbi algorithm [14]. We first define

$$\delta_t(i) = \max_{x_1,\ldots,x_{t-1}} P(x_1, \ldots, x_{t-1}, x_t = i, y_{1:t}), \quad (4)$$

for $i = 1, \ldots, N$ and $t = 1, \ldots, T$. $\delta_t(i)$ is the maximum probability of a single path ending in state i at time t, given observations $y_{1;t}$. Once we have all $\delta_T(i)$, we can trace backward from the state which maximizes $\delta_T(i)$ to find the sequence $x^*_{1:T}$. For more detail about the Viterbi algorithm, see [14].

We compactly denote the event $\{x_t = i\}$ by z^i_t. Notice that

$$\begin{aligned} \delta_t(i) &= \max_{x_{1:t-1}} P(x_{1:t-1}, z^i_t, y_{1:t}) \\ &= \max_{x_{1:t-1}} P(z^i_t|x_{t-1})P(y_t|z^i_t)P(x_{1:t-1}, y_{1:t-1}) \\ &= \left[\max_{x_{1:t-1}} P(z^i_t|x_{t-1})P(x_{1:t-1}, y_{1:t-1})\right] P(y_t|z^i_t) \\ &= \left[\max_{j\in\{1,\ldots,N\}} P(z^i_t|z^j_{t-1}) \max_{x_{1:t-2}} P(x_{1:t-2}, z^j_{t-1}, y_{1:t-1})\right] \\ &\times\ P(y_t|z^i_t) \\ &= \left[\max_{j\in \mathrm{NB}_i} P(z^i_t|z^j_{t-1})\delta_{t-1}(j)\right] P(y_t|z^i_t), \end{aligned} \quad (5)$$

where the domain of the maximization is reduced from $\{1, \ldots, N\}$ to NB_i in the last equality since $P(z^i_t|z^j_{t-1}) = 0$ for $j \notin \mathrm{NB}_i$ ($i \notin \mathrm{NB}_j$ by symmetry). Hence, in order to compute $\delta_t(i)$, we only need $\{\delta_{t-1}(j) : j \in \mathrm{NB}_i\}$ from the neighbors of s_i, transition probabilities and the likelihood $P(y_t|z^i_t)$.

Since $P(y^i_t = 1|z^i_t) = \eta_i$ and $P(y^i_t = 0|z^i_t) = 1 - \eta_i$, (5) can be further simplified as

$$\delta_t(i) = \left[\max_{j\in \mathrm{NB}_i} P(z^i_t|z^j_{t-1})\delta_{t-1}(j)\right] \eta_i^{y^i_t}(1-\eta_i)^{1-y^i_t}. \quad (6)$$

Thus, we can compute $\delta_t(i)$ with only local information. The distributed implementation to find the most probable trajectory of a moving object is composed of two steps: tracking and backtracking.

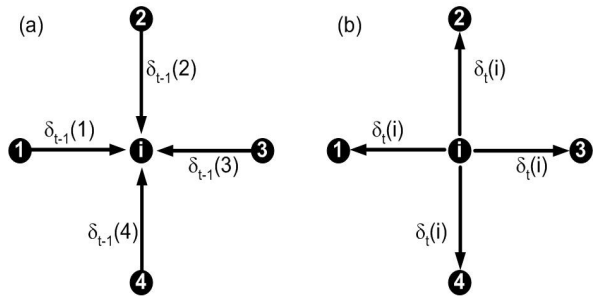


Fig. 2. A graphical illustration of tracking. (a) Node i receives δ_{t-1} from its neighbors (step 2a). (b) Node i transmits $\delta_t(i)$ to its neighbors (step 2c)

A. Tracking

We introduce variables $\psi_t(i)$ to keep the best previous location of an object for the backtracking step. For each sensor node i and $t = 1, 2, \ldots, T$,

1) If $t = 1$,
   a) compute
   $$\begin{aligned} \delta_1(i) &= \pi_i \eta_i^{y^i_1}(1-\eta_i)^{1-y^i_1} \\ \psi_1(i) &= 0; \end{aligned}$$
   b) transmit $\delta_1(i)$ to NB_i.
2) If $t > 1$,
   a) receive $\{\delta_{t-1}(j) : j \in \mathrm{NB}_i\}$;
   b) compute
   $$\begin{aligned} \delta_t(i) &= \left[\max_{j\in \mathrm{NB}_i} P(z^i_t|z^j_{t-1})\delta_{t-1}(j)\right] \eta_i^{y^i_t}(1-\eta_i)^{1-y^i_t} \\ \psi_t(i) &= \arg\max_{j\in \mathrm{NB}_i} P(z^i_t|z^j_{t-1})\delta_{t-1}(j); \end{aligned}$$
   c) transmit $\delta_t(i)$ to NB_i.

The tracking step of the algorithm is graphically illustrated in Figure 2. In Figure 2 (a), node i receives δ_{t-1} from its neighbors. After computing $\delta_t(i)$ and $\psi_t(i)$, node i transmits $\delta_t(i)$ to its neighbors (Figure 2 (b)).

