Instrumenting Wireless Sensor Networks for Real-Time Surveillance

Songhwai Oh, Phoebus Chen, Michael Manzo, and Shankar Sastry

Department of Electrical Engineering and Computer Sciences
University of California, Berkeley
Berkeley, CA 94720, U.S.A.
`{sho,phoebusc,sastry}@eecs.berkeley.edu,mike@manzo.org`

Abstract— **This paper considers the problem of tracking an unknown number of targets using a wireless sensor network for surveillance. In particular, we consider the case in which each sensor reports only a binary value indicating whether an object is detected near the reporting sensor or not. Since the number of targets and initial states of targets are unknown in advance, the task of tracking with coarse measurements from binary sensors is extremely challenging. This paper develops an efficient multi-sensor fusion algorithm which converts binary detections into finer position reports using spatial correlation. The fused measurements are then used by the Markov chain Monte Carlo data association (MCMCDA) algorithm to track an unknown number of targets. The algorithm has been successfully applied in real-time to track an unknown number of human subjects moving through an outdoor field monitored by a wireless sensor network. To our knowledge, this paper presents the first large-scale demonstration of multi-target tracking using a wireless sensor network without relying on classification.**

I. INTRODUCTION

In wireless ad-hoc sensor networks, a large number of small and cheap sensor-rich devices forms an ad-hoc network to monitor changes in our environment [1,2,3]. Each device, called a sensor node, is capable of sensing, computation and communication. It is envisioned that an abundant number of spatially spread sensors will enable us to monitor changes in our environment [2]. In particular, wireless sensor networks are useful in applications that require locating and tracking moving targets. Typical examples include search-and-rescue operations, civil surveillance systems, inventory systems for moving parts in a warehouse, and search-and-capture missions in military scenarios. The problems appearing in these applications can be cast as multi-target tracking problems. However, while sensor networks can provide observability over a large area, they cannot provide consistent, high-quality measurements in a timely manner due to packet losses, communication delays, and false detections. This has been the main challenge in developing a real-time surveillance system using sensor networks.

The capability of each sensor node is limited by constraints such as a limited supply of power, manufacturing costs, and limited package sizes. Consequently, each sensor node has short communication and sensing ranges, a limited amount of

memory and limited computational power. Considering these limitations, the well known multi-target tracking algorithms such as joint probabilistic data association filter (JPDAF) [4] and multiple hypothesis tracker (MHT) [5, 6] are not feasible for sensor networks due to their time and space complexities. As a result, many new tracking algorithms have been developed recently.

Most of the algorithms developed for sensor networks are designed for single-target tracking [7, 8, 9, 10, 11] and some of these algorithms are applied to track multiple targets using classification [7, 10] or heuristics, such as the nearest-neighbor filter (NNF) [11]. A few algorithms are designed for multi-target tracking [12, 13] but the complexity of the data association problem inherent to multi-target tracking is avoided by classification. When tracking targets of a similar type or when reliable classification information is not available, the classification-based tracking algorithm behaves as the NNF. Considering the fact that the complexity of the data association problem is NP-hard [14], a heuristic approach breaks down under difficult circumstances.

In this paper, we develop algorithms for a real-time surveillance system using a sensor network of binary sensors and describe the demonstration of the system. Each sensor reports only a binary value indicating whether an object is detected near the reporting sensor or not. We develop an autonomous tracking system which can initiate and terminate tracks of multiple targets based on Markov chain Monte Carlo data association (MCMCDA) [15]. MCMCDA can track an unknown number of targets in real-time and is robust against false detections and missing detections [15], making it ideal for our application. But since the binary detection measurements provide coarse measurements, it is difficult to use them directly to initiate, maintain, disambiguate, and terminate tracks of multiple targets. Taking advantage of spatial correlation, we develop an efficient multi-sensor fusion algorithm which converts binary detections into finer position reports. Since we do not know the number of targets in advance, the fusion algorithm can provide incorrect and inconsistent position reports. However, the inconsistency in position reports are later fixed by MCMCDA using spatio-

temporal correlation.

Only a handful of tracking algorithms have been demonstrated using a physical sensor network deployment. Most often, the algorithms are used to track a single target [11,9,8] while multiple targets are tracked using classification, *e.g.*, [10]. To our knowledge, this paper presents the first large-scale demonstration of multi-target tracking using sensor networks without relying on classification.

