Occam's Razor Applied to Network Topology Inference

Dimitri Marinakis and Gregory Dudek

***Abstract*—We present a method for inferring the topology of a sensor network given nondiscriminating observations of activity in the monitored region. This is accomplished based on no prior knowledge of the relative locations of the sensors and weak assumptions regarding environmental conditions. Our approach employs a two-level reasoning system made up of a stochastic expectation maximization algorithm and a higher level search strategy employing the principle of Occam's Razor to look for the simplest solution explaining the data. The result of the algorithm is a Markov model describing the behavior of agents in the system and the underlying traffic patterns. Numerical simulations and experimental assessment conducted on a real sensor network suggest that the technique could have promising real-world applications in the area of sensor network self-configuration.**

***Index Terms*—Expectation maximization (EM), learning systems, Markov processes, multisensor systems, Occam's Razor, self-configuring systems, sensor networks, topology.**

I. Introduction

IN THIS PAPER, we address the self-calibration problem of inferring the *topology*, or internode connectivity, of a sensor network given nondiscriminating observations of activity in the environment. We are interested in recovering a representation of the network that identifies physical intersensor connectivity from the point of view of an agent navigating the environment (Fig. 1). This topological information differs from a metric representation that identifies the relative locations of the sensors but does not provide information about the layout of the region, or obstructing objects within it. We assume that we have no prior knowledge of the relative locations of the sensors and that we have only a limited knowledge of the type of activity present in the environment. We must use observational data returned from our sensors to understand the motion of agents present in the environment. By inferring underlying patterns in their motions we can then recover the relationships between the sensors of our network.

Our approach employs a two-level reasoning system. The first level is made up of our fundamental topology inference algorithm that takes the sensor observations and environmental assumptions as inputs and returns the network parameters. This algorithm is formulated using Monte Carlo expectation maximization (MCEM), but it depends on fixed values for certain numerical parameters that represent *a priori* knowledge regarding traffic patterns in the environment. The second level searches over the input parameter space of the first-level algorithm to find a global solution that optimizes a more abstract objective function based on the principle of Occam's Razor.[1]

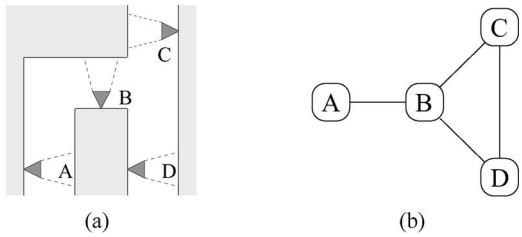

Fig. 1. (a) Example of a sensor network layout and (b) corresponding topology.

The final output of the two-level approach is a probabilistic model of the sensor network connectivity graph and the underlying traffic trends. It is worth noting that the technique recovers a much more complete description of network connectivity than just a topological map of the environment. We learn information about the number of agents in the system, internode delay distributions, internode transition likelihoods, and additional statistics regarding motion activity.

As sensor networks are established in more locations for monitoring and surveillance purposes, there will be a demand for algorithms and software approaches that can make inferences about the environment based on large quantities of highly distributed and possibly low-quality sensing information. This is especially true in areas where we are unable to venture ourselves, or unwilling to venture for fear of influencing the data we are collecting. On Great Duck Island, Maine, for example, a sensor network was successfully employed to collect habitat data without disturbing wildlife with human presence [1], [2]. Another example is the proposed underwater observing system NEPTUNE [3], which plans to wire the Juan de Fuca tectonic plate off the coast of the North-West Pacific ocean. The underwater network will generate observational data from a variety of distributed sensors that could be used to infer additional information about the ocean environment which would be difficult to collect directly for logistical and financial reasons.

This paper addresses a single aspect of the more general problem of inferring information about the environment given distributed sensor data: recovering connectivity parameters. Monitoring projects that log data for offline analysis should be able to benefit from our technique. For example, a vehicle monitoring network distributed about a city could help make decisions about

Manuscript received November 3, 2006; revised June 22, 2007. This paper was recommended for publication by Associate Editor D. Fox and Editor L. Parker upon evaluation of the reviewers' comments.

The authors are with the Centre for Intelligent Machines, McGill University, Montreal QC H3A2B4, Canada (e-mail: dmarinak@cim.mcgill.ca; dudek@cim.mcgill.ca.)

Digital Object Identifier 10.1109/TRO.2008.918098

[1]Occam's Razor is the principle enunciated by William of Occam that the simplest explanation is the best.

road improvements that might best alleviate congestion. Another motivation are applications using the connectivity information inferred by our technique for sensor network self-calibration efforts; e.g., the calibration of a surveillance system. With this paper, we are addressing a type of problem that will grow in importance as distributed sensing becomes more prevalent. We believe that techniques for solving questions such as the one we address in this paper are of practical interest and are worth investigating.

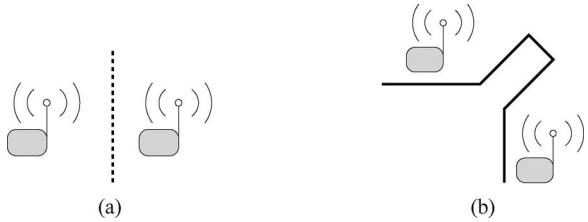

Fig. 2. Examples where communication signal strength is misleading. (a) Thin interior wall prevents passage but signal is strong. (b) Blocking exterior wall prevents signal but nodes are topologically adjacent.

II. Background

It is recognized that self-calibration and other more general self-configuration algorithms are important issues for both multirobot systems and for sensor networks [4], [5]. The main point is that a network must operate autonomously in a dynamic environment. It should be capable of reorganizing itself to handle network changes such as individual node failures or changes in communication range.

A key requirement for many network applications is the ability to self-localize [6]; i.e., recover the relative metric positions of the individual sensors in the network. This is especially important where global positioning system (GPS) solutions are too expensive, not available, or otherwise impractical [7], [8]. In general, localization efforts are based on methods for estimating the distances between sensors. Common techniques include the use of received communication signal strength in radio networks [9], or time-of-arrival ranging using ultrasound [10], [11]. This positional information can also be calculated explicitly through the use of a mobile robot [12].

The problem of inferring the topology of a sensor network is closely related to that of metric self-localization. In self-localization, the goal is to recover the relative locations of the nodes independent of the layout of the space in which the network is embedded. Topology inference as we define it, however, must take into account the spatial constraints of the environment since they determine the internode connectivity parameters. Although the topological mapping problem has been well explored in mobile robotics [13]–[16], most sensor-network-related investigations have been more recent [17], [18]; the outcome is generally a graph where vertices represent embedded sensors in the region and edges indicate navigability.

By combining a topological description of the environment with metric data obtained from self-localization algorithms, further information regarding obstructions and motion corridors could be inferred. For example, two spatially proximal nodes that were not topologically adjacent would suggest a barrier of some sort. Additionally, the topological and metric data could complement each other. Information regarding the spatial locations of the nodes as well as their communication connectivity can make it easier to determine topologically adjacent nodes and vice versa. However, in many cases, the information can be misleading (Fig. 2). Spatial proximity does not necessarily imply a topological connection, and likewise, two nodes that are topologically adjacent do not have to be close to each other.

In this paper, we attempt to solve for the topology of a network, accounting for spatial constraints, without relying on traditional self-localization techniques such as time-to-arrival and signal strength. However, along with agent signatures, these methods could be incorporated into our approach as part of the probabilistic framework.

While much of the research conducted on sensor networks is based on developing distributed and efficient algorithms appropriate for networks of low-powered sensor platforms, recently there has been a shift toward more complex approaches incorporating advanced probabilistic techniques and graphical models [19], [20]. The traditional sensor network assumption of homogenous systems of impoverished nodes is making way for tiered architectures that incorporate network components of some computational sophistication [21]. This lack of emphasis on the traditional sensor network concerns of efficiency and distributed processing is especially true in networks of vision-based sensors.

In the domain of vision-based networks, there are a number of examples of recent work that have looked at self-calibration though the exploitation of motion in the environment [17], [22]–[27]. These efforts generally focus on research issues regarding the processing of observations collected from distributed sensors. Of particular relevance is the work of Ellis, Makris, and Black [17], [27] on the topology inference of camera networks.

Ellis, Makris, and Black [17], [27] presented a technique for topology recovery based on event detection only. In their approach, they first identified entrance and exit points in camera fields of view and then attempted to find correspondences between these entrances and exits based on video data. Their technique relies on exploiting temporal correlation in observations of agent movements. The method employs a threshold-based technique that looks for peaks in the temporal distribution of travel times between entrance–exit pairs; a clear peak suggesting that a correspondence exists. The technique gave promising results on experiments carried out on a six-camera network. Although it requires a large number of observations, the method does not rely on object correlation across specific cameras. Thus, the approach can be used to efficiently produce an approximate network connectivity graph; however, when the network dynamics are complex or the traffic distribution exhibits substantial variation, it would appear the technique will have difficulty.

Unlike the threshold-based approach of Ellis, Makris, and Black [17], [27], our method for topology inference is based on constructing plausible trajectories of motion sources in the environment. This approach is closely related to multitarget

tracking, which is a well-established research area in sensor networks [28]–[30] and multirobot systems [31]. One of the key difficulties faced is that of maintaining target identities during periods when two or more targets move close together or are unobserved for a period of time. Probabilistic techniques, such as identity mass flow [32], have been devised to handle this situation. Other work poses the target identity problem as a data association problem [33]–[35].