B. Backtracking

Suppose that the sensor node s_b requests a trajectory of the moving object at time T. Since trajectory information $\psi_t(i)$ is stored in a distributed manner, we backtrack to recover the most probable trajectory by collecting vertices along the path directed by $\psi_t(i)$. For each sensor node i and $t = T, T-1, \ldots, 1$,

1) If $t = T$
   a) compute $\phi_T(i) = \delta_T(i)$;
   b) transmit $(\phi_T(i), x^*_T(i) = \{i\})$ to $\psi_T(i)$.
2) If $1 \le t < T$
   a) receive $D_i = \{(\phi_{t+1}(j), x^*_{t+1:T}(j)) : j \in J_i\}$, where $J_i = \{j \in \mathrm{NB}_i : \psi_{t+1}(j) = i\}$;
   b) if $D_i \neq \emptyset$,

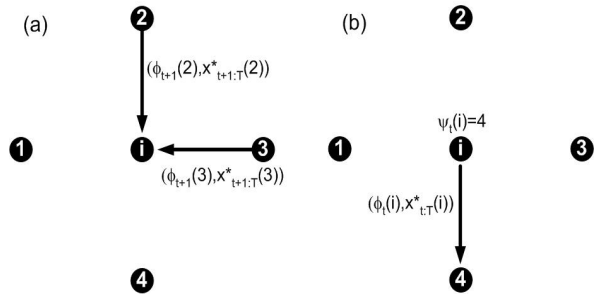


Fig. 3. A graphical illustration of backtracking. (a) Node i receives $(\phi_{t+1}, x^*_{t+1:T})$ from its neighbors (step 2a). (b) Node i transmits $(\phi_t(i), x^*_{t:T}(i))$ to $\psi_t(i)$ (step 2(b)ii)

i) compute

$$\begin{aligned} j &= \arg\max_{k\in D_i} \phi_{t+1}(k) \\ \phi_t(i) &= \phi_{t+1}(j) \\ x^*_{t+1:T}(i) &= x^*_{t+1:T}(j) \\ x^*_t(i) &= i; \end{aligned}$$

ii) if $t > 1$, transmit $(\phi_t(i), x^*_{t:T}(i))$ to $\psi_t(i)$;
iii) if $t = 1$, transmit $x^*_{1:T}(i)$ to $s_{\rm b}$.

The trajectory $x^*_{t:T}(i)$ is simply concatenated at each step and a complete trajectory is available when the backtracking is terminated. However, an alternative approach is to transmit x^*_t to $s_{\rm b}$ at each t and transmit only $\phi_t(j)$ to $\psi_t(i)$ so that the transmission packets are of the same size. The backtracking step of the algorithm is graphically illustrated in Figure 3. In Figure 3 (a), node i receives $(\phi_{t+1}, x^*_{t+1:T})$ from its neighbors. After the step 2(b)i, node i transmits $(\phi_t(i), x^*_{t:T}(i))$ to $\psi_t(i)$ (Figure 3 (b)).

C. Discussion

Notice that in order to carry out above calculations, node i needs to know only π_i, η_i, and $P_{\rm t}$ restricted to NB_i and the remaining variables are computed on the fly. Hence, the calculations required for tracking and backtracking can be done in a completely distributed manner. The algorithm implements the Viterbi algorithm [14] and finds the most probable trajectory $x^*_{1:T}$. Since the computation and storage of track information are done in a completely distributed manner, the method is robust against node failures and transmission failures. If d is the maximum degree of the graph G, the number of transmissions at each time is bounded above by dN and does not depend on T. But the memory required to keep track information increases as T increases. For a large graph, the number of sensors involved in tracking increases as T increases. Hence, in order to bound the memory requirement and the number of sensors involved in tracking and make the algorithm scalable, we need a method to prune away unlikely tracks. For this purpose, we present provably good pruning strategies in Section IV.

IV. Pruning

The main idea behind the pruning strategies discussed in this section is to prune a trajectory hypothesis if the number of detections is low. Since pruning is performed, based solely on local information, it is important to analyze the performance loss of the algorithm for the chosen pruning strategy. We first present the updated tracking and backtracking steps and then describe pruning strategies such that, for given $\epsilon > 0$, the algorithm finds an optimal trajectory $x^*_{1:T}$ with probability at least $1 - \epsilon$. An effect of pruning is shown in Figure 4 where the pruning strategy given in Theorem 1 is used.

A. Tracking with Pruning

We introduce a variable $\varphi_t(i)$ to keep the number of detections along the path directed by $\psi_t(i)$. A track hypothesis is pruned away if $g(t, \varphi_t(i)) = 1$ and the choice of $g(t, \varphi_t(i))$ is discussed in Section IV-C. $\delta_t(i)$ and $\psi_t(i)$ are all initialized to zeros.

1) If $t = 1$,
   a) compute

$$\begin{aligned} \delta_1(i) &= \pi_i \eta_i^{y_1^i}(1-\eta_i)^{1-y_1^i} \\ \psi_1(i) &= 0 \\ \varphi_1(i) &= y_1^i; \end{aligned}$$

   b) transmit $(\delta_1(i), \varphi_1(i))$ to NB_i.
2) If $t > 1$,
   a) receive $D_i = \{\delta_{t-1}(j), \varphi_{t-1}(j) : j \in \mathrm{NB}_i\}$;
   b) if $D_i \neq \emptyset$,
      i) compute

$$\begin{aligned} \delta_t(i) &= \left[\max_{j\in\mathrm{NB}_i} P(z_t^i|z_{t-1}^j)\delta_{t-1}(j)\right] \eta_i^{y_t^i}(1-\eta_i)^{1-y_t^i} \\ \psi_t(i) &= \arg\max_{j\in\mathrm{NB}_i} P(z_t^i|z_{t-1}^j)\delta_{t-1}(j) \\ \varphi_t(i) &= \varphi_{t-1}(\psi_t(i)) + y_t^i; \end{aligned}$$

      ii) if $g(t, \varphi_t(i)) = 0$, transmit $(\delta_t(i), \varphi_t(i))$ to NB_i.

B. Backtracking with Pruning

Backtracking is the same as Section III-B but the step 1.(b) is replaced by

1.(b') if $\phi_T(i) > 0$, transmit $(\phi_T(i), x^*_T(i) = \{i\})$ to $\psi_T(i)$.

C. Pruning Strategies

This section gives two pruning strategies such that, for given $\epsilon > 0$, the algorithm finds an optimal trajectory $x^*_{1:T}$ with probability at least $1-\epsilon$. Let W_t be the event that the algorithm terminated at time t returns an incorrect solution, *i.e.*, when a correct solution is pruned away. The omitted proofs appear in Appendix.