The remainder of this paper is structured as follows. The multi-target tracking problem and its probabilistic model are described in Section II. A multi-sensor fusion algorithm is described in Section III. The MCMCDA algorithm for multi-target tracking is presented in Section IV. The results from the experiments are shown in Section V.

II. MULTI-TARGET TRACKING

Since the number of targets is unknown and time-varying, we need a general formulation of the multi-target tracking problem. Let $T \in \mathbb{Z}^+$ be the duration of surveillance. Let K be the number of targets that appear in the surveillance region $\mathcal{R}$ during the surveillance period. Each target k moves in $\mathcal{R}$ for some duration $[t_i^k, t_f^k] \subset [1, T]$. Notice that the exact values of K and $\{t_i^k, t_f^k\}$ are unknown. Each target arises at a random position in $\mathcal{R}$ at t_i^k, moves independently around $\mathcal{R}$ until t_f^k and disappears. At each time, an existing target persists with probability $1 - p_z$ and disappears with probability p_z. The number of targets arising at each time over $\mathcal{R}$ has a Poisson distribution with a parameter $\lambda_b V$ where λ_b is the birth rate of new targets per unit time, per unit volume, and V is the volume of $\mathcal{R}$. The initial position of a new target is uniformly distributed over $\mathcal{R}$.

Let $F^k : \mathbb{R}^{n_x} \to \mathbb{R}^{n_x}$ be the discrete-time dynamics of the target k, where n_x is the dimension of the state variable, and let $x^k(t) \in \mathbb{R}^{n_x}$ be the state of the target k at time t for $t = 1, \ldots, T$. The target k moves according to

$$x^k(t+1) = F^k(x^k(t)) + w^k(t), \text{ for } t = t_i^k, \ldots, t_f^k - 1, \quad (1)$$

where $w^k(t) \in \mathbb{R}^{n_x}$ are white noise processes. The white noise process is included to model non-rectilinear motions of targets. When a target is present, a noisy observation (or measurement[1]) of the state of the target is measured with a detection probability p_d. There are also false alarms and the number of false alarms has a Poisson distribution with a parameter $\lambda_f V$, where λ_f is the false alarm rate per unit time, per unit volume. Let $n(t)$ be the number of observations at time t, including both noisy observations and false alarms. Let $y^j(t) \in \mathbb{R}^{n_y}$ be the j-th observation at time t for $j = 1, \ldots, n(t)$, where n_y is the dimension of each observation vector. Each target generates a unique observation at each sampling time if it is detected. Let $H^j : \mathbb{R}^{n_x} \to \mathbb{R}^{n_y}$ be the

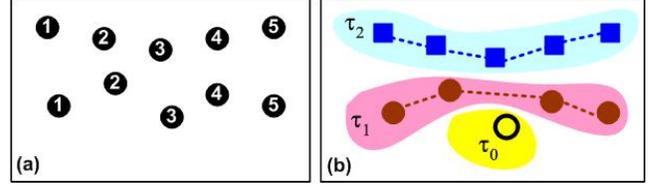[1]Note that the terms *observation* and *measurement* are used interchangeably in this paper.

Fig. 1. (a) An example of observations Y (each circle represents an observation and numbers represent observation times). (b) An example of a partition ω of Y

observation model. Then the observations are generated as follows:

$$y^j(t) = \begin{cases} H^j(x^k(t)) + v^j(t) & \text{if } y^j(t) \text{ is from } x^k(t) \\ u_f(t) & \text{otherwise,} \end{cases}$$

$$(2)$$

where $v^j(t) \in \mathbb{R}^{n_y}$ are white noise processes and $u_f(t) \sim \text{Unif}(\mathcal{R})$ is a random process for false alarms. We assume that the targets are indistinguishable in this paper, but if observations include target type or attribute information, the state variable can be extended to include target type information.

The main objective of the multi-target tracking problem is to estimate K, $\{t_i^k, t_f^k\}$ and $\{x^k(t) : t_i^k \leq t \leq t_f^k\}$, for $k = 1, \ldots, K$, from noisy observations.

Let $Y(t) = \{y^j(t) : j = 1, \ldots, n(t)\}$ be all measurements at time t and $Y = \{Y(t) : 1 \leq t \leq T\}$ be all measurements from $t = 1$ to $t = T$. Let Ω be a collection of partitions of Y such that, for $\omega \in \Omega$,

1) $\omega = \{\tau_0, \tau_1, \ldots, \tau_K\}$;
2) $\bigcup_{k=0}^{K} \tau_k = Y$ and $\tau_i \cap \tau_j = \emptyset$ for $i \neq j$;
3) τ_0 is a set of false alarms;
4) $|\tau_k \cap Y(t)| \leq 1$ for $k = 1, \ldots, K$, $t = 1, \ldots, T$; and
5) $|\tau_k| \geq 2$ for $k = 1, \ldots, K$.