Pasula *et al*. [36] successfully approached a traffic monitoring problem from the data association perspective through a stochastic sampling technique, although only in very simple networks. Given known sensor positions and topology, the goal of the work was to track multiple objects passing through the network and recover their long-range origin/destination information. An iterative EM algorithm was employed that assigned probable trajectories to each vehicle. These samples were then used to update model parameters such as link-travel time and vehicle characteristics. Our method of generating trajectory samples shares some techniques employed by Pasula *et al*. [36]; however, our implementation differs due to the specifics of the problem.

Another example of an approach with some similarity to ours is the paper by Dellaert *et al*. [37] in the domain of finding structure from motion. These authors also employ EM and Markov chain Monte Carlo (MCMC) methods to solve a difficult data association problem.

Another related problem domain is the simultaneous localization and mapping (SLAM) problem in mobile robotics. Recently, hybrid robot/sensor network systems have been employed to address SLAM issues. Examples include the work of Rekleitis *et al*. [38] in their use of an extended Kalman filter, and work by Djugash *et al*. [39] who incorporate intersensor range data from a deployed sensor network in their approach.

In the remainder of this paper, we describe a computationally intensive but powerful approach for constructing a topological representation of a network-embedded region based on distributed observations collected from information-poor sensors. Our work addresses aspects of sensor network self-configuration, but has techniques in common with multitarget tracking, SLAM, and other problem domains where data association is an issue. The algorithm uses only detection events from the deployed sensors and is based on reconstructing plausible agents trajectories through statistical techniques.

III. Problem Description

We describe the problem of topology inference in terms of the inference of a weighted directed graph that captures the spatial relationships between the positions of the sensor nodes. The motion of multiple agents moving asynchronously through a sensor-network-embedded region can be modeled as a semi-Markov process. The network of sensors is described as a directed graph $G = (V, E)$, where the vertices $V = v_i$ represent the locations where sensors are deployed, and the edges $E = e_{i,j}$ represent the connectivity between them; an edge $e_{i,j}$ denotes a path from the position of sensor v_i to the position of sensor v_j. The motion of each of the N agents in this graph can be described in terms of their transition probability across each of the edges $A_n = \{a_{ij}\}$, as well as a temporal distribution indicating the duration of each transition D_n. The observations $O = \{o_t\}$ are a list of events detected at arbitrary times from the various vertices of the graph, which indicate the likely presence of one of the N agents at that position at that time.

The goal of our paper is to estimate the parameters describing this semi-Markov process based on a number of assumptions. We assume that each observation was generated by exactly one agent, and furthermore, that the behavior of all the agents in the system can be approximated as being homogeneous; i.e., the motion of all agents is described by the same A and D. In addition, we must make some assumptions about the distribution of the intervertex transition times. Generally, we make the assumption that the delays fit some family of distributions and are bounded within a fixed range. We will show later, however, that we relax this assumption in some situations.

In the approach that we have outlined earlier, we are making some inherent assumptions about the behavior and quality of our sensors. We assume that an individual sensor generates only a single observation for an agent despite the fact that the agent will spend some finite amount of time within the detection range. This assumption can usually be guaranteed in practice through postprocessing of raw sensor measurements; a technique sometimes referred to as "debouncing." More difficult, in practice, is the assumption that, ideally, this sensor can generate a second observation when a second agent enters its detection range, even if the first agent is still detectable. A more easily satisfied assumption is that agents do not travel close enough together for this duplicate detection situation to be encountered. If this assumption is only occasionally violated, the system can model one of these events as a missing observation (i.e. a false-negative). As we will show in later sections, our technique is robust to moderate levels of sensor error. As long as the assumptions that we have outlined earlier are approximately correct, i.e. are violated infrequently and in a nonsystematic manner, the inference process produces accurate results.

Given the observations O and the vertices V, the problem is to estimate the network connectivity parameters A and D, subsequently referred to as θ.

IV. First Level: Topology Inference Through EM

The algorithm that makes up the first level of our technique infers the connectivity of a sensor network given nondiscriminating observations. It assumes knowledge of the number of agents in the environment and attempts to augment the given observations with an additional data association that links each observation to an individual agent. The approach is based on the statistical technique of EM [40]. The algorithm iterates over constructing plausible trajectories of agent motions based on current estimates of connectivity parameters (E Step), and then, updating the parameters to maximum likelihood estimates based on the sampled trajectories (M Step). Fig. 3 shows a block diagram illustrating the control form of the inference algorithm.

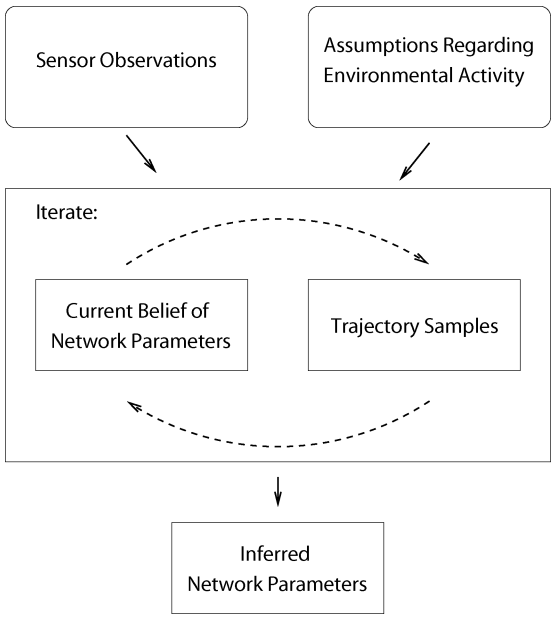

Fig. 3. Block diagram of level one of the two-level approach, where the blocks indicate algorithmic components and the arrows indicate the transfer of data.

A. Expectation Maximization

The algorithm simultaneously converges toward high-likelihood observation data correspondences and network parameters values. We iterate over the following two steps:

1) *The E-Step:* Calculates the expected log likelihood of the complete data given the current parameter guess

$$Q\left(\theta, \theta^{(i-1)}\right) = E[\log p(O, Z|\theta)|O, \theta^{(i-1)}]$$

where O is the vector of binary observations collected by each sensor, and Z represents the hidden variable that determines the data correspondence between the observations and agents moving throughout the system.

2) *The M-Step:* Updates our current parameter guess with a value that maximizes the expected log likelihood

$$\theta^{(i)} = \arg\max_{\theta} Q\left(\theta, \theta^{(i-1)}\right).$$

We employ MCEM [41] to calculate the E-Step because of the intractability of summing over the high-dimensional data correspondences. Note that there is one dimension for each element of the observation vector O. We approximate $Q\left(\theta, \theta^{(i-1)}\right)$ by drawing M samples of an ownership vector $L^{(m)} = \{l_i^m\}$ (an instance of Z) that uniquely assigns the agent i to the observation o_i in sample m

$$\theta^{(i)} = \arg\max_{\theta} \left[\frac{1}{M}\sum_{m=1}^{M} \log p(L^{(m)}, O|\theta)\right]$$

where $L^{(m)}$ is drawn using the previously estimated $\theta^{(i-1)}$ according to an MCMC sampling technique.

At every iteration, we obtain M samples of the ownership vector L, which are then used to reestimate the connectivity parameter θ (the M-Step). We continue to iterate over the E-Step and the M-Step until we obtain a final estimate of θ. At every iteration of the algorithm, the likelihood of the ownership vector tends to increase, and the process is terminated when subsequent iterations result in very small changes to θ.

In general, we make the assumption that the intervertex delays fit some family of distributions and determine the maximum likelihood parameters for each of the intervertex distributions. In a subsequent section, we will describe how we occasionally reject outlying low-likelihood delay data and omit it from the parameter update stage.

B. Trajectory Sampling

We use MCMC sampling to assign each of the observations to one of the agents, thereby breaking the multiagent problem into multiple versions of a single-agent problem. In the single-agent case, the observations O specify a single trajectory through the graph, which can be used to obtain a maximum likelihood estimate for θ. Therefore, we look for a data association that breaks O into multiple single-agent trajectories. We express this data association as an ownership vector L that assigns each of the observations to a particular agent.

Given some guess of the connectivity parameter θ, we can obtain a likely data association L using the Metropolis algorithm; an established method of MCMC sampling [42]. From our current state in the Markov Chain specified by our current observation assignment L, we propose a symmetric transition to a new state by reassigning a randomly selected observation to a new agent selected uniformly at random. This new data association L' is then accepted or rejected based on an acceptance probability, which is defined by the relative probabilities of L and L' according to the Metropolis algorithm.

The acceptance probability α can be expressed in a simple form since the trajectories described by L' differ from those in L by only a few edge transitions. Consider L as a collection of ordered nonintersecting sets containing the observations assigned to each agent $L = (T_1 \cup T_2 \cup \ldots \cup T_N), T_n = \{w_{jk}\}$, where w_{jk} refers to the edge traversal between vertices j and k. The probability of a single-agent trajectory is then the product of all of its edge transitions.

Therefore, a proposed change that reassigns the observation o_n from agent y to agent x must remove an edge traversal w from T_y and add it to T_x. Only the change in the trajectories of these two agents need be considered, since all other transitions remain unchanged.

In between each complete sample of the ownership vector L, each of the observations are tested for a potential transition to an alternative agent assignment. This testing is accomplished in random order and should provide a large-enough spacing between realizations of the Markov Chain that we can assume to have some degree of independence in-between samples.

