IEEE International Conference on Robotics and Automation, Barcelona, 2005.

Learning Sensor Network Topology through Monte Carlo Expectation Maximization *

Dimitri Marinakis, Gregory Dudek
Centre for Intelligent Machines, McGill University
3480 University St, Montreal, Quebec, Canada H3A 2A7
{*dmarinak,dudek*}*@cim.mcgill.ca*

David J. Fleet
Dept. of Computer Science, University of Toronto
10 King's College Rd, Toronto Ontario, Canada
fleet@cs.toronto.edu

***Abstract*— We consider the problem of inferring sensor positions and a topological (i.e. qualitative) map of an environment given a set of cameras with non-overlapping fields of view. In this way, without prior knowledge of the environment nor the exact position of sensors within the environment, one can infer the topology of the environment, and common traffic patterns within it. In particular, we consider sensors stationed at the junctions of the hallways of a large building. We infer the sensor connectivity graph and the travel times between sensors (and hence the hallway topology) from the sequence of events caused by unlabeled agents (i.e. people) passing within view of the different sensors. We do this based on a first-order semi-Markov model of the agent's behavior. The paper describes a problem formulation and proposes a stochastic algorithm for its solution. The result of the algorithm is a probabilistic model of the sensor network connectivity graph and the underlying traffic patterns. We conclude with results from numerical simulations**

***Index Terms*— sensor networks, learning, Markov Chain Monte Carlo, Expectation Maximization**

I. Introduction

In this paper we propose and solve the non-standard problem of inferring the relative positions of a set of sensors that all look at the same scene, yet which have completely non-overlapping fields of view. This is in contrast to the somewhat more traditional problem of inferring the structure of a scene or tracking activities using a network of sensors whose positions are known. Inferring the position of well-separated cameras can be viewed as the precursor to this traditional problem. Although, on the surface this seems to be a difficult problem, in our approach, we exploit the motion of objects between the fields of view of the different sensors to probabilistically infer their positions. That is, when an object leaves the neighborhood of one sensor and subsequently appears within range of another, this suggests that the two sensors are proximal. This leads, as well, to a hierarchy of problems as a function of the sensor separation and the domain assumptions involved. We will specify this new problem hierarchy in the next section.

In this paper we will infer the *topology* (i.e. connectivity) of a set of sensors in the plane assuming they are separated by substantial distances in the environment relative to their measurement range. For example, our sensors might be stationed at the intersections between the hallways in a large building. The result of our inference procedure is the topological layout of the hallways, based on very weak assumptions about what the sensors can discriminate. All we demand of the sensors is that they can probabilistically detect and signal the passage of an agent (e.g. a person or other target) through the intersection at which they are stationed. While we can cope with both spurious detections (false positives) and missed detections (false negatives), we assume that these types of errors are less probable than correct measurements. Our formulation of the problem does not require that individual agents be distinguished or tracked by the sensor network. In practice, we are developing an implementation of this system based on video data, but that is outside the scope of this paper.

The simplest notional application of the work would allow a set of sensors to be "dropped" into an environment and to automatically learn the topology of their layout.

Our approach to the problem can be divided into two main inter-dependent sub-problems: inferring the association between observations and individual people (or agents) moving in the world, and inferring the possible transitions between the sites where the sensors are located. Our approach is to jointly solve for the transition probabilities, delays, and data associations, and then subsequently determine the topological map from the transition probabilities.

II. Background

Self-localization and other more general self-configuration algorithms are considered important issues for both multi-robot systems and for sensor networks [1] [2]. For many outdoor applications such as large scale monitoring or tracking, a desired property is ad hoc deployability, and yet the sensors will generally be too small and inexpensive to employ advanced positioning techniques such as GPS. In addition, even if GPS is available, it can not be used for estimating traffic flow or connectivity information.

Most network self-configuration efforts to date aim to recover the relative or metric location of each node [3] [4] or to self-organize for routing and networking efficiency purposes [5]. Several authors have also considered the

*This work partially supported by NSERC.

issues that arise when using a sensor network with known sensor positions to track a robot as it moves and refines its quantitative pose estimate [6], [7]. In addition, Batalin *et al.* also consider the iterative update of both the robot and sensor positions based on range data in two dimensions (although results in [8] are based on simulation data).