Theorem 1 (Finite Horizon): Let G be a passage connectivity graph of sensor nodes $s_1, \ldots, s_N$. Suppose that $\eta_i > 0$, for all $i = 1, \ldots, N$, and let $\eta = \min \eta_i$. For $\epsilon > 0$ and $T > 0$, choose $s \in \mathbb{N}$, such that $\frac{T}{\epsilon}\exp\left(-\frac{\eta t_0}{2}\right) < s \leq T$

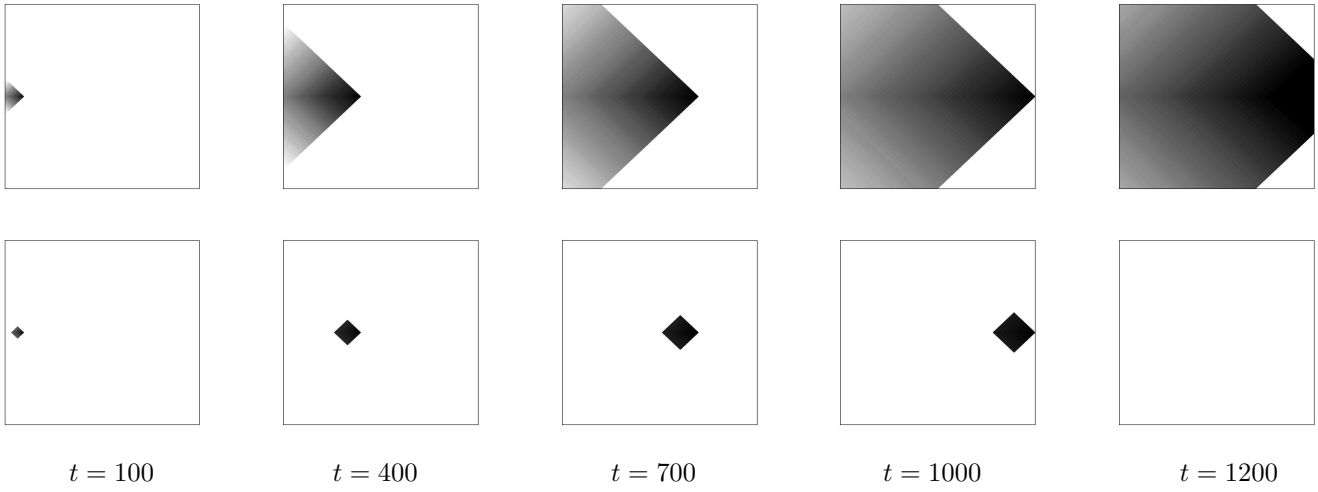


Fig. 4. An effect of pruning: an object moves horizontally along the center row of 1000×1000 sensor grid ($\eta = .7$). A sensor node involved in tracking is painted with darker color. (Top) Without pruning. (Bottom) With pruning ($t_0 = 100$, $s = 100$ and $\epsilon = .05$).

for some $t_0 \geq 1$, and choose $\theta_t \leq \eta t \left(1 - \sqrt{\frac{2}{\eta t} \log\left(\frac{T}{s\epsilon}\right)}\right)$. If the following pruning strategy,

$$g(t, \varphi_t(i)) = \begin{cases} 1 & \text{if } \varphi_t(i) < \theta_t \text{ for } t = ks \geq t_0, k \in \mathbb{N} \\ 0 & \text{otherwise,} \end{cases} \tag{7}$$

is used, then $P(W_T) < \epsilon$.

The following theorem shows an existence of a pruning strategy when $T \to \infty$.

Theorem 2 (Infinite Horizon): Let G be a passage connectivity graph of sensor nodes $s_1, \ldots, s_N$. Suppose that $\eta_i > 0$, for all $i = 1, \ldots, N$, and let $\eta = \min \eta_i$. For $\epsilon > 0$, choose $s \in \mathbb{N}$, such that $s > \frac{2}{\eta} \log\left(\frac{1+\epsilon}{\epsilon}\right)$, and choose $\theta_t \leq \eta t \left(1 - \sqrt{\frac{2}{\eta s} \log\left(\frac{1+\epsilon}{\epsilon}\right)}\right)$. If the following pruning strategy

$$g(t, \varphi_t(i)) = \begin{cases} 1 & \text{if } \varphi_t(i) < \theta_t \text{ for } t = ks \geq s, k \in \mathbb{N} \\ 0 & \text{otherwise,} \end{cases} \tag{8}$$

is used, then $P(W_\infty) < \epsilon$.

V. Robustness

In previous sections, we have assumed that there are no false detections in our observation model. In this section, we study the robustness of the algorithm in the presence of false detections. We assume that the algorithm is run without pruning so if a pruning strategy is used, unless stated otherwise, the arguments here should be understood in a probabilistic sense, *i.e.*, given $\epsilon > 0$ and a pruning strategy given in Section IV, the arguments hold with probability larger than $1 - \epsilon$. Let ξ_i be the false detection probability for the sensor s_i. Thus, when an object is not in C_i at time t, the sensor s_i makes a false detection, $Y_t^i = 1$, with probability ξ_i and $Y_t^i = 0$ with probability $1 - \xi_i$. We need to update the joint distribution (1) as following to allow false detections.

$$P(X_{1:T}, Y_{1:T}) = P(X_1) \prod_{t=2}^{T} P(X_t | X_{t-1}) \prod_{t=1}^{T} P(Y_t | X_t), \tag{9}$$

where

$$\begin{aligned} P(Y_t | X_t) &= \textstyle\prod_{i=1}^{N} P(Y_t^i | X_t) \\ &= (\eta_{X_t})^{Y_t^{X_t}} (1 - \eta_{X_t})^{1 - Y_t^{X_t}} \textstyle\prod_{j \neq X_t} (\xi_j)^{Y_t^j} (1 - \xi_j)^{1 - Y_t^j}. \end{aligned}$$

Unfortunately, we can no longer compute the likelihood $P(Y_t | X_t)$ with only local information as shown in (2). We need observations from all sensors in order to compute the likelihood and it is difficult to guarantee the optimality of any distributed algorithm. We focus on the case, in which the detection probabilities and the false detection probabilities are uniform, for its simplicity and show the optimality conditions for the algorithm given in Section III in the presence of false detections.