An example of a partition is shown in Figure 1 and ω is also known as a *joint association event* in literature. Here, K is the number of tracks for the given partition $\omega \in \Omega$ and $|\tau_k|$ denotes the cardinality of the set τ_k. We call τ_k a track when there is no confusion although the actual track is the set of estimated states from the observations τ_k. Note that a track is assumed to contain at least two observations since we cannot distinguish a track with a single observation from a false alarm, assuming $\lambda_f > 0$. Notice that the definition of Ω can be adjusted for special cases, in which $p_d = 1$ or $\lambda_f = 0$.

Let $e(t - 1)$ be the number of targets at time $t - 1$, $z(t)$ be the number of targets terminated at time t and $c(t) = e(t - 1) - z(t)$ be the number of targets from time $t - 1$ that have not terminated at time t. Let $a(t)$ be the number of new targets at time t, $d(t)$ be the number of actual target detections at time t and $g(t) = c(t) + a(t) - d(t)$ be the number of undetected targets. Finally, let $f(t) = n(t) - d(t)$ be the number of false alarms. Using the Bayes rule, it can

be shown that the posterior of ω is:

$$P(\omega|Y) \propto P(Y|\omega) \prod_{t=1}^{T} p_z^{z(t)} (1-p_z)^{c(t)} p_d^{d(t)} (1-p_d)^{g(t)} \lambda_b^{a(t)} \lambda_f^{f(t)} \tag{3}$$

where $P(Y|\omega)$ is the likelihood of observations Y given ω, which can be computed based on the chosen dynamic and measurement models, *e.g.*, [15].

There are two major approaches to solve the multi-target tracking problem [16]: *maximum a posteriori* (MAP) and Bayesian approaches. The MAP approach finds a partition of observations such that $P(\omega|Y)$ is maximized and estimates the states of the targets based on this partition. A Bayesian approach, called *minimum mean square error* (MMSE), finds estimates which minimizes the expected square error. For instance, $\mathbb{E}(x^k(t)|Y)$ is the MMSE estimate for the state $x^k(t)$ of target k. However, when the number of targets is not fixed, a unique labeling of each target is required to find $\mathbb{E}(x^k(t)|Y)$ under the MMSE approach. In this paper, we take the MAP approach to the multi-target tracking problem for its convenience.

III. MULTI-SENSOR FUSION ALGORITHM

In order to obtain finer position reports from binary detections, we use spatial correlation among detections from neighboring sensors. The idea behind the fusion algorithm is to compute the likelihood of a target given detections assuming there is a single target. This is only an approximation since there can be more than one target. However, any inconsistencies caused by this approximation are fixed by the MCMCDA algorithm using spatio-temporal correlation.

For each sensor i, let R_i be the sensing region of i. R_i can be an arbitrary shape but we assume that it is known to the system in advance. Let $y_i \in \{0, 1\}$ be the detection made by sensor i, such that sensor i reports $y_i = 1$ if it detects a moving object in R_i, and $y_i = 0$ otherwise. Let p_i be the detection probability and q_i be the false detection probability of sensor i. Let x be the position of an object. For the purpose of illustration, suppose that there are two sensors, sensor 1 and sensor 2, and $R_1 \cap R_2 \neq \emptyset$ (see Figure 2 (left)). The overall sensing region $R_1 \cup R_2$ can be partitioned into a set of non-overlapping cells (or blocks) as shown in Figure 2 (right). The likelihoods can be computed as follows:

$$\begin{aligned} P(y_1, y_2 | x \in S_1) &= p_1^{y_1} (1-p_1)^{1-y_1} q_2^{y_2} (1-q_2)^{1-y_2} \\ P(y_1, y_2 | x \in S_2) &= q_1^{y_1} (1-q_1)^{1-y_1} p_2^{y_2} (1-p_2)^{1-y_2} \\ P(y_1, y_2 | x \in S_3) &= p_1^{y_1} (1-p_1)^{1-y_1} p_2^{y_2} (1-p_2)^{1-y_2}, \end{aligned} \tag{4}$$

where $S_1 = R_1 \setminus R_2$, $S_2 = R_2 \setminus R_1$ and $S_3 = R_1 \cap R_2$ (see Figure 2 (right)). Hence, for any deployment we can first partition the surveillance region into a set of non-overlapping cells. Then, given detection data, we can compute the likelihood of each cell as shown in the previous example.