Some recent work in the related area of multi-camera calibration has looked at automatically learning the correspondence between non-overlapped cameras by exploiting the motion of agents present in the environment. These efforts present methods for learning the spatial and temporal relationships between the cameras, ultimately recovering the topology and inter-camera travel times of the network.

Javed *et al.* [9] incorporated learning the topology of a camera network as part of the task of tracking multiple agents across disjoint fields of view. They employed a Parzen window technique that looks for correspondences in agent velocity, inter-camera travel time, and the location of agent exit and entry in the fields of view of the camera. Their method was verified experimentally on two and three camera networks.

Ellis, Markis, and Black [10] [11] approached the topology inference problem in the context of camera-based sensing by exploiting temporal correlation in observations of agents' movements. They outlined an approach in which they first identified entrance and exit points in camera fields of view to generate a graph from video data. They then used a thresholding technique to look for peaks in the temporal distribution of travel times between entrance-exit pairs; a clear peak suggesting that the cameras are linked. 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 method can be used to efficiently produce an approximate network connectivity graph, but when the network dynamics are complex or the traffic distribution exhibits substantial variation it would appear the technique will have substantial difficulty.

Unlike the Parzen window, or thresholding approach, we propose a method for finding the topology of a network by observing agents in the environment and constructing plausible trajectories of their motion. This is closely related to multi-target tracking, which is a well established research area in sensor networks [12] [13] and multi-robot systems [14]. One of the key difficulties is 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 [15] have been devised to handle this situation. Other work poses the target identity problem as a data association problem [16] [17]. Pasula et al. [18] approached a traffic monitoring problem in this manner and demonstrated promising results on small networks through the application of Monte-Carlo Expectation Maximization (MCEM).

Expectation Maximization (EM) is a well known statistical method for parameter estimation for incomplete data models [19] and has been applied in many fields including robotics [20], and tracking [21]. The technique iteratively calculates the expected log-likelihood of the data and current parameter guess with respect to the incomplete data (E Step), and then updates the parameter guess to maximize the expected log-likelihood (M Step).

MCEM expands the scope of this technique by executing the E Step through Markov Chain Monte Carlo (MCMC) sampling[22]. MCEM and other stochastic versions of EM have been applied to the vision problem of structure from motion [23] and to other fields such as bio-informatics [24].

III. Description of Algorithm

In this section we describe an algorithm for inferring the connectivity of a sensor network given binary observations collected from each of the sensors.

A. The Problem

We describe the problem of topology inference in terms of the inference of a weighted directed graph which captures the spatial relationships between the positions of the sensors' nodes. The motion of multiple agents moving asynchronously through a sensor network 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 work is to estimate the parameters describing this semi-Markov process. For now, we assume that we know the number of agents, and that their behavior can be approximated as being roughly homogeneous; i.e. the motion of all agents are described by the same A and D. In addition, we assume that the inter-vertex transition times are normally distributed, but bounded within a fixed range. Given the observations O, and the number of agents in the environment N, the problem is to estimate the network connectivity parameters A and D, subsequently referred to as θ. Finally, we assume that the probability of actually taking any existing edge in the environment is non-negligible.

a) How hard is this problem?: In principle, one might consider solving for the topology of the network by guessing the topology and then verifying that it was

consistent with the observed behavior. However, exhaustively evaluating every possible topology of a network with even a moderate number of nodes is intractable. A network of n nodes may have up to n^2 directed connections (including self-connected nodes). This gives us 2^{n^2} possible topologies. If we could evaluate 1 billion topologies per second, inferring the network transition probabilities using brute-force methods for even a 19-node graph would take substantially longer than the age of the universe as there are more than one googol (10^{100}) different instances to verify.

B. Monte Carlo Expectation Maximization

We approach the connectivity problem outlined above by attempting to simultaneously converge towards both the correct observation data correspondences and the correct network parameters through the use of the EM algorithm [19].

We iterate over the following two steps:

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

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

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:* which then updates our current parameter guess with a value that maximizes the expected log likelihood:

$$\theta^{(i)} = \underset{\theta}{\operatorname{argmax}}\, Q\big(\theta, \theta^{(i-1)}\big)$$

However, we employ the technique of MCEM to calculate the E-Step because of the intractability of summing over the high dimensional data correspondences. MCEM differs from traditional EM by executing the E-Step through a Monte-Carlo estimation process [22].