Theorem 3: Let G be a passage connectivity graph of sensor nodes $s_1, \ldots, s_N$. Suppose that $\eta = \eta_i \geq .5$ and $\xi = \xi_i \leq .5$, for all $i = 1, \ldots, N$ and $m = \min P(x_{1:T}) > 0$. Let $q_{1:T}$ be the solution computed by the algorithm given in Section III and let $x^*_{1:T}$ be the optimal solution, an MAP estimate to (9) given $y_{1:T}$. Then the following inequalities hold

$$P(q_{1:T}, y_{1:T}) \leq P(x^*_{1:T}, y_{1:T}) \leq \alpha P(q_{1:T}, y_{1:T}), \tag{10}$$

where $\alpha = \left(\frac{1-\xi}{\xi}\right)^{\frac{\log(P(q_{1:T})/m)}{\log(\eta/(1-\eta))}}$.

In Theorem 3, $\alpha \geq 1$ since $\xi \leq .5$. To get a tighter bound, we want α to be as small as possible. Notice that when $P(q_{1:T})$ is small, α is small and we can be confident that the estimate $q_{1:T}$ from the algorithm given in Section III is close to the MAP estimate $x^*_{1:T}$. An extreme case is when $\xi = .5$. Then $\alpha = 1$ and $q_{1:T} = x^*_{1:T}$. However, (10) is the worst-case bound. To increase the performance of the algorithm for an average case, we require small ξ. Hence, we assume that ξ is fixed at some small value and discuss approaches to make the worst-case bound tight.

Corollary 1: Let G be a passage connectivity graph of sensor nodes $s_1, \ldots, s_N$. Suppose that G is a regular graph

with a uniform initial state distribution and uniform transition probabilities. Then, under the conditions specified in Theorem 3, $q_{1:T} = x^*_{1:T}$.

Proof: G is a regular graph with a uniform initial state distribution and uniform transition probabilities so we have a uniform prior. In particular, $P(q_{1:T}) = P(x^*_{1:T})$. Hence, $\alpha = 1$ and $q_{1:T} = x^*_{1:T}$. □

Corollary 2: Assume the conditions specified in Theorem 3 and let $r = \max P(x_{1:T}) / \min P(x_{1:T})$. For $\epsilon_1 > 0$, let $c = \exp\left(\log r \log\left(\frac{1-\xi}{\xi}\right) / \log(1+\epsilon_1)\right)$. If $\eta \geq \frac{c}{1+c}$, then $P(q_{1:T}, y_{1:T}) \leq P(x^*_{1:T}, y_{1:T}) \leq (1+\epsilon_1) P(q_{1:T}, y_{1:T})$.

Proof: Since $\eta \geq \frac{c}{1+c}$, $\frac{\eta}{1-\eta} \geq c$. So

$$\begin{aligned}
\alpha &= \left(\frac{1-\xi}{\xi}\right)^{\frac{\log(P(q_{1:T})/m)}{\log(\eta/(1-\eta))}} \\
&\leq \left(\frac{1-\xi}{\xi}\right)^{\frac{\log r}{\log(\eta/(1-\eta))}} \\
&\leq \left(\frac{1-\xi}{\xi}\right)^{\frac{\log r}{\log c}} \\
&= \left(\frac{1-\xi}{\xi}\right)^{\frac{\log(1+\epsilon_1)}{\log\left(\frac{1-\xi}{\xi}\right)}} \\
&= 1+\epsilon_1.
\end{aligned}$$

□

Hence, an ideal case is when r is small and η is big (as specified in Corollary 2). For example, when $r = 1000$ and $\xi = .45$, it requires $\eta \geq .8808$ for $\epsilon_1 = 1$ and $\eta \geq .7793$ for $\epsilon_1 = 2$. But if $r = 1000$ and $\xi = .3$, we need $\eta \geq .9998$ for $\epsilon_1 = 1$ and $\eta \geq .9952$ for $\epsilon_1 = 2$. We note that, for binary sensors, the detection probability can be boosted to a desired value by deploying multiple sensors over the same surveillance region.

VI. Non-disjoint Sensing Regions

Until now, we have assumed that the sensing regions of sensors are disjoint, leading to independent observations. However, this assumption may not be satisfied in all cases. In some cases, we may want to introduce more sensors to improve the detection probability. In this section, we describe how to handle non-disjoint sensing regions.

Suppose that G is a passage connectivity graph. For the simplicity of exposition, we consider the case when the sensing regions of two sensors A and B are not disjoint (see Figure 5). However, the method can be applied to the case with more than two sensors. Consider all disjoint segments of $C_A \cup C_B$, *i.e.*, $C_{A\setminus B} = C_A \setminus C_B$, $C_{B\setminus A} = C_B \setminus C_A$ and $C_{A\cap B} = C_A \cap C_B$ (see Figure 6). Recall that C_A is the sensing region of sensor A. In this example, an extra vertex $A \cap B$ is added into G along with new edges from $A \cap B$ (see Figure 6). The sensor nodes A and B exchange observations and the necessary computations at $A \cap B$ can be distributed between A and B. The additional parameters of this updated graph can be learned from historic data using Baum-Welch method as mentioned earlier. However, if the sensor model of each sensor node is known, we can compute the sensor models of each disjoint segment. Let η_A and η_B be the detection probabilities of sensor A and sensor B, respectively, and let ξ_A and ξ_B be the false detection probabilities of sensor A and sensor B, respectively. Let Y_A and Y_B be the observations made by sensor A and sensor B, respectively, and let X be the true position of the object. Then the sensor models of disjoint segments can be computed as shown in Table I. Now tracking can be done as discussed in Section III and IV.