Although our method of proposing transitions is simple and does not result in large jumps through the state space, the acceptance test can be evaluated efficiently, and hence we can afford to test many proposals. The resulting chain is ergodic and reversible, and thus, ultimately produces samples representative of the true probability distribution.

C. Delay Model

To make the algorithm more robust to realistic traffic patterns, we have introduced an intervertex delay model that allows for the possibility of agent transitions to and from sources and sinks. This makes the algorithm more robust both to shifting numbers of agents in the environment and to agents that pause or delay their motion in between sensors. Additionally, assuming the existence of sources and sinks, we can recover their connectivity to each of the sensors in our network.

In addition to maintaining a vertex that represents each sensor in our network, we introduce an additional vertex that represents the greater environment outside the monitored region: a *source/sink node*. A mixture model is employed during the E-Step of our iterative EM process in which we evaluate potential changes to agent trajectories. An intervertex delay time is assumed to arise from some specified family of distributions (e.g., a gamma distribution or a truncated normal) or else from a uniform distribution of fixed likelihood. This model allows for low-probability jumps of almost arbitrary length.

The data assigned to the internode delay distribution are assumed to be generated by direct transitions between nodes and are used during the M-Step to update our belief of the internode delay times and transition likelihoods. On the other hand, the data fit to the uniform distribution are used to model transitions from the first vertex into the sink/source node, and then, from the sink/source node to the second vertex. Therefore, they are not used for updating intervertex delay parameters of the two nodes, but rather are considered outliers and are used only for updating the belief of transitions to and from the source/sink node for the associated vertices. Note that no parameters are used to characterize the distribution suggested by the outlying data points; i.e., we are not attempting to learn the delay distribution between any particular node in the system and the sink/source node. Instead, we specify when a data point should be considered an outlier given only our current belief of the parameters for the associated delay distribution. This value cannot be estimated explicitly without attempting to parameterize a second distribution that would not be consistent with our model.

While the data assigned to the internode delay distributions are expected to be within a realistic temporal range for direct agent transitions, the delay data fit to the uniform distribution are more loosely bounded. This gives the inference technique a manner of temporarily removing agents from the system by assigning them to long transitions, or to explain events that would otherwise seem extremely unlikely such as the disappearance of an agent from one node and its almost immediate appearance at a second.

The delay model provides robustness to noise by discarding outliers in the delay data assigned to each pair of vertices and explaining their existence as transitions to and from a source/sink node. The key to this process is determining whether or not a delay value should be considered an outlier. This is implemented through a tunable parameter, called source sink likelihood (SSL) that determines the threshold probability necessary for the delay data to be incorporated into parameter updates (Fig. 4). The probability for an intervertex delay is first calculated given the current belief of the delay distribution. If this probability is lower than the SSL, then this motion is interpreted as a transition made *via* the source/sink node. The delay is given a probability equal to the SSL, and the transition is not used to update the network parameters associated with the origin and destination vertices.

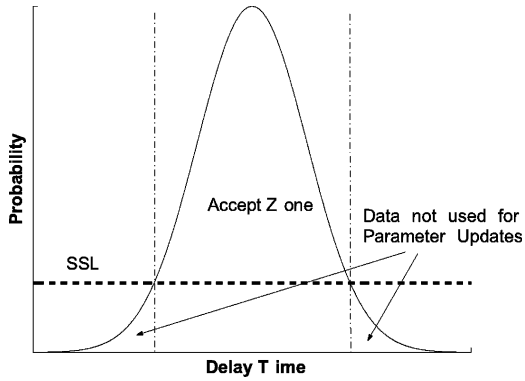

Fig. 4. Graphical description of the SSL Parameter.

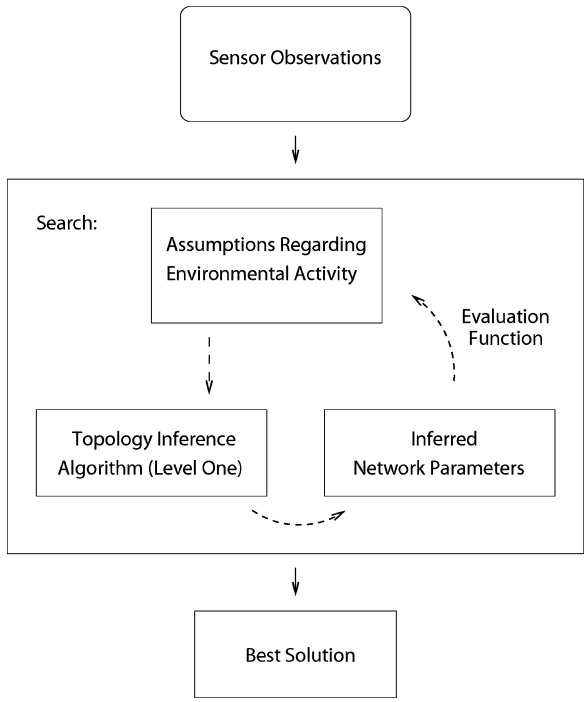

Fig. 5. Block diagram of level two of the two-level approach, where the blocks indicate algorithmic components and the arrows indicate the transfer of data.

The value assigned to the SSL parameter determines how easily the algorithm discards outliers, and hence, provides a compromise between robustness to observational noise and a tendency to discard useful data.

V. Level Two: Network Parameter Evaluation

The second level of our approach treats the topology inference algorithm described in the previous section as a "black box" and attempts to search over its input parameter space to find reasonable solutions (Fig. 5). We construct a heuristic evaluation function that quantitatively assesses a potential solution based

on the principle of Occam's Razor. The first-level topology inference algorithm takes the following inputs: the observations O, the assumed number of agents in the environment N, and the SSL parameter. The outputs of the algorithm are the network parameters θ and the *ratio* of data R_{data} incorporated into the parameter updates

$$(\theta, R_{\mathrm{data}}) \leftarrow \mathrm{alg}(O, N, \mathrm{SSL}).$$

Different input values result in different environmental assumptions, and hence, produce different outputs.

We have created a metric that attempts to assess the validity of a solution by making the assumption that a good solution both explains the majority of the data and is as *simple* as possible. This principle, known as Occam's razor, states, "if presented with a choice between indifferent alternatives, then one ought to select the simplest one." The concept is a common theme in computer science and underlies a number of approaches in AI; e.g., hypothesis selection in decision trees and Bayesian classifiers [43].

Our simplicity metric incorporates a measure of the simplicity of the transition matrix and the amount of data explained by the solution. We measure the simplicity of a transition matrix by rewarding it in inverse proportion to how close it is to a uniform belief of transition probabilities

$$A_{\mathrm{simp}} = \sum_{a_i \in A} (a_i)^{\beta}$$

where β determines the degree of the reward. We measure the utility of a given data use ratio by constructing an adjusted data ratio that attempts to reflect our belief in the solution as a function of the data used. The adjusted data ratio should incorporate the fact that some small portion of discarded data is actually optimal, but that our belief tails off rapidly as the discarded portion grows

$$R_{\mathrm{data}} = \frac{|\mathrm{Explained\ Obs}|}{|\mathrm{Obs}|}$$

$$R_{\mathrm{adj}} = \exp^{-\frac{1}{\tau}(R_{\mathrm{data}} - \gamma)^2}$$

where γ and τ describe the shape of the belief curve (Fig. 6). The final simplicity metric incorporates a weighted combination of A_{simp} and R_{adj}:

$$Q_{\mathrm{simp}} = (A_{\mathrm{simp}})^{\kappa} \times (R_{\mathrm{adj}})^{\lambda}$$

where κ and λ reflect the relative weights assigned to the two portions.

With the construction of the simplicity metric Q_{simp}, we have shifted our dependence from specific *a priori* assumptions that must be made on a case-to-case basis. Instead, we depend on more general assumptions regarding the attributes of a believable solution for our problem domain.

Instead of the two-level approach outlined in this and the previous section, an alternative approach to recovering the network connectivity parameters would be to stay within the EM framework of the fundamental algorithm. At this level, one could attempt to infer the MAP solution for a particular problem using the Q_{simp} metric as a Bayesian prior for favoring appropriate models. However, there would be some difficulties with this approach. The first disadvantage is that, although varying the number of agents at the MCMC proposal level is possible and is related to the work of Oh *et al.* [44], one must invent a suitable model capable of preventing the algorithm from improving configuration likelihoods through overfitting. The specification of the priors could be sufficiently challenging to make this approach difficult in practice. Additionally, incorporating an arbitrary prior into the Q function of the *M* step of the EM loop prevents a closed-form solution for maximizing the parameter values, hence forcing the use of a numerical estimation method. A potential danger here is that by formulating the problem in this manner, we risk destabilizing or substantially slowing the convergence of the EM algorithm, which is not guaranteed for stochastic variants. Instead, we have chosen to clearly delineate between the inner, fundamental algorithm, which, in our investigations, has shown robust dependable behavior, and a second higher level component that attempts to enforce the priors we desire.