An example of detections of two targets by a 10×10 sensor grid is shown in Figure 3 (left). In this example,

Fig. 2. (left) Sensing regions of two sensors 1 and 2. R_i is the sensing region of sensor i. (right) A partition of the overall sensing region $R_1 \cup R_2$ into non-overlapping cells S_1, S_2 and S_3, where $S_1 = R_1 \setminus R_2$, $S_2 = R_2 \setminus R_1$ and $S_3 = R_1 \cap R_2$.

the sensing region is assumed to be a disk with radius of 7.62m (10 ft). We have assumed $p_i = 0.7$ and $q_i = 0.05$ for all i. From the detections shown in Figure 3 (left), its likelihood can be computed using equations similar to (4) for each non-overlapping cell (see Figure 3 (middle)). Notice that it is a time-consuming task to find all non-overlapping cells for arbitrary sensing region shapes and sensor deployments. Hence, we quantized the surveillance region and the likelihoods are computed for a finite number of points as shown in Figure 3 (middle).

There are two parts in this likelihood computation: the detection part (terms involving p_i) and the false detection part (terms involving q_i). Hereafter, we call the detection part of the likelihood as the *detection-likelihood* and the false detection part of the likelihood as the *false-detection-likelihood*. Notice that the computation of the false-detection-likelihood requires measurements from all sensors. However, for a large wireless sensor network, it is not feasible to exchange detection data with all other sensors. Instead, we use a threshold test to avoid computing the false-detection-likelihood and distribute the likelihood computation. The detection-likelihood of a cell is computed if there are at least n_d detections, where n_d is a user-defined threshold. Using $n_d = 3$, the detection-likelihood of the detections from Figure 3 (left) can be computed as shown in Figure 3 (right). The computation of the detection-likelihood can be done in a distributed manner. Assign a set of non-overlapping cells to each sensor such that no two sensors share the same cell and each cell is assigned to a sensor whose sensing region includes the cell. For each sensor i, let $\{S_{i_1}, \ldots, S_{i_{m(i)}}\}$ be a set of non-overlapping cells, where $m(i)$ is the number of cells assigned to sensor i. Then, if sensor i reports a detection, it computes the likelihoods of each cells in $\{S_{i_1}, \ldots, S_{i_{m(i)}}\}$ based on its own measurements and the measurements from neighboring sensors. A neighboring sensor is a sensor whose sensing region intersects the sensing region of sensor i.

Based on the detection-likelihoods, we compute target position reports by clustering. Let $\mathcal{S} = \{S_1, \ldots, S_m\}$ be a set of cells whose detection-likelihoods are computed, *i.e.*, the number of detections for each S_i is at least n_d. First, randomly pick S_j from $\mathcal{S}$ and remove S_j from $\mathcal{S}$. Then cluster around S_j the remaining cells in $\mathcal{S}$ whose set-distance to S_j is less than the sensing radius. The cells clustered with S_j are then removed from $\mathcal{S}$. Now repeat the procedure until
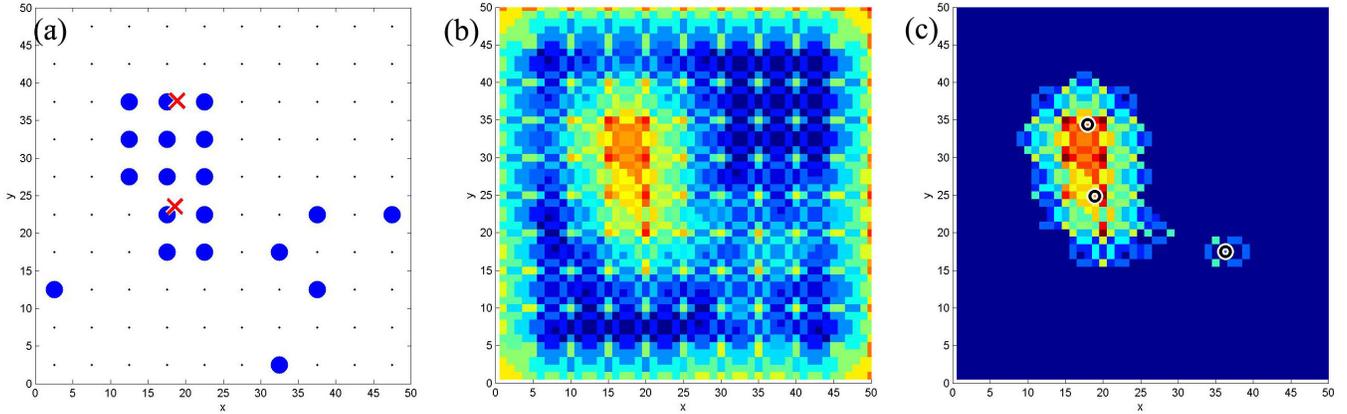
Fig. 3. (a) Detections of two targets by a 10×10 sensor grid (targets in (red) $\times$, detections in (blue) disks, and sensor positions in small dots). (b) Likelihood of detections. (c) Detection-likelihood of detections with threshold $n_{\mathrm{d}} = 3$. Estimated positions of targets are shown in (black) circles. (Figures are best viewed in color.)