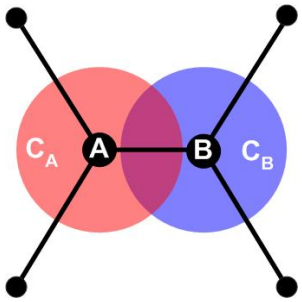


Fig. 5. An example of non-disjoint sensing regions. Sensor nodes are shown in dots and disks around sensors represent sensing regions

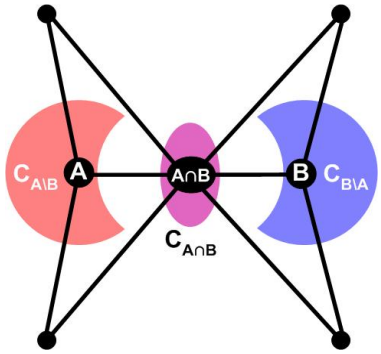


Fig. 6. A partition of sensing regions from the example shown in Figure 5

VII. Multiple Object Tracking

Suppose there are K independently moving objects and assume that each sensor can identify one object from another. Let Y^i_t be an observation made by sensor s_i at time t, which takes a value from a power set of $\{1, \ldots, K\}$. Then we can express the sensor model as, for all $k = 1, \ldots, K$ and $i = 1, \ldots, N$,

$$\begin{aligned}
P(Y^i_t \ni k | X^k_t = i) &= \eta^k_i \\
P(Y^i_t \not\ni k | X^k_t = i) &= 1 - \eta^k_i,
\end{aligned} \tag{11}$$

where $X^k_t \in \{1, \ldots, N\}$ is the location of the object k at time t and η^k_i is the detection probability at sensor i for object k. Then we can individually track each object using the algorithm given in Section III and IV. For the problem of tracking multiple objects without classification information, we refer readers to the hierarchical multiple-target tracking algorithm for sensor networks [15].

TABLE I

SENSOR MODEL $P(Y_A, Y_B|X)$ OF DISJOINT SEGMENTS IN FIGURE 6

	$Y_A = 0, Y_B = 0$	$Y_A = 0, Y_B = 1$	$Y_A = 1, Y_B = 0$	$Y_A = 1, Y_B = 1$
$X \in C_{A\setminus B}$	$(1-\eta_A)(1-\xi_B)$	$(1-\eta_A)\xi_B$	$\eta_A(1-\xi_B)$	$\eta_A\xi_B$
$X \in C_{B\setminus A}$	$(1-\xi_A)(1-\eta_B)$	$(1-\xi_A)\eta_B$	$\xi_A(1-\eta_B)$	$\xi_A\eta_B$
$X \in C_{A\cap B}$	$(1-\eta_A)(1-\eta_B)$	$(1-\eta_A)\eta_B$	$\eta_A(1-\eta_B)$	$\eta_A\eta_B$

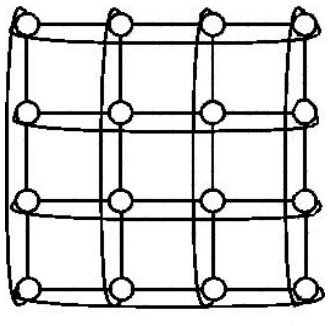


Fig. 7. A toroidal grid

A. Simulation

For simulation, we consider a toroidal grid passage connectivity graph G. An example of a toroidal grid is shown in Figure 7. We assume there are two objects and they all have the uniform initial state distribution while $p_{ii} = .1$ and $p_{ij} = .9/|\text{NB}_i \setminus \{i\}|$ for $j \in \text{NB}_i \setminus \{i\}$. The detection probabilities are randomly chosen from $[.7, 1)$. However, we consider the situation described in [16], [9], in which a sensor does not correctly classify an object when multiple objects are present in the same sensing region. So when multiple objects are present in the same sensing region, we assume that the corresponding sensor does not report detection. We approximate this situation by the sensing model (11). Two scenarios are considered and they are shown in Figure 8 (a) and (c). The estimated tracks are shown in Figure 8 (b) and (d). The estimated tracks are the same as the true trajectories when objects are detected while the estimated tracks follow the path with maximum transition probabilities and minimum detection probabilities when objects are not detected.

VIII. CONCLUSION

In this paper, we have described a novel tracking algorithm for sensor network. The tracking problem is formulated as a hidden state estimation problem in a hidden Markov model over the finite state space of sensors. Then an optimal distributed tracking algorithm is derived from the Viterbi algorithm. The algorithm finds an optimal solution in a fully distributed manner. Furthermore, the algorithm is based on the simple binary sensors and does not depend on sensor network localization. We have described provably good pruning strategies for scalability of the algorithm and showed the conditions under which the algorithm is robust against false detections. We have also presented extensions of the algorithm to handle non-disjoint sensing regions and to track multiple objects. Since the computation and storage of track information are done in a completely distributed manner, the method is robust against node failures and transmission failures.

ACKNOWLEDGMENT

This material is based upon work supported by the National Science Foundation under Grant No. EIA-0122599.

APPENDIX

A. Proof of Theorem 1

We first need the following lemma to prove Theorem 1 and Theorem 2.