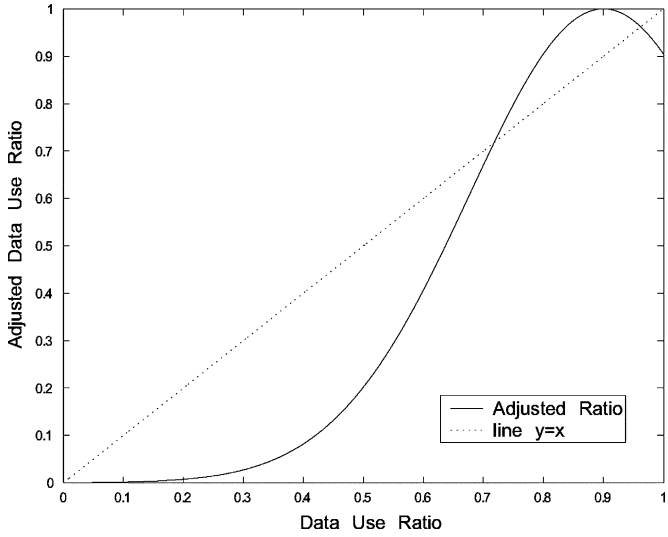

Fig. 6. Example relationship between R_{data} and R_{adj} with $\gamma = 0.9$ and $\tau = 0.1$.

VI. Simulation Results

In this section, we will examine our approach through a number of experiments conducted in simulation. We will assess the performance of the first-level topology inference algorithm and examine the effect of varying the input parameters. Then, we will justify our tuning of the parameters shaping the Q_{simp} metric.

A. Simulator

We have developed a tool that simulates agent traffic through an environment represented as a planar graph. Our simulation tool takes as input the number of agents in the system and a weighted graph, where the edge weights are proportional to mean transit times between the nodes. All connections are considered two ways; i.e., each connection is made up of two

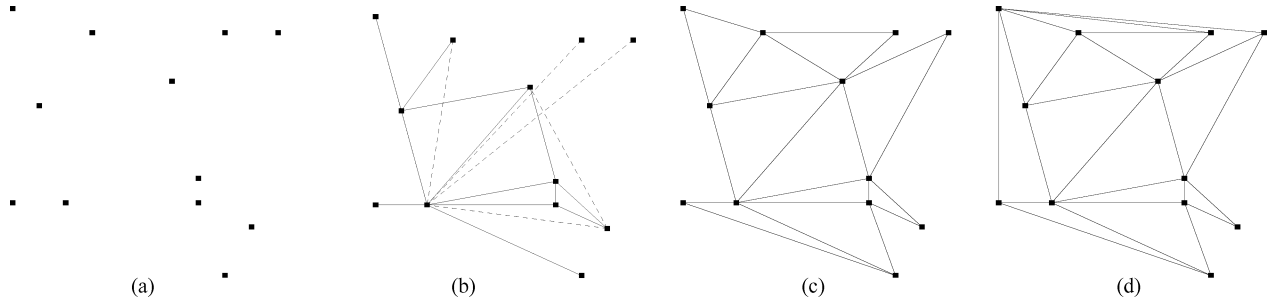

Fig. 7. Incremental belief of the topology of a 12-node, 48 (directed) edge graph using four simulated agents on 8000 observations: (a) initially; (b) after one iteration; (c) after two iterations; (d) after three iterations (the true graph). Dotted lines indicate incorrect transitions.

unidirectional edges. The output is a list of observations generated by randomly walking the agents through the environment. Internode transit times are determined based on a normal distribution with a standard deviation equal to the square root of the mean transit time.[2]

Two types of noise were modeled in order to assess performance using data that we believe more closely reflects observations collected from realistic traffic patterns. First, a "white" noise was generated by removing a percentage of correct observations and replacing them with randomly generated spurious observations. Second, a more systematic noise was generated by taking a percentage of intervertex transitions and increasing the Gaussian distributed delay time between them by an additional delay value selected uniformly at random. The range of this additional delay time was selected to be from zero to twenty times the average normal delay time. The hope is that small values of these types of noise simulate the effects of both imperfect sensors and also the tendency for agents to stop occasionally along their trajectories; e.g. to talk, use the water fountain, or enter an office for a period.

A number of experiments were run using the simulator on randomly generated planar, connected graphs. The graphs were produced by selecting a subgraph of the Delaunay triangulation of a set of randomly distributed points.

For each experiment, the results were obtained by comparing the final estimated transition matrix A' to the real transition matrix A. A graph of the inferred environment was obtained by thresholding A'. The Hamming error was then calculated by measuring the distance between the true and inferred graphs normalized by the number of directed edges m in the true graph:

$$\text{HamErr}_A = \left(\frac{1}{m}\right) \sum_{a_{ij} \in A, a'_{ij} \in A'} \left[\text{thr}(a_{ij}) - \text{thr}(a'_{ij})\right]^2$$

where thr(a) $= \lceil a_{ij} - \theta \rceil$.[3]

B. Assessment of the Topology Inference Algorithm (Level One)

Our experiments show that in the absence of significant measurement noise, the network structure can be determined very reliably with a handful of agents and a sufficient number of observations (e.g. four or ten agents). This appears to be true for various graph sizes, although for this low-noise condition, we have only tested graphs of limited size. For example, the topology of 95% of 12 node graphs was perfectly inferred with zero Hamming error for 200 simulations with four agents. Generally, the algorithm converged quickly, finding most of the coarse structure in the first few iterations and making incrementally smaller changes until convergence (Fig. 7).

While the algorithm is robust to moderate levels of "white" observational noise, its sensitivity to significant levels of systematic noise depends on the tuning of the delay model. The delay model is controlled by the SSL parameter, which determines the probability threshold for including delay data in the update of the network connectivity parameters. Note that for purposes of brevity, any value assigned to the SSL parameter in these results is given in the logarithmic form. Fig. 8 shows the result of varying the value assigned to the SSL parameter for different types of noise.

When assigned a high-SSL value, the mixture approach for modeling delays was successful at minimizing the effects of systematic noise. Even when 10% of the delay times were uniformly increased, the Hamming error of the inferred transition matrix was near zero [Fig. 8(a)]. When the SSL parameter was assigned a value of zero, the algorithm had no method of discarding delay data and had to update its network parameters given all the observations.

Moderate amounts of "white," unbiased observational noise could be handled regardless of the tuning of the delay distribution mixture [Fig. 8(b)]. However, the inferred transition beliefs were strongly effected by heavy amounts of this type of noise. The effect of randomly inserting and deleting observations is to skew the distribution of likely sampled trajectories. Hence, the inference technique develops an incorrect belief of the underlying network and its intersensor transition probabilities. Since determining the correlation between the various sensor observations is key to our approach, it is unsurprising that, after about 10% of both missing and spurious observations, the performance of the algorithm drops significantly.

The robust behavior of the algorithm under noisy conditions demonstrates both the general stability of the sampling-based approach and the success of the delay model. With an appropriately selected value assigned to the SSL parameter, the technique can infer highly accurate connectivity information even with moderate levels of both systematic and white noise.

[2]Negative transit times are rejected.
[3]A threshold value of $\theta = 0.1$ was selected for our experiments.

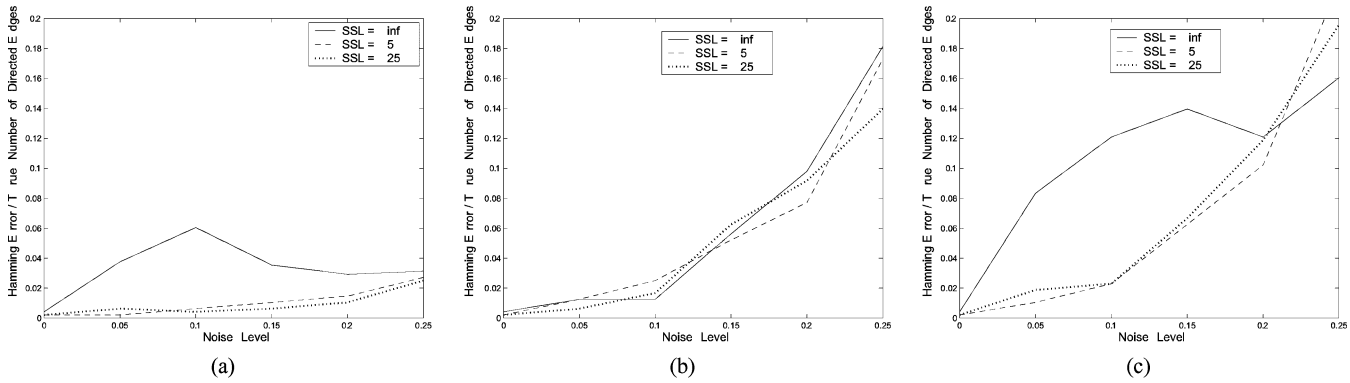

Fig. 8. Hamming error as a function of observational noise. The results are averaged over ten graphs using four simulated agents on 12-node, 48-edge graphs with 4000 observations. The horizontal axis indicates proportions of: (a) systematic noise; (b) white noise; (c) both systematic and white noise.

TABLE I
TABLE OF VALUES USED TO SHAPE THE SIMPLICITY QUOTIENT Q_{SIMP}

β	γ	τ	κ	λ
2	0.9	0.2	2	1

C. Automatic Parameter Selection (Level Two)

In this section, we attempt to validate our general approach for selecting nearly-optimal input parameters for the first-level topology inference algorithm by using attributes of the solution it produces. We select parameters defining the Q_{simp} metric based both on domain knowledge and experimental methods (Table I).

In order to justify these parameter values and to assess the effectiveness of this approach, we conducted a number of simulations in which we varied the input parameters and looked for a correlation between the performance of the algorithm and the simplicity metric.