$\mathcal{S}$ is empty. Let $\{C_k : 1 \leq k \leq K_{\mathrm{cl}}\}$ be the clusters formed by this procedure, where K_{cl} is the total number of clusters. For each cluster C_k, its center of mass is computed to obtain a fused position report, *i.e.*, an estimated position of a target. An example of position reports is shown in Figure 3 (right).

IV. Markov Chain Monte Carlo Data Association

Markov chain Monte Carlo (MCMC) plays a significant role in many fields such as physics, statistics, economics, and engineering [17]. In some cases, MCMC is the only known general algorithm that finds a good approximate solution to a complex problem in polynomial time [18]. MCMC techniques have been applied to complex probability distribution integration problems, counting problems such as #P-complete problems, and combinatorial optimization problems [18, 17].

MCMC is a general method to generate samples from a distribution π on a space Ω by constructing a Markov chain $\mathcal{M}$ with states $\omega \in \Omega$ and stationary distribution $\pi(\omega)$. We now describe an MCMC algorithm known as the Metropolis-Hastings algorithm [19]. If we are at state $\omega \in \Omega$, we propose $\omega' \in \Omega$ following the proposal distribution $q(\omega, \omega')$. The move is accepted with an acceptance probability $A(\omega, \omega')$ where

$$A(\omega, \omega') = \min\left(1, \frac{\pi(\omega')q(\omega', \omega)}{\pi(\omega)q(\omega, \omega')}\right), \qquad (5)$$

otherwise the sampler stays at ω, so that the detailed balance is satisfied. If we make sure that $\mathcal{M}$ is irreducible and aperiodic, then $\mathcal{M}$ converges to its stationary distribution by the ergodic theorem [20].

The MCMC data association (MCMCDA) algorithm is described in Algorithm 1. MCMCDA is an MCMC algorithm whose state space is Ω, as described in Section II, and whose stationary distribution is the posterior (3). The acceptance probability $A(\omega, \omega')$ is defined in (5) where $\pi(\omega) = P(\omega|Y)$ from (3). The proposal distribution for MCMCDA consists

---

Algorithm 1 MCMCDA (MAP)

Input: $Y, n_{\mathrm{mc}}, \omega_{\mathrm{init}}$
Output: $\hat{\omega}$
1: $\omega = \omega_{\mathrm{init}}$; $\hat{\omega} = \omega_{\mathrm{init}}$;
2: **for** $n = 1$ to n_{mc} **do**
3:     propose ω' based on ω (see Figure 4)
4:     sample U from Unif$[0, 1]$
5:     $\omega = \omega'$ if $U < A(\omega, \omega')$
6:     $\hat{\omega} = \omega$ if $p(\omega|Y)/p(\hat{\omega}|Y) > 1$
7: **end for**

---

of five types of moves (a total of eight moves). They are (1) a birth/death move pair; (2) a split/merge move pair; (3) an extension/reduction move pair; (4) a track update move; and (5) a track switch move. The MCMCDA moves are illustrated in Figure 4. For a detailed description of each move, see [15]. The inputs for MCMCDA are the set of all observations Y, the number of samples n_{mc}, the initial state ω_{init}. The output $\hat{\omega}$ approximates the MAP estimate $\arg\max P(\omega|Y)$. For given $\hat{\omega}$, the states of the targets can be easily computed by running any filtering algorithm since the associations between the targets and the measurements are now completely known. In the experiment, the online version of MCMCDA [15] is used.