Lemma 1: Let $Y_1, \ldots, Y_t$ be independent random variables such that $P(Y_t = 1) = p_t$ and $P(Y_t = 0) = 1 - p_t$, for arbitrary $0 < p_t < 1$. Let Z_t be a Bernoulli process with parameter p and, for all t, $p \le p_t$. Let $S_Y^t = \sum_{k=1}^t Y_k$ and $S_Z^t = \sum_{k=1}^t Z_k$. Then, for all $t \in \mathbb{N}$ and $d \le t$,

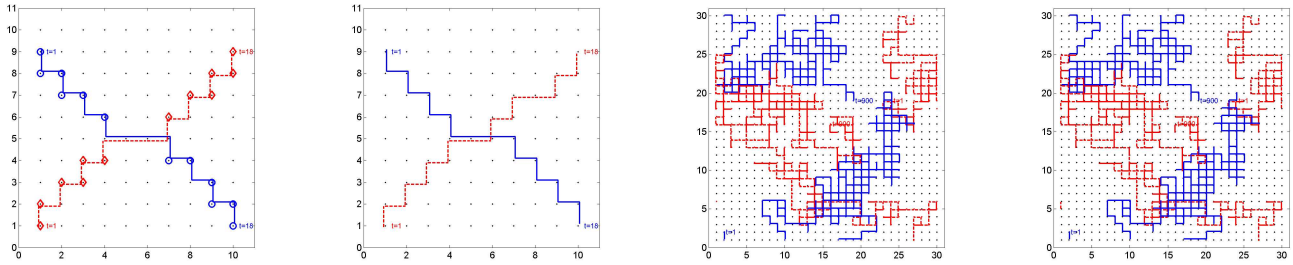
$$P(S_Y^t < d) \le P(S_Z^t < d). \tag{12}$$

Proof: We prove the claim by induction. For $t = 1$, $(1-p_1) \le (1-p)$. So $P(Y_1 < 1) \le P(Z_1 < 1)$. Now suppose that (12) is true for t. Then, for $t+1$ and any $d \le t+1$, we have, using the induction hypothesis,

$$\begin{aligned}
P(S_Y^{t+1} < d) &= P(S_Y^t + Y_{t+1} < d) \\
&= P(S_Y^t + Y_{t+1} < d | Y_{t+1} = 0) P(Y_{t+1} = 0) \\
&+ P(S_Y^t + Y_{t+1} < d | Y_{t+1} = 1) P(Y_{t+1} = 1) \\
&= P(S_Y^t < d) P(Y_{t+1} = 0) \\
&+ P(S_Y^t < d-1) P(Y_{t+1} = 1) \\
&\le P(S_Z^t < d) P(Y_{t+1} = 0) \\
&+ P(S_Z^t < d-1) P(Y_{t+1} = 1) \\
&= P(S_Z^t < d) - P(S_Z^t = d-1) P(Y_{t+1} = 1) \\
&\le P(S_Z^t < d) - P(S_Z^t = d-1) P(Z_{t+1} = 1) \\
&= P(S_Z^t < d) P(Z_{t+1} = 0) \\
&+ P(S_Z^t < d-1) P(Z_{t+1} = 1) \\
&= P(S_Z^{t+1} < d).
\end{aligned}$$

□

We now prove Theorem 1. Let Z_t be a Bernoulli process with parameter η and let $S_Z^t = \sum_{k=1}^t Z_k$. Let $X_t \in \{1, \ldots, N\}$ be a Markov chain on G with the initial state distribution and transition probabilities given in Section II. Let Y_t be an observation made at time t. $Y_t = 1$ with probability η_{X_t} and $Y_t = 0$ with probability $(1 - \eta_{X_t})$. Let $S_Y^t = \sum_{k=1}^t Y_k$. Notice that, for all t, $\eta_{X_t} \ge \eta$. Given a track $x_{1:t}$, the observations are independent. By Lemma 1, for any



(a) (b) (c) (d)

Fig. 8. (a) Scenario 1 (10×10 toroidal grid, trajectory of object 1 in solid line, trajectory of object 2 in dashed line, observations from object 1 in circles, observations from object 2 in diamonds). (b) Estimated tracks for scenario 1 (trajectory of object 1 in solid line, trajectory of object 2 in dashed line). (c) Scenario 2 (30×30 toroidal grid). (d) Estimated tracks for scenario 2.

t, $P(S_Y^t < d|x_{1:t}) \leq P(S_Z^t < d)$ for $d \leq t$. Furthermore, the inequality holds for any $x_{1:t}$. Hence,

$$\begin{aligned} P(S_Y^t < d) &= \sum_{x_{1:t}} P(S_Y^t < d|x_{1:t}) P(x_{1:t}) \\ &\leq \left(\sum_{x_{1:t}} P(x_{1:t}) \right) P(S_Z^t < d) \\ &= P(S_Z^t < d). \end{aligned}$$

Let $\delta_t = \sqrt{\frac{2}{\eta t} \log\left(\frac{T}{s\epsilon}\right)}$. $\delta_t < 1$ for all $t \geq t_0$ since $s > \frac{T}{\epsilon} \exp\left(-\frac{\eta t_0}{2}\right)$. Thus,

$$\begin{aligned} P(W_T) &\leq \sum_{t=1}^{T} \sum_{k=1}^{\infty} \mathbb{I}(t = ks) P(S_Y^t < \theta_t) \\ &\leq \sum_{t=1}^{T} \sum_{k=1}^{\infty} \mathbb{I}(t = ks) P(S_Z^t < \theta_t) \\ &= \sum_{k=1}^{\lfloor \frac{T}{s} \rfloor} P(S_Z^{ks} < \theta_{ks}) \\ &\leq \sum_{k=1}^{\lfloor \frac{T}{s} \rfloor} P(S_Z^{ks} < (1 - \delta_{ks}) \eta ks) \\ &< \sum_{k=1}^{\lfloor \frac{T}{s} \rfloor} \exp\left(-\frac{\eta ks \delta_{ks}^2}{2}\right) \\ &= \sum_{k=1}^{\lfloor \frac{T}{s} \rfloor} \exp\left(-\frac{\eta ks}{2} \frac{2}{\eta ks} \log\left(\frac{T}{s\epsilon}\right)\right) \\ &= \sum_{k=1}^{\lfloor \frac{T}{s} \rfloor} \epsilon \cdot \frac{s}{T} \leq \epsilon \cdot \frac{s}{T} \cdot \frac{T}{s} = \epsilon \end{aligned}$$

where $\mathbb{I}$ is an indicator function and the Chernoff bound [17] is used in the fourth inequality.