1) Effect of Input Parameters: Input parameters that resulted in good algorithm performance also resulted in simple models, as measured by the Q_{simp} quotient (Figs. 9 and 10). For example, under noise-free operation, the most accurate solutions also generated the highest Q_{simp} values. This result gives support for our adoption of Occam's Razor as a mechanism for selecting input parameters.

The accuracy of the solution we obtain depends heavily on the assumed number of agents in the environment. The lowest error was consistently observed when the assumed number of agents was set to the correct value, and generally, the closer to the correct value this parameter was set, the better the results. Overestimating the assumed number of agents had less impact on accuracy than underestimation.

A correctly tuned SSL parameter was also important to the accuracy of the final solution. As the input value for this parameter was increased, there appeared to be a "phase transition" in the accuracy of the results. Past a certain threshold, the error suddenly increased dramatically. Interestingly, the best results for both the inferred mean delay times and transition likelihoods seems to be obtained just before this sudden degradation in performance.

2) Direct Correlation Between Performance and the Simplicity Quotient: When the error in the inferred transition matrix was plotted against the value obtained for the simplicity quotient Q_{simp} for a number of simulations, there was evidence of a definite correspondence (Fig. 11). The effect appeared robust to moderate levels of observational noise and different sizes of graphs. While, the shaping of the Q_{simp} metric is an ongoing task, the current parameter values are adequate to demonstrate the correlation between the correctness and simplicity of the inferred transition matrix. In our experimental work, described in the next section, we took advantage of this correlation to select appropriate input parameters since the 'correct' values were unknown.

VII. EXPERIMENTS CONDUCTED ON A HETEROGENEOUS SENSOR NETWORK

In order to test our technique under real-world conditions, we set up an experiment using a network of sensors and analyzed the results using our two-level approach.

A. Experimental Setup

The sensor network was made up of two types of platforms: vision-based sensors running on single-board computers; and photocell-based sensors running on low-powered commercial devices. Both types of sensors were programmed to act as simple motion detectors sending event messages to a central server, which logged the origin and time of the activity.

The vision-based sensor nodes were constructed of inexpensive single-board computers running the Linux operating system networked together over Ethernet using custom software. A standard client/server architecture was implemented over the Transmission control protocol/Internet protocol (TCP/IP) in the C language. The client software functions as a motion detector employing a Labtech Webcam.

The second type of motion sensor consisted of a flashlight and a MICA2 Crossbow wireless sensor with an attached sensor board (Fig. 12). Custom firmware developed for this project ran on the TinyOS real-time operating system [45], [46]. The flashlight beam was focused on the photocell of the sensor board. Any decrease in the intensity of the light was detected by the firmware which sent an event message to a central server.

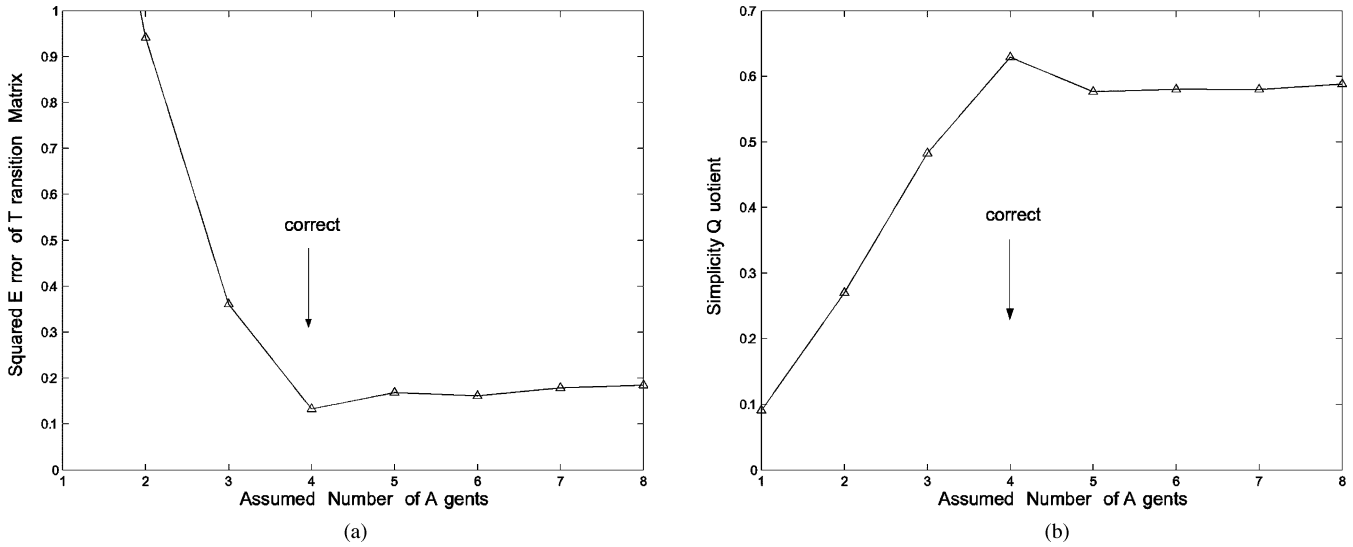

Fig. 9. Effect of varying assumed numbers of agents on performance and the simplicity quotient. Results are averaged over 20 graphs using four simulated agents on 12-node, 48-edge graphs with 4000 observations. The arrow indicates the true system parameter for the number of agents, which is associated with external error and simplicity values.

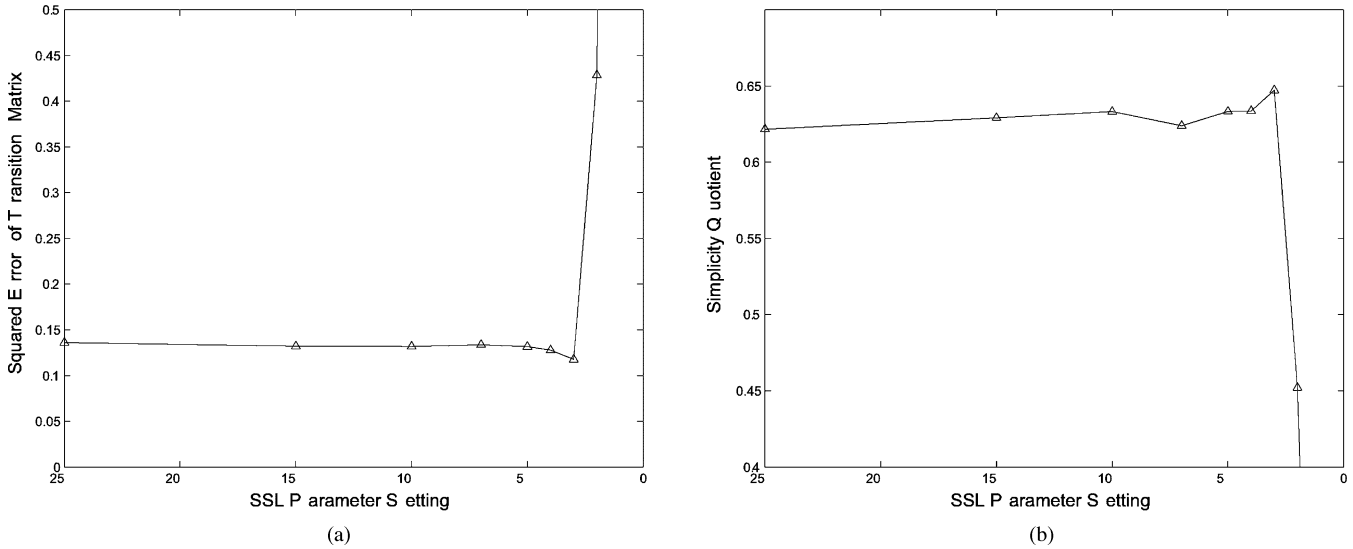

Fig. 10. Effect of varying the value assigned to the SSL parameter on performance and the simplicity quotient. Results are averaged over 20 graphs using the four simulated agents on 12-node, 48-edge graphs with 4000 observations.

The experiment was conducted in the hallways of one wing of an office building (Fig. 13). The data were collected during a 6 h 30 min period from 10:00 a.m. to 4:30 p.m. on a weekday. In total, approximately 4700 time-stamped events were collected.

The three low-powered sensors were placed close to the central server to accommodate their shorter communication range. Despite this layout, the furthest low-powered sensor, I, was only able to communicate to the central server via an intermediate sensor, H, using a multihop communication protocol.

B. Ground Truth

Ground truth values were calculated in order to assess the results inferred by the approach. A topological map of the environment was determined [Fig. 15(a)] based on an analysis of the sensor network layout. In addition, intervertex transitions times for the connected sensors were recorded with a stopwatch for a typical subject walking at a normal speed (Table II).

C. Selection of Input Parameters

To determine appropriate input parameters for our inference algorithm, we conducted an exhaustive search over the range of $N = 2, \ldots, 6$ and $\mathrm{SSL} = -7, \ldots, -3$ (Fig. 14). We then chose the output values that maximized our Q_{simp} metric. (We used the same shaping parameters for the Q_{simp} metric that were verified through simulations.) The maximizing arguments were: $N = 5$ and $\mathrm{SSL} = -5$. Therefore, we selected the solution generated by these parameter values as our inferred network.