It has been shown that MCMCDA is an *optimal Bayesian filter* in the limit, *i.e.*, given a bounded function $X : \Omega \rightarrow \mathbb{R}^n$, $\hat{X} \rightarrow \mathbb{E}_\pi X$ as $n_{\mathrm{mc}} \rightarrow \infty$ [16]. In addition, in terms of time and memory, MCMCDA is more computationally efficient than MHT and outperforms MHT with heuristics (*i.e.*, pruning, gating, clustering, N-scan-back logic and k-best hypotheses) under extreme conditions, such as a large number of targets in a dense environment, low detection probabilities, and high false alarm rates [15]. In addition, the algorithm is robust against transmission failures, communication delays and sensor localization error. In simulation,
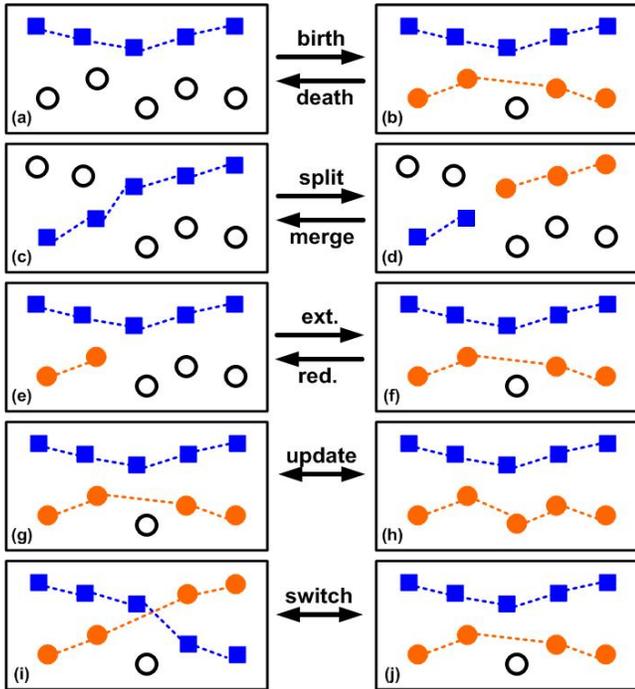
Fig. 4. Graphical illustration of MCMCDA moves (associations are indicated by dotted lines and hollow circles are false alarms). Each move proposes a new joint association event ω' which is a modification of the current joint association event ω. The birth move proposes ω' by forming a new track from the set of false alarms ($(a) \rightarrow (b)$). The death move proposes ω' by combining one of the existing tracks into the set of false alarms ($(b) \rightarrow (a)$). The split move splits a track from ω into two tracks ($(c) \rightarrow (d)$) while the merge move combines two tracks in ω into a single track ($(d) \rightarrow (c)$). The extension move extends an existing track in ω ($(e) \rightarrow (f)$) and the reduction move reduces an existing track in ω ($(f) \rightarrow (e)$). The track update move chooses a track in ω and assigns different measurements from the set of false alarms ($(g) \leftrightarrow (h)$). The track switch move chooses two track from ω and switches some measurement-to-track associations ($(i) \leftrightarrow (j)$). A move is chosen randomly from the distribution $\xi_K(m)$, where K is the number of tracks of the current partition ω and m is used to index a move, such that $m = 1$ for a birth move, $m = 2$ for a death move and so on. For more detail, see [15].

there is no performance loss up to an average localization error of 0.7 times the separation between sensors, and the algorithm tolerates up to 50% lost-to-total packet ratio and 90% delayed-to-total packet ratio [21].

V. EXPERIMENTS

Multi-target tracking was demonstrated at the Defense Advanced Research Projects Agency (DARPA) Network Embedded Systems Technology (NEST) final experiment on August 30, 2005. The experiment was performed using an outdoor sensor network testbed consisting of 557 Trio motes [22] (see Figure 5). 144 of these nodes were used for the tracking experiment. However, six out of the 144 nodes used in the experiment were not functioning on the day of the demo. The 144 nodes used for the tracking experiment were deployed at approximately 5 meter spacing in a 12×12 grid.

Fig. 5. Hardware for the sensor nodes. (left) Trio sensor node on a tripod. On top is the microphone, buzzer, solar panel, and user and reset buttons. On the sides are the windows for the passive infrared sensors. (right) A live picture from the experiment.