We approximate $Q\big(\theta, \theta^{(i-1)}\big)$ by drawing M samples of an ownership vector $L^{(m)} = \{l_i^m\}$ which uniquely assigns the agent i to the observation o_i in sample m. This is accomplished through a MCMC sampling technique which we will explain in the next section. We end up with

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

where $L^{(m)}$ is drawn using the previously estimated $\theta^{(i-1)}$.

At every iteration we obtain M samples of the ownership vector L, which are then used to re-estimate the connectivity parameter θ (the M-Step). This process is continued until we obtain a final estimate of θ. The pseudo code for the algorithm is shown in Algorithm 1.

At every iteration of the algorithm the likelihood of the ownership vector increases and the process is terminated when subsequent iterations result in very small changes to θ.

Algorithm 1 MCEM Inference Algorithm

```
LOOP:
  Draw sample L until p(L, O|θ) stops increasing
  IF p(L, O|θ) > CurrentLH
    CurrentLH = p(L, O|θ)
    Draw M samples L^(k)
    Update θ = given L^(1) ... L^(M)
  ELSE
    TERMINATE
  ENDIF
```

C. Sampling through Markov Chain Monte Carlo

We use Markov Chain Monte Carlo sampling to assign each of the observations to one of the agents, thereby breaking the multi-agent problem into multiple versions of a single-agent problem. The single agent case where $N = 1$ is relatively straightforward. In that 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 observation to a particular agent.

Given some guess of the connectivity parameter θ, we can obtain a likely data association L using the Metropolis algorithm; a popular method of MCMC sampling [25]. We implement the algorithm in the following manner. Given 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 the acceptance probability

$$\alpha = \min\left(1, \frac{p(L', Y|\theta)}{p(L, Y|\theta)}\right)$$

However, 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 (Figure 1). Consider L as a collection of ordered non-intersecting 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

$$p(T|\theta) = \prod_{w \in T} p(w|\theta)$$

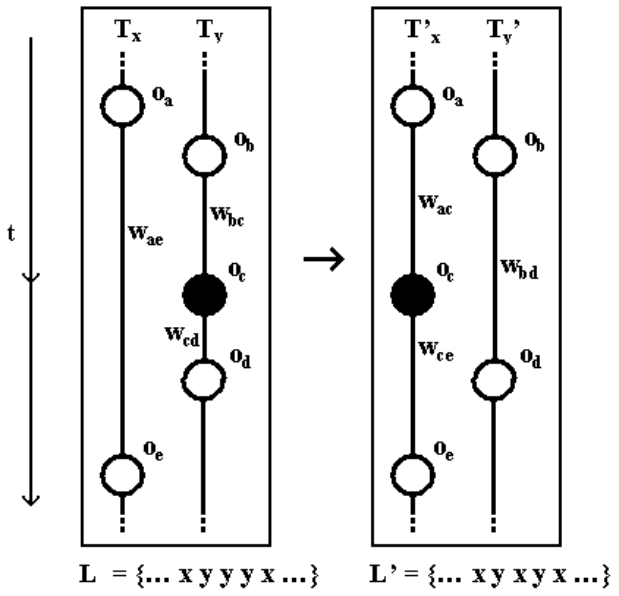


Fig. 1. An example of a proposed Markov Chain transition. The ownership assigned to o_c has been shifted from agent y to agent x. To evaluate this transition, the probability of the edge traversals w_{ac}, w_{ce}, w_{bd} must be compared to the original traversals w_{ae}, w_{bc}, w_{cd}.

and 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 the example shown in Figure 1:

$$\begin{aligned} \alpha &= \min\left(1, \frac{p(T'_x, T'_y|\theta)}{p(T_x, T_y|\theta)}\right) \\ &= \min\left(1, \frac{p(w_{ac}, w_{ce}, w_{bd}|\theta)}{p(w_{ae}, w_{bc}, w_{cd}|\theta)}\right) \end{aligned}$$

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 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 we can thus afford to test many proposals. The resulting chain is ergodic and reversible and thus produces samples representative of the true probability distribution.