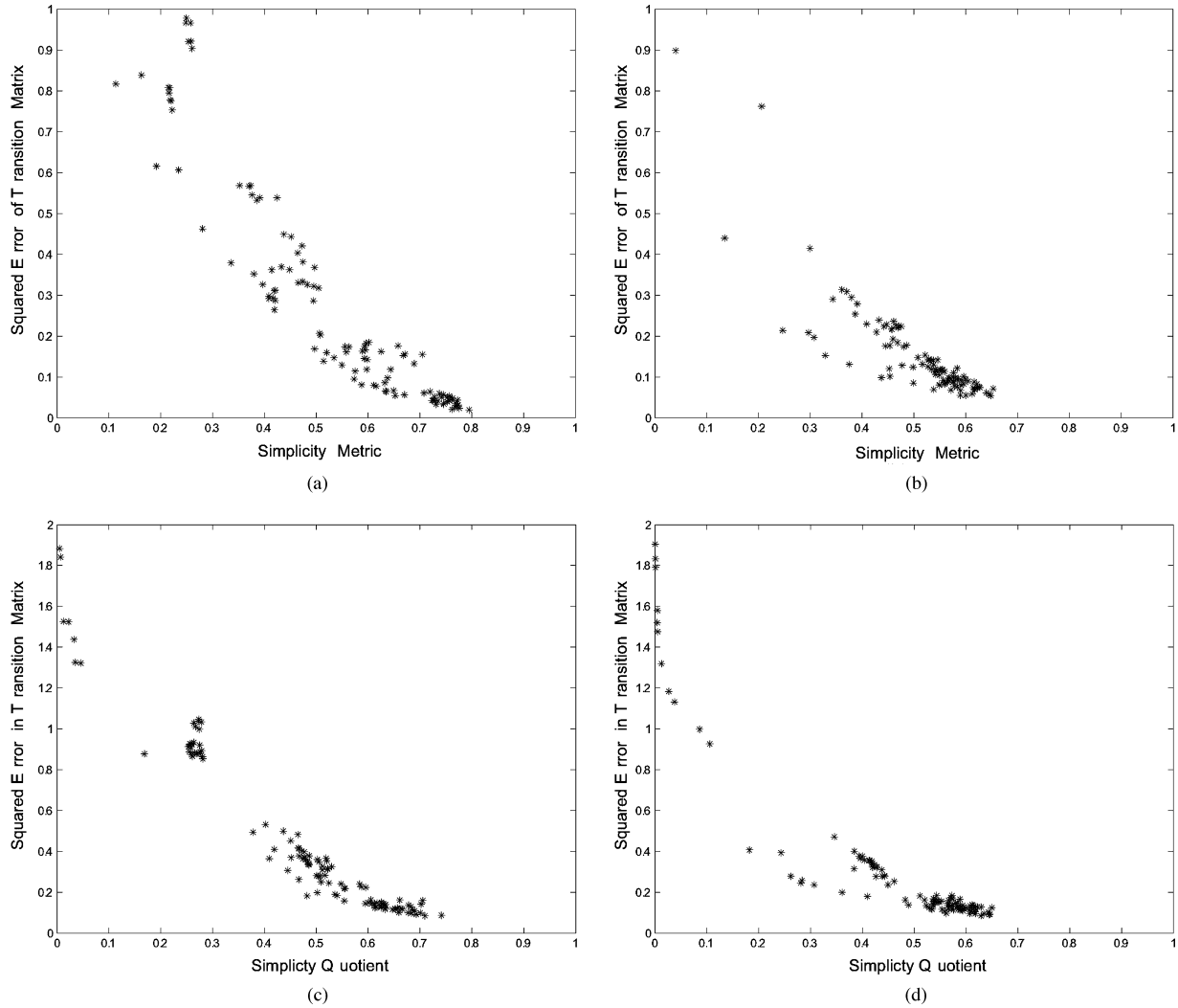

Fig. 11. Mean error in the inferred transition matrix elements plotted against Q_{simp} for data obtained from the simulator with four true agents. Input parameters to the algorithm were varied: assumed number of agents from two to seven; and SSL from -2 to -7. The results are obtained using the simulator on: (a) four random graphs of six nodes, 14 edges with 2000 noise-free observations (144 trials); (b) four random graphs of six nodes, 14 edges with 2000 observations containing 5% white and systematic noise (144 trials); (c) four random graphs of 12 nodes, 48 edges with 4000 noise-free observations (144 trials); (d) four random graphs of 12 nodes, 48 edges with 4000 observations containing 5% white and systematic noise (144 trials). Observe that the solutions obtaining high simplicity quotient values are consistently among those with the lowest transition matrix error.

(a) (b)

Fig. 12. (a) Complete setup and (b) close up of a deployed photocell-based sensor constructed out of a flashlight and a crossbow wireless sensor. (Plastic containers were used as protective covering during experiments.)

D. Assessment of Results

Except for a few small differences, the network parameters computed by our topology inference algorithm closely corresponded to the ground truth values. Fig. 15 compares the analytically determined and inferred topological maps. Disregarding reflexive links, the difference between the inferred and "ground truth" results amounted to a Hamming error of 2. The two significant errors are: an extra edge found between sensors A and B; and a missing one-way edge from sensor D to I.

The missing edge from D to I is likely due to the tendency of people to go straight rather than turn right when navigating the corridor on the bottom right of the region (heading left), as shown in Fig. 13. This missing inferred connection demonstrates a limitation in the approach of exploiting motion in the environment. Our technique can only learn traffic patterns common enough to be easily recognized and distinguished.

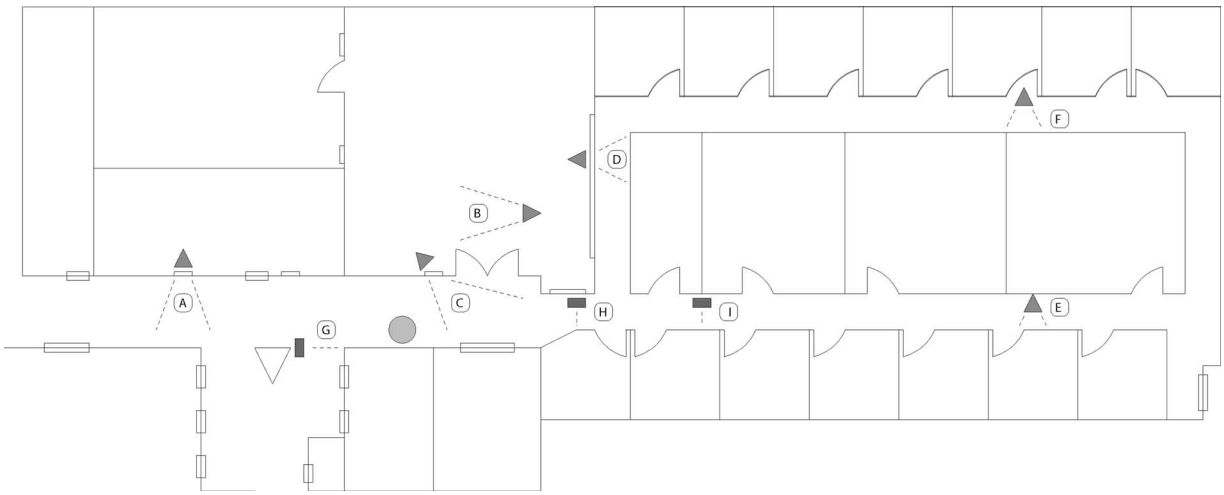

Fig. 13. Layout of the nine-sensor (heterogeneous) network used for the experiment. Labeled triangles represent vision-based sensor positions (A–F) and labeled rectangles represent low-powered photobased sensors (G–I). The circle represents the location of the central server.

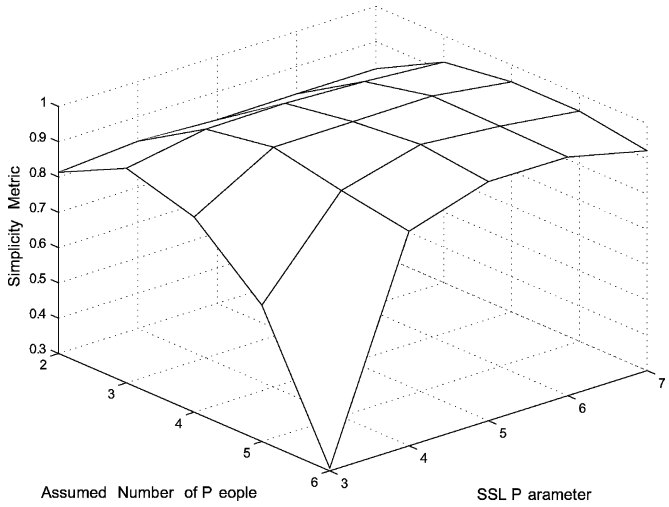

Fig. 14. Plotof the Q_{simp} metric as a function of input parameters.