Each sensor node includes a microphone, x-y axis magnetometers, and four passive infrared (PIR) motion sensors. But we found that the PIR sensors were the most effective for sensing human subjects moving through the sensor field. The magnetometer sensor had limited range and the acoustic sensor required complex signal processing to separate the acoustic signatures of a moving target from background noise. The PIR sensors provided an effective range of approximately 8 meters. The variability in the signal strength of the PIR sensor reading prohibited extraction of ranging information from the sensor, so the PIR sensors were used as binary detectors.

Tracking was demonstrated on one, two, and three human targets, with targets entering the field at different times. The tracking algorithm used in the experiment is a combination of the multi-sensor fusion algorithm described in Section III and the online version of the MCMCDA algorithm [15]. In all three experiments, the tracking algorithm correctly estimated the number of targets and produced good tracks of targets. Furthermore, the algorithm was robust to crossing tracks and correctly disambiguated crossing targets in the two and three target experiments without generating a spurious track. The algorithm does this without classification labels on the targets. Instead, the algorithm uses the dynamic models of the targets and the past measurements to help determine their trajectories.

Figure 6 shows the multi-target tracking results with three people walking through the field. The three people entered and exited the field around time 10 and 80, respectively. During the experiment, the algorithm correctly rejected false alarms and compensated for missing detections. There were many false alarms during the span of the experiments, as can be seen in Figure 7, before time 10 and after time 80. Also, though not shown in the figures, the algorithm dynamically corrected previous track hypotheses as it received more sensor readings. Figure 7 also gives a sense of the burstiness of the traffic. The spike in traffic shortly after time 50 was approximately when two of the targets crossed. It
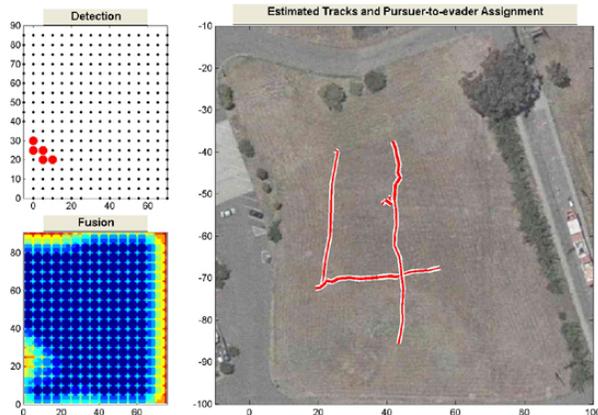
Fig. 6. Estimated tracks of targets from the experiment with three people walking in the field. (upper left) Detection panel. Sensors are marked by small dots and detections are shown in large disks. (lower left) Fusion panel shows the fused likelihood. (right) Estimated Tracks and Pursuer-to-evader Assignment panel shows the tracks estimated by the algorithm. (This figure is best viewed in color.)
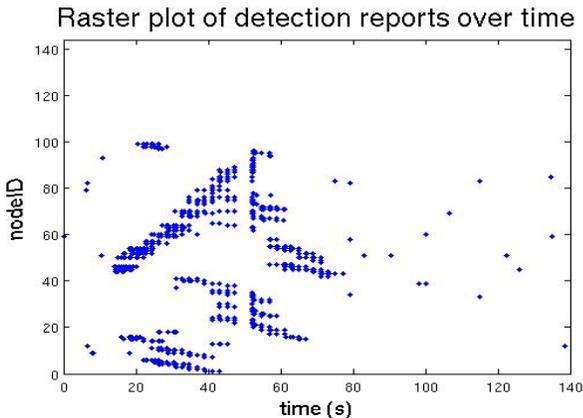

Fig. 7. Raster plot of the binary detection reports from the three target tracking demo. Dots represent detections from nodes received at the base station (node IDs on the y-axis).

shows that the proposed algorithm is robust against missing measurements, false measurements and the burstiness in the network. For detail information about the experiment setup and additional experimental results, we refer readers to our companion paper [23].

VI. CONCLUSION

In this paper, we have considered a problem of tracking an unknown number of targets using a wireless sensor network with binary sensors for surveillance and presented a tracking algorithm which combines the multi-sensor fusion algorithm and the MCMCDA algorithm. The multi-sensor fusion algorithm converts binary detections into finer approximate position reports using spatial correlation. However, the inconsistency in these position reports are later fixed by the MCMCDA algorithm using spatio-temporal correlation. The combined algorithm has been successfully applied in real-time to track an unknown number of human subjects moving through an outdoor field monitored by a wireless sensor network. Although, each sensor node provides noisy and inconsistent measurements, this paper has shown that the proper use of spatio-temporal correlation allows us to find accurate and robust estimates of the states of a complex dynamical system using wireless sensor networks.