IV. Methods

The approach has been examined with a number of experiments conducted in simulation. Our simulation tool takes as input the number of agents in the environment 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 uni-directional edges. The output is a list of observations generated by randomly walking the agents through the environment. Inter-node transit times are determined based on a normal distribution with a standard deviation equal to the square root of the mean transit time.[1]

A number of experiments were run using the simulator on randomly generated planar, connected graphs (Figure 2). The graphs were produced by selecting a sub-graph 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 in the following two ways: First, 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:

$$E_h = \frac{\sum_{a_{ij} \in A, a'_{ij} \in A'} \left(thr(a_{ij}) - thr(a'_{ij}) \right)^2}{m}$$

where $thr(a) = \lceil a_{ij} - \theta \rceil$.[2] Second, the absolute squared error between the true and inferred transition matrices was calculated:

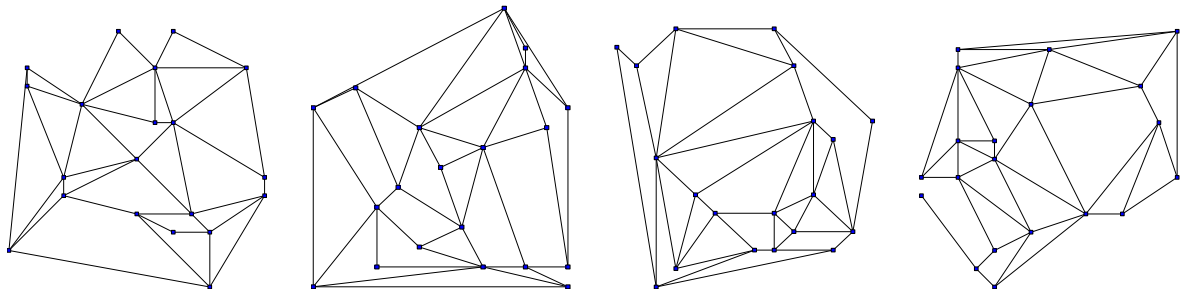
$$E_{sqr} = \sum_{a_{ij} \in A, a'_{ij} \in A'} \left(a_{ij} - a'_{ij} \right)^2$$

V. Results

The results show that problems involving a limited number of agents were easy to solve given an adequate number of observations (Figures 3, 4). For example, the topology of 95 *per cent* of the generated 12 node graphs was perfectly inferred with zero Hamming error for simulations with 4 agents. Generally, the algorithm converged quickly, finding most of the coarse structure in the first few iterations, and making incrementally smaller changes until convergence. Figure 5 shows as example of this process with a moderately sized graph. Although the majority of our investigation assumed no sensor error, tests confirmed that our method is also robust to missing and spurious observations (Figure 6).

[1] Negative transit times are rejected.

[2] A threshold value of $\theta = 0.1$ was selected for our experiments.



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

Fig. 2. Examples of randomly created 20 node, 80 directed edge graphs.

In most cases the thresholding method presented in [10] did not give results as accurate as our method. Figure 8 shows a comparison of an implementation of their method with our MCEM approach. Although shown to be less accurate in our simulations, this thresholding technique is very fast and does not rely on knowing the number of agents in the system.

A critical parameter is the number of agents moving in the system relative to the number of vertices. Clearly, if there is only one agent in the network the problem is trivial since (ignoring detection errors) its event sequence can simply be "traced out". However, in the case of multiple agents, the events generated by a given agent's movements in the network risk being incorrectly associated with those of any other agents'. It is the relative density of the correct pairings relative to the incorrect ones that makes the problem more or less easy to solve.

Increasing either the number of agents present in the environment or the size of the graph made the problem more difficult to solve, albeit for rather different reasons. While increasing the number of agents allowed a greater number of probable trajectories, and was analogous to decreasing the signal to noise ratio in the system, increasing the graph size while holding the number of observations steady reduced the expected number of observations per edge in the graph. Experiments support the idea that the accuracy of our approach for a particular number of agents seems to depend on the ratio of observations to edges (Figure 7).

In the extreme case, if there are some edges that have no observations recorded along them at all, our approach will not have enough information to infer the correct graph. At the minimum, an observed agent must traverse each edge at least once. Hence, we are effected by the *edge* cover time of the graph which refers to the expected amount of time it takes a random walk to visit every edge.