B. Proof of Theorem 2

Let Z_t be a Bernoulli process with parameter η and let $S_Z^t = \sum_{k=1}^{t} Z_k$. Let $S_Y^t = \sum_{k=1}^{t} Y_k^{X_k}$. Then, as in the proof of Theorem 1, $P(S_Y^t < d) \leq P(S_Z^t < d)$. Let $\delta = \sqrt{\frac{2}{\eta s} \log\left(\frac{1+\epsilon}{\epsilon}\right)}$. Then $\delta < 1$ by our choice of s. Thus,

$$\begin{aligned} P(W_\infty) &\leq \sum_{t=1}^{\infty} \sum_{k=1}^{\infty} \mathbb{I}(t = ks) P(S_Y^t < \theta_t) \\ &\leq \sum_{t=1}^{\infty} \sum_{k=1}^{\infty} \mathbb{I}(t = ks) P(S_Z^t < \theta_t) \\ &= \sum_{k=1}^{\infty} P(S_Z^{ks} < \theta_{ks}) \\ &\leq \sum_{k=1}^{\infty} P(S_Z^{ks} < (1 - \delta) \eta ks) \\ &< \sum_{k=1}^{\infty} \exp\left(-\frac{\eta ks \delta^2}{2}\right) \\ &= \sum_{k=1}^{\infty} \exp\left(-\frac{\eta ks}{2} \frac{2}{\eta s} \log\left(\frac{1+\epsilon}{\epsilon}\right)\right) \\ &= \sum_{k=1}^{\infty} \left(\frac{\epsilon}{1+\epsilon}\right)^k = \epsilon, \end{aligned}$$

where the Chernoff bound [17] is used in the fourth inequality.

C. Proof of Theorem 3

To avoid confusion, let us denote the joint distribution without false detections (1) by $Q(X_{1:T}, Y_{1:T})$ and the joint distribution with false detections (9) by $P(X_{1:T}, Y_{1:T})$. By the optimality, for any feasible track $x_{1:T}$, *i.e.*, $P(x_{1:T}) > 0$,

$$\begin{aligned} P(x_{1:T}^*, y_{1:T}) &\geq P(x_{1:T}, y_{1:T}) \\ Q(q_{1:T}, y_{1:T}) &\geq Q(x_{1:T}, y_{1:T}). \end{aligned}$$

In particular, we have

$$\begin{aligned} P(x_{1:T}^*, y_{1:T}) &\geq P(q_{1:T}, y_{1:T}) \quad (13) \\ Q(q_{1:T}, y_{1:T}) &\geq Q(x_{1:T}^*, y_{1:T}). \quad (14) \end{aligned}$$

From (13), we obtain the first inequality in (10). Now

consider the following joint distribution ratio

$$\begin{aligned} R &:= \frac{P(q_{1:T}, y_{1:T})}{P(x^*_{1:T}, y_{1:T})} = \frac{P(q_{1:T})}{P(x^*_{1:T})} \frac{\prod_{t=1}^T P(y_t|q_t)}{\prod_{t=1}^T P(y_t|x^*_t)} \\ &= \frac{P(q_{1:T}) \prod_{t=1}^T \left[\eta^{y_t^{q_t}} (1-\eta)^{1-y_t^{q_t}} \prod_{i \neq q_t} \xi^{y_t^i} (1-\xi)^{1-y_t^i}\right]}{P(x^*_{1:T}) \prod_{t=1}^T \left[\eta^{y_t^{x_t^*}} (1-\eta)^{1-y_t^{x_t^*}} \prod_{i \neq x_t^*} \xi^{y_t^i} (1-\xi)^{1-y_t^i}\right]} \\ &= \frac{P(q_{1:T})}{P(x^*_{1:T})} \frac{\prod_{t=1}^T \eta^{y_t^{q_t}} (1-\eta)^{1-y_t^{q_t}}}{\prod_{t=1}^T \eta^{y_t^{x_t^*}} (1-\eta)^{1-y_t^{x_t^*}}} \frac{\prod_{t=1}^T \xi^{y_t^{x_t^*}} (1-\xi)^{1-y_t^{x_t^*}}}{\prod_{t=1}^T \xi^{y_t^{q_t}} (1-\xi)^{1-y_t^{q_t}}}. \end{aligned}$$

Let $S_q = \sum_{t=1}^T y_t^{q_t}$ and $S_x = \sum_{t=1}^T y_t^{x_t^*}$. Then

$$\begin{aligned} R &= \frac{P(q_{1:T})}{P(x^*_{1:T})} \frac{\eta^{S_q}(1-\eta)^{T-S_q}}{\eta^{S_x}(1-\eta)^{T-S_x}} \frac{\xi^{S_x}(1-\xi)^{T-S_x}}{\xi^{S_q}(1-\xi)^{T-S_q}} \\ &= \frac{P(q_{1:T})}{P(x^*_{1:T})} \left(\frac{\eta}{1-\eta}\right)^{S_q - S_x} \left(\frac{\xi}{1-\xi}\right)^{S_x - S_q} \qquad (15) \\ &= \frac{Q(q_{1:T}, y_{1:T})}{Q(x^*_{1:T}, y_{1:T})} \left(\frac{\xi}{1-\xi}\right)^{S_x - S_q}. \end{aligned}$$