REFERENCES

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

[2] I. Akyidliz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Communications Magazine*, vol. 40, no. 8, pp. 102–114, August 2002.

[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] Y. Bar-Shalom and T. Fortmann, *Tracking and Data Association*. San Diego, CA: Academic Press, 1988.

[5] D. Reid, "An algorithm for tracking multiple targets," *IEEE Trans. Automatic Control*, vol. 24, no. 6, pp. 843–854, December 1979.

[6] C. Chong, S. Mori, and K. Chang, "Distributed multitarget multisensor tracking," in *Multitarget-Multisensor Tracking: Advanced Applications*, Y. Bar-Shalom, Ed. Artech House: Norwood, MA, 1990, pp. 247–295.

[7] 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.

[8] J. Liu, J. Reich, and F. Zhao, "Collaborative in-network processing for target tracking," *J. of Applied Signal Processing*, April 2003.

[9] 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.

[10] A. Arora, P. Dutta, S. Bapat, V. Kulathumani, H. Zhang, V. Naik, V. Mittal, H. Cao, M. Demirbas, M. Gouda, Y. Choi, T. Herman, S. Kulkarni, U. Arumugam, M. Nesterenko, A. Vora, , and M. Miyashita, "A line in the sand: A wireless sensor network for target detection, classification, and tracking," *Computer Networks*, vol. 46, no. 5, pp. 605–634, Dec. 2004.

[11] R. R. Brooks, D. Friedlander, J. Koch, and S. Phoha, "Tracking multiple targets with self-organizing distributed ground sensors," *J. Parallel Distrib. Comput.*, vol. 64, pp. 874–884, 2004.

[12] 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.

[13] 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.

[14] A. Poore, "Multidimensional assignment and multitarget tracking," *Partitioning Data Sets. DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, vol. 19, pp. 169–196, 1995.

[15] S. Oh, S. Russell, and S. Sastry, "Markov chain Monte Carlo data association for general multiple-target tracking problems," in *Proc. of the 43rd IEEE Conference on Decision and Control*, Paradise Island, Bahamas, Dec. 2004.

[16] ——, "Markov chain Monte Carlo data association for multiple-target tracking," Univ. of California, Berkeley, Tech. Rep. UCB//ERL M05/19, 2005.

[17] I. Beichl and F. Sullivan, "The Metropolis algorithm," *Computing in Science and Engineering*, vol. 2, no. 1, pp. 65–69, 2000.

[18] M. Jerrum and A. Sinclair, "The Markov chain Monte Carlo method: An approach to approximate counting and integration," in *Approximations for NP-hard Problems*, D. Hochbaum, Ed. PWS Publishing, Boston, MA, 1996.

[19] W. Gilks, S. Richardson, and D. Spiegelhalter, Eds., *Markov Chain Monte Carlo in Practice*, ser. Interdisciplinary Statistics Series. Chapman and Hall, 1996.

[20] G. Roberts, "Markov chain concepts related to sampling algorithms," in *Markov Chain Monte Carlo in Practice*, ser. Interdisciplinary Statistics Series, W. Gilks, S. Richardson, and D. Spiegelhalter, Eds. Chapman and Hall, 1996.

[21] S. Oh, L. Schenato, P. Chen, and S. Sastry, "A scalable real-time multiple-target tracking algorithm for sensor networks," Univ. of California, Berkeley, Tech. Rep. UCB//ERL M05/9, 2005.

[22] P. Dutta, J. Hui, J. Jeong, S. Kim, C. Sharp, J. Taneja, G. Tolle, K. Whitehouse, and D. Culler, "Trio: Enabling sustainable and scalable outdoor wireless sensor network deployments," in *Proc. of the International Conference on Information Processing in Sensor Networks: Special track on Platform Tools and Design Methods for Network Embedded Sensors*, 2006.

[23] P. Chen, S. Oh, M. Manzo, B. Sinopoli, C. Sharp, K. Whitehouse, G. Tolle, J. Jeong, P. Dutta, J. Hui, S. Schaffert, S. Kim, J. Taneja, B. Zhu, T. Roosta, M. Howard, D. Culler, and S. Sastry, "Experiments in instrumenting wireless sensor networks for real-time surveillance," in *Proc. of the International Conference on Robotics and Automation (video)*, Orlando, FL, May 2006.