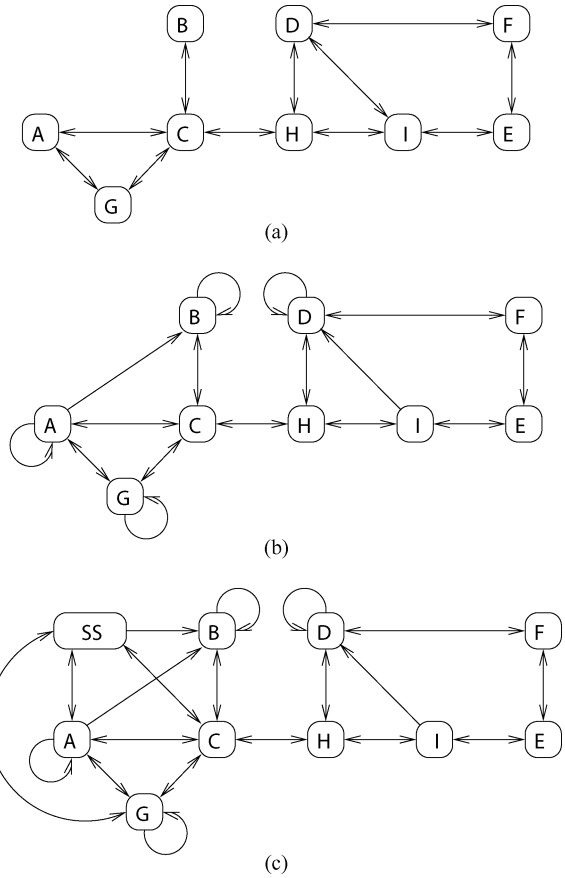

Fig. 15. Topological maps of the environment that were: (a) analytically determined based on the layout; (b) inferred by the algorithm; (c) inferred by the algorithm including the source/sink node.

The extra edge found leading from sensor A to sensor B is likely due to a correlation in the detection intervals between these two nodes. Since both sensors are in boundary locations, they are likely to receive events caused by people that then leave the monitored region for some time. Both of these areas see heavy traffic, much of which does not directly lead to another monitored area. Fig. 16(c) shows the inferred delay distribution between these two nodes; the distribution is far from what would be expected from "through-traffic." It is possible that erroneous edges of this type could be eliminated based on the shape of their associated delay distribution. Although a probabilistic belief of likely delay distributions could be incorporated into the algorithm, a simpler solution could be to implement a post-processing technique. For example, edges could be eliminated if the standard deviation of their delay distribution is large in comparison to the mean.

It should be noted that, in this paper, a truncated normal was used to model the delay distributions; however, results were also obtained using a gamma distribution. Interestingly, better results were obtained using the truncated normal. It is possible that when using this distribution family, the algorithm is better at symmetrically rejecting outliers on both sides of the mean, and as a consequence, finds parameters that form tighter, more

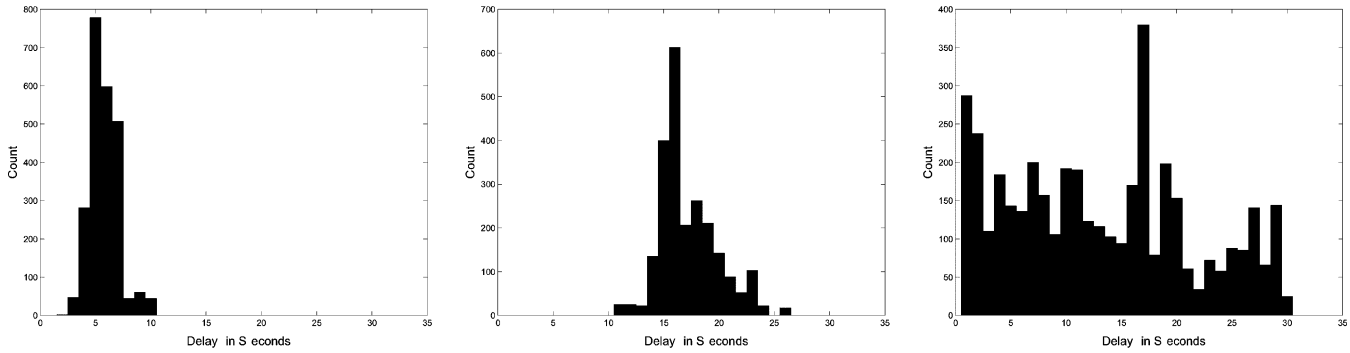

Fig. 16. Examples of delay distributions inferred for: (a) sensor D to sensor H; (b) sensor F to sensor D; (c) sensor A to sensor B (an erroneously inferred edge).

TABLE II
COMPARISON OF TIMED AND INFERRED DELAY TIMES (BOTH WAYS) BETWEEN SENSORS. ALL VALUES ARE ROUNDED TO THE NEAREST SECOND

Connection	Timed	Inferred
A,G	6	8 / 11
A,C	9	12 / 10
B,C	5	6 / 8
C,G	5	5 / 5
C,H	5	6 / 6
D,F	14	15 / 17
D,H	5	5 / 6
D,I	6	7 / 7
E,F	13	13 / 13
E,I	13	15 / 14
H,I	4	4 / 4

decisive intervertex distributions. Presumably, this has the effect of improving the accuracy of the inference process.

The mean transition times produced by the algorithm were consistent to those determined by the stopwatch (Table II, Fig. 16). Additionally, the number of reflexive links or self-connections inferred by the algorithm also seem to be consistent with the expected results. Except for node D, the other reflexive links all occur on sensors that are on the boundary of the monitored region [Fig. 15(b)]. Traffic passing node G, for example, might be correlated to the arrival of the elevator. (The elevators are located to the right, immediately below sensor G. See Fig. 13.) Hence, a pattern in the detection events that was uncorrelated to other sensor detections would lead to a belief in a self-connection. The self-connection seen on sensor D might be due to regular traffic to an office located in the hallway between sensor D and sensor F.

The connections to the source/sink node also occur only for boundary nodes [Fig. 15(c)] and are, therefore, consistent with an analytical assessment of the traffic patterns. Since traffic commonly enters and exits the monitored region via one of the boundary nodes, the inference algorithm should commonly employ the source/sink node in order to bring the agent back into the system.

VIII. CONCLUSION

In this paper, we presented a method for inferring the topology of a sensor network given nondiscriminating observations of activity in the monitored region. Our technique recovers the network connectivity information opportunistically through the exploitation of existing motion. This task is accomplished based on no prior knowledge of the relative locations of the sensors and only a limited knowledge of the type of activity present in the environment.

We described a formulation of the problem such that it could be iteratively solved with a stochastic EM algorithm. The method uses the observational data and MCMC sampling to construct likely trajectories describing the motion of agents present in the environment. By inferring underlying patterns in their motions, the technique recovers the connectivity relationships between the sensors and constructs a Markov model describing their behavior. From this information, a topological description of the network can be constructed.

The work also entails some open problems. We assume that agents in the system tend to transit the sensors separately. While we can tolerate some degree of inconsistency with this assumption, an explicit model might be required to deal with an environment in which this happens with high frequency. We also assume that the behavior of the agents in the environment is statistically independent. Dealing explicitly with correlated behavior is an interesting problem and is related to the work of Haigh [47].

Results from both simulations and experiments have shown the ability of our algorithm to generate accurate results under conditions of sensor noise and complex traffic patterns. The technique compares favorably to related approaches and could have promising real-world applications in the area of sensor network calibration and self-configuration.

ACKNOWLEDGMENT

We would like to thank P. Giguere, I. Rekleitis, M. Garden, E. Bourque, J. Sattar, S. Simhon, P. Di Marco, L. A. Torres-Méndez, D. Burfoot, and others of the Mobile Robotics Laboratory, along with D. Fleet of the University of Toronto, and the Center for Intelligent Machines (CIM) administration for their technical help and good ideas. In addition, we thank M. Theberge for the photographs, proof reading, and valuable assistance during the experiment.

REFERENCES

[1] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, and J. Anderson, "Wireless sensor networks for habitat monitoring," presented at the ACM Int. Workshop Wireless Sensor Netw. Appl. (WSNA 2002), Atlanta, GA, Sep.

[2] A. Cerpa, J. Elson, D. Estrin, L. Girod, M. Hamilton, and J. Zhao, "Habitat monitoring: Application driver for wireless communications technology," presented at the 2001 ACM SIGCOMM Workshop Data Commun. Latin Amer. Caribbean, San Francisco, CA, Apr.

[3] C. R. Barnes, J. R. Delaney, B. M. Howe, and N. Penrose, "Neptune: A regional cabled observatory in the northeast Pacific," presented at the White Paper Ocean Res. Interact. Obs. Netw. (ORION) Meet., San Diego, CA, Jan. 2004.

[4] N. Bulusu, D. Estrin, L. Girod, and J. Heidemann, "Scalable coordination for wireless sensor networks: Self-configuring localization systems," in *Proc. 6th Int. Symp. Commun. Theory Appl. (ISCTA 2001)*, Ambleside, U.K., Jul., pp. 1–6.

[5] N. Correal and N. Patwari, "Wireless sensor networks: Challenges and opportunities," presented at the MPRG/Virgina Tech. Wireless Symp., Blacksburg, VA, 2001.

[6] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. A. Cayirci, "A survey on sensor networks," *IEEE Commun. Mag.*, vol. 40, no. 8, pp. 102–114, Aug. 2002.

[7] S. Capkun, M. Hamdi, and J.-P. Hubaux, "GPS-free positioning in mobile ad-hoc networks," presented at the 34th Annu. Hawaii Int. Conf. Syst. Sci., Maui, HI, Jan. 2001.

[8] A. Savvides, C. Han, and M. Strivastava, "Dynamic fine-grained localization in ad-hoc networks of sensors," in *Proc. 7th Annu. Int. Conf. Mobile Comput. Netw.*, Rome, Italy, 2001, pp. 166–179.