By (14), $\frac{Q(q_{1:T}, y_{1:T})}{Q(x^*_{1:T}, y_{1:T})} \geq 1$. But by (13), $\frac{P(q_{1:T}, y_{1:T})}{P(x^*_{1:T}, y_{1:T})} \leq 1$. Hence, $\left(\frac{\xi}{1-\xi}\right)^{S_x - S_q} \leq 1$. Since $\xi \leq .5$, $\left(\frac{\xi}{1-\xi}\right) \leq 1$ and we must have $S_x \geq S_q$. But S_q cannot be arbitrarily small. Since $\frac{Q(q_{1:T}, y_{1:T})}{Q(x^*_{1:T}, y_{1:T})} \geq 1$,

$$\begin{aligned} \frac{P(q_{1:T})}{P(x^*_{1:T})} \left(\frac{\eta}{1-\eta}\right)^{S_q - S_x} &\geq 1 \\ S_q - S_x &\geq \frac{\log\left(\frac{P(x^*_{1:T})}{P(q_{1:T})}\right)}{\log\left(\frac{\eta}{1-\eta}\right)} \\ S_x &\leq S_q + \frac{\log\left(\frac{P(q_{1:T})}{P(x^*_{1:T})}\right)}{\log\left(\frac{\eta}{1-\eta}\right)}. \end{aligned}$$

Let $b = \frac{\log(P(q_{1:T})/P(x^*_{1:T}))}{\log(\eta/(1-\eta))}$. Then, we can bound S_x as

$$S_q \leq S_x \leq S_q + b.$$

Notice that if $P(q_{1:T}) = P(x^*_{1:T})$, $S_q \geq S_x$ and we get an equality $S_q = S_x$. Hence, $\alpha = 1$ and $q_{1:T} = x^*_{1:T}$. On the other hand, we cannot have $P(q_{1:T}) < P(x^*_{1:T})$, since this requires $S_q > S_x$, contradicting the optimality of $x^*_{1:T}$. Now R takes the minimum value if $S_x = S_q + b$. So

$$\begin{aligned} R &\geq \frac{P(q_{1:T})}{P(x^*_{1:T})} \left(\frac{\eta}{1-\eta}\right)^{-b} \left(\frac{\xi}{1-\xi}\right)^{b} \\ &= \left(\frac{\xi}{1-\xi}\right)^{b} \\ &\geq \left(\frac{\xi}{1-\xi}\right)^{\frac{\log(P(q_{1:T})/m)}{\log(\eta/(1-\eta))}} \\ &= \frac{1}{\alpha}. \end{aligned}$$

Hence $P(q_{1:T}, y_{1:T})/P(x^*_{1:T}, y_{1:T}) \geq \frac{1}{\alpha}$ and we get the second inequality in (10).

REFERENCES

[1] D. Culler, D. Estrin, and M. Srivastava, "Overview of sensor networks," *IEEE Computer, Special Issue in Sensor Networks*, Aug. 2004.

[2] D. Culler and H. Mulder, "Smart sensors to network the world," *Scientific American*, June 2004.

[3] D. Estrin, D. Culler, K. Pister, and G. Sukhatme, "Connecting the physical world with pervasive networks," *IEEE Pervasive Computing*, vol. 1, no. 1, pp. 59–69, 2002.

[4] L. Schenato, S. Oh, and S. Sastry, "Swarm coordination for pursuit evasion games using sensor networks," in *Proc. of the International Conference on Robotics and Automation*, Barcelona, Spain, 2005.

[5] J. Hightower and G. Borriello, "Location systems for ubiquitous computing," *Computer*, vol. 34, no. 8, pp. 57–66, Aug. 2001.

[6] D. Li, K. Wong, Y. H. Hu, and A. Sayeed, "Detection, classification and tracking of targets," *IEEE Signal Processing Magazine*, vol. 17-29, March 2002.

[7] J. Aslam, Z. Butler, V. Crespi, G. Cybenko, and D. Rus, "Tracking a moving object with a binary sensor network," in *ACM International Conference on Embedded Networked Sensor Systems*, 2003.

[8] J. Liu, J. Liu, J. Reich, P. Cheung, and F. Zhao, "Distributed group management for track initiation and maintenance in target localization applications," in *Proc. of the 2nd workshop on Information Processing in Sensor Networks*, April 2003.

[9] J. Liu, J. Liu, M. Chu, J. Reich, and F. Zhao, "Distributed state representation for tracking problems in sensor networks," in *Proc. of the 3nd workshop on Information Processing in Sensor Networks*, April 2004.

[10] M. Coates, "Distributed particle filters for sensor networks," in *Proc. of the 3nd workshop on Information Processing in Sensor Networks*, April 2004.

[11] K. Whitehouse, F. Jiang, A. Woo, C. Karlof, and D. Culler, "Sensor field localization: a deployment and empirical analysis," Univ. of California, Berkeley, Tech. Rep. UCB//CSD-04-1349, April 9 2004.

[12] X. Nguyen, M. I. Jordan, and B. Sinopoli, "A kernel-based learning approach to ad hoc sensor network localization," in *AAAI-2004 Workshop on Sensor Networks*, July 2004.

[13] J. Aspnes, D. Goldenberg, and Y. Yang, "On the computational complexity of sensor network localization," in *Proc. of the First International Workshop on Algorithmic Aspects of Wireless Sensor Networks*, Turku, Finland, July 2004.

[14] L. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–285, 1989.

[15] S. Oh, L. Schenato, and S. Sastry, "A hierarchical multiple-target tracking algorithm for sensor networks," in *Proc. of the International Conference on Robotics and Automation*, Barcelona, Spain, April 2005.

[16] J. Shin, L. Guibas, and F. Zhao, "A distributed algorithm for managing multi-target identities in wireless ad-hoc sensor networks," in *Proc. of the 2nd workshop on Information Processing in Sensor Networks*, April 2003.

[17] R. Motwani and P. Raghavan, *Randomized Algorithms*. New York: Cambridge University Press, 1995.