[9] N. Bulusu, J. Heidemann, and D. Estrin, "GPS-less low cost outdoor localization for very small devices," *IEEE Pers. Commun. Mag.*, vol. 7, no. 5, pp. 28–34, Oct. 2000.

[10] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan, "The cricket location-support system," in *Proc. Mobile Comput. Netw.*, 2000, pp. 32–43.

[11] D. Niculescu and B. Nath, "*Ad hoc* positioning system (APS) using AoA," in *Proc. 22nd Annu. Joint Conf. IEEE Comput. Commun. Soc.*, San Francisco, CA, 2003, vol. 3, pp. 1734–1743.

[12] D. Marinakis, D. Meger, I. Rekleitis, and G. Dudek, "Hybrid inference for sensor network localization using a mobile robot," in *Proc. AAAI Natl. Conf. Artif. Intell.*, Vancouver, Canada, Jul. 2007, pp. 1089–1094.

[13] H. Shatkay and L. P. Kaelbling, "Learning topological maps with weak local odometric information," in *Proc. Int. Joint Conf. Artif. Intell.*, San Mateo, CA, 1997, pp. 920–929.

[14] H. Choset and K. Nagatani, "Topological simultaneous localization and mapping (SLAM): Toward exact localization without explicit localization," *IEEE Trans. Robot. Autom.*, vol. 17, no. 2, pp. 125–137, Apr. 2001.

[15] E. Remolina and B. Kuipers, "Towards a general theory of topological maps," *Artif. Intell.*, vol. 152, no. 1, pp. 47–104, 2004.

[16] A. Ranganathan and F. Dellaert, "Data driven MCMC for appearance-based topological mapping," presented at the Robot. Sci. Syst., Cambridge, MA, Jun. 2005.

[17] D. Makris, T. Ellis, and J. Black, "Bridging the gaps between cameras," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR 2004)*, Washington, DC, Jun., vol. 2, pp. 205–210.

[18] D. Marinakis and G. Dudek, "Topological mapping through distributed, passive sensors," in *Proc. Int. Joint Conf. Artif. Intell.*, Hyderabad, India, Jan. 2007, pp. 2147–2152.

[19] A. T. Ihler, J. W. Fisher, III, R. L. Moses, and A. S. Willsky, "Nonparametric belief propagation for self-calibration in sensor networks," *IEEE J. Sel. Areas Commun.*, vol. 23, no. 4, pp. 809–819, Apr. 2005.

[20] M. A. Paskin, C. E. Guestrin, and J. McFadden, "A robust architecture for inference in sensor networks," in *Proc. 4th Int. Symp. Inf. Process. Sensor Netw. (IPSN 2005)*, Apr., pp. 55–62.

[21] K. Dantu and G. S. Sukhatme, "Rethinking data-fusion based services in sensor networks," in *Proc. 3rd IEEE Workshop Embedded Netw. Sensors*, 2006, pp. 3665–3672.

[22] G. P. Stein, "Tracking from multiple view points: Self-calibration of space and time," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jun., 1999, vol. 1, pp. 521–527.

[23] R. B. Fisher, "Self-organization of randomly placed sensors," in *Proc. Eur. Conf. Comput. Vis.*, Copenhagen, Denmark, May 2002, pp. 146–160.

[24] C. Stauffer and K. Tieu, "Automated multi-camera planar tracking correspondence modeling," in *Proc. IEEE Comput. Vis. Pattern Recognit.*, Jul., 2003, vol. 1, pp. 259–266.

[25] A. Rahimi, B. Dunagan, and T. Darrell, "Simultaneous calibration and tracking with a network of non-overlapping sensors," in *Proc. CVPR 2004*, Jun., vol. 1, pp. 187–194.

[26] O. Javed, Z. Rasheed, K. Shafique, and M. Shan, "Tracking across multiple cameras with disjoint views," in *Proc. 9th IEEE Int. Conf. Comput. Vis.*, Nice, France, Oct. 2003, vol. 2, pp. 952–957.

[27] T. Ellis, D. Makris, and J. Black, "Learning a multicamera topology," in *Proc. Joint IEEE Int. Workshop Vis. Surveill. Perform. Eval. Tracking Surveill.*, Nice, France, Oct.2003, pp. 165–171.

[28] Y. Bar-Shalom, Ed., *Multitarget Multisensor Tracking: Advanced Applications*. vol. II, Norwod, MA: Artech House, 1992.

[29] 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. 2nd Int. Workshop Inf. Process. Sensor Netw. (IPSN)*, Palo Alto, CA, Apr. 2003, pp. 113–128.

[30] M. M. S. Oh, P. Chen, and S. Sastry, "Instrumenting wireless sensor networks for real-time surveillance," in *Proc. Int. Conf. Robot. Autom.*, May 2006, pp. 3128–3133.

[31] M. Rosencrantz, G. Gordon, and S. Thrun, "Locating moving entities in indoor environments with teams of mobile robots," in *Proc. 2nd Int. Joint Conf. Auton. Agents Multiagent Syst.*, Melbourne, Australia, 2003, pp. 233–240.

[32] J. Shin, L. J. Guibas, and F. Zhao, "A distributed algorithm for managing multi-target identities in wireless *ad-hoc* sensor networks," in *Proc. 2nd Int. Workshop Inf. Process. Sensor Netw. (IPSN 2003)*, Palo Alto, CA, Apr., pp. 223–238.

[33] C. Rasmussen and G. Hager, "Probabilistic data association methods for tracking multiple and compound visual objects," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 6, pp. 560–576, Jun. 2001.

[34] T. Huang and S. J. Russell, "Object identification in a Bayesian context," in *Proc. IJCAI*, 1997, pp. 1276–1283.

[35] T. Huang and S. J. Russell, "Object identification: A Bayesian analysis with application to traffic surveillance," *Artif. Intell.*, vol. 103, no. 1–2, pp. 77–93, 1998.

[36] H. Pasula, S. Russell, M. Ostland, and Y. Ritov, "Tracking many objects with many sensors," in *Proc. IJCAI 1999*, Stockholm, Sweden, pp. 1160–1171.

[37] F. Dellaert, S. Seitz, C. Thorpe, and S. Thrun, "EM, MCMC, and chain flipping for structure from motion with unknown correspondence," *Mach. Learn., Spec. Issue Markov Chain Monte Carlo Method*, vol. 5, pp. 45–71, 2003.

[38] I. Rekleitis, D. Meger, and G. Dudek, "Simultaneous planning localization, and mapping in a camera sensor network," *Robot. Auton. Syst. (RAS) J., Spec. Issue Plann. Uncertainty Robot.*, vol. 54, no. 11, pp. 921–932, Nov. 2006.

[39] J. Djugash, G. Kantor, S. Singh, and W. Zhang, "Range-only SLAM for robots operating cooperatively with sensor networks," in *Proc. Int. Conf. Robot. Autom.*, May 2006, pp. 2078–2084.

[40] A. Dempster, N. Laird, and D. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *J. R. Stat. Soc.*, vol. 39, pp. 1–38, 1977.

[41] G. Wei and M. Tanner, "A Monte-Carlo implementation of the EM algorithm and the poor man's data augmentation algorithms," *J. Amer. Stat. Assoc.*, vol. 85, no. 411, pp. 699–704, 1990.

[42] M. Tanner, *Tools for Statistical Inference*, 3rd ed. New York: Springer-Verlag, 1996.

[43] T. M. Michell, *Machine Learning*. Boston, MA: McGraw-Hill, 1997.

[44] S. Oh, S. Russell, and S. Sastry, "Markov chain Monte Carlo data association for general multiple-target tracking problems," in *Proc. 43rd IEEE Int. Conf. Decis. Control (CDC)*, Paradise Island, Bahamas, Dec. 2004, vol 1, pp. 735–742.

[45] P. Levis, S. Madden, D. Gay, J. Polastre, R. Szewczyk, A. Woo, E. Brewer, and D. Culler, "The emergence of networking abstractions and techniques in tinyos," in *Proc. 1st USENIX/ACM Symp. Netw. Syst. Des. Implementation (NSDI, 2004)*, pp. 1–14.

[46] D. Gay, P. Levis, and D. Culler, "Software design patterns for tinyos," in *Proc. ACM SIGPLAN/SIGBED 2005 Conf. Lang. Complilers Tools Embedded Syst. (LCTES, 2005)*, Chicago, IL, Jun., pp. 40–49.

[47] K. Z. Haigh, W. Foslien, and V. Guralnik, "Visual query language: Finding patterns in and relationships among time series data," in *Proc. 7th Workshop Min. Sci. Eng. Datasets*, Lake Buena Vista, FL, Apr. 2004.

Dimitri Marinakis received the B.Sc. degree from the University of Victoria, BC, Canada, in 1999. He is working toward the Ph.D. degree in computer science from the Centre for Intelligent Machines, McGill University, Montreal, QC, Canada.

He has been in industry in the fields of embedded systems and wireless communications. His current research interests include applying statistical and probabilistic methods to problems in sensor networks and mobile robotics.

Gregory Dudek received the Ph.D. degree from the University of Toronto, Toronto, ON, Canada.

He is currently a Professor in the School of Computer Science, McGill University, Montreal, QC, Canada, and a member of the McGill Research Centre for Intelligent Machines. He directs the McGill Mobile Robotics Laboratory. His currrent research interests include perception for mobile robotics, vision-based robotics, computational vision, and collaborative filtering.