It is possible to determine a theoretical upper bound on the edge cover time for connected, planar graphs. It is known that a random walk on a connected planar graph has an expected vertex cover time that is bounded by $6n^2$ where n is the number of vertices in the graph [26]. If the graph has a maximum degree of k, then we have an expected time of covering all the edges leading from the maximum degree node of

$$\begin{aligned} E\{T_c\} &= k\sum_{j=1}^{k}\frac{1}{j} \\ &= kH_k \end{aligned}$$

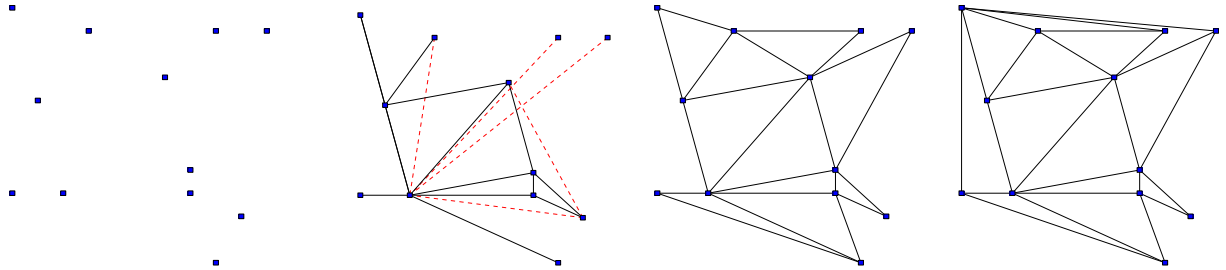
and since the *ith* harmonic number H_i is bounded above by $\log(i)+1$, we end up with an upper bound on the expected edge cover time of $6n^2(k\log(k)+k)$. An alternative bound, $6(7n-12)^2$ arrived at though Euler's formula might be tighter for large values of k.

As expected, our results indicate that the error rate flattens out well before this bound. For example, using a value of $n-1$ for k on a 12 node graph suggests that the edge cover time will be bounded by an observation to edge ratio of over 600. In fact Figure 7 shows that Hamming error approaches zero for a 12 node graph with 4 agents after a observation to edge ratio of only one tenth of this value.

VI. Conclusions and Future Work

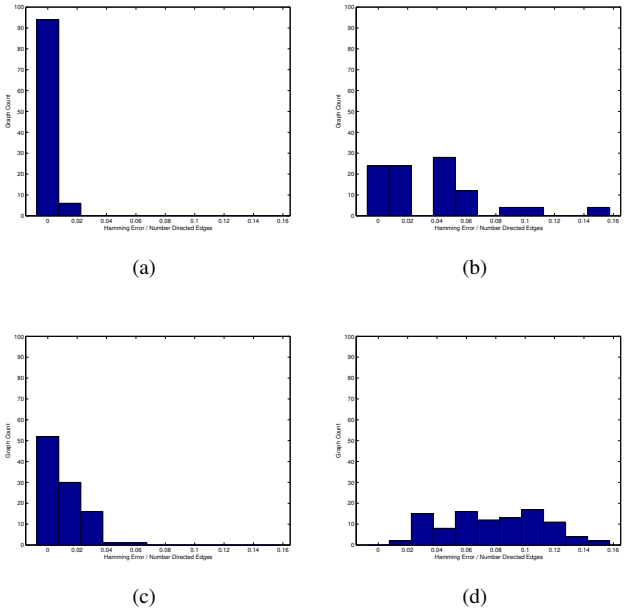
We have described a MCEM algorithm for learning connectivity information of an environment embedded with a sensor network and we have demonstrated the effectiveness of this approach with a number of experiments based on simulations. As well as developing and demonstrating a useful technique with some practical applications, we have provided additional evidence of the effectiveness of MCEM based algorithms to find near optimal solutions in a high dimensionality state space.

There are a number of areas in which we intend to conduct further work. First, we would like to infer the number of people (agents) moving in the system, since this should broaden the practical applications of the approach. The approach we describe appears to be robust to variations in the number of people in the system so long as we overestimate this parameter in the modeling stage. It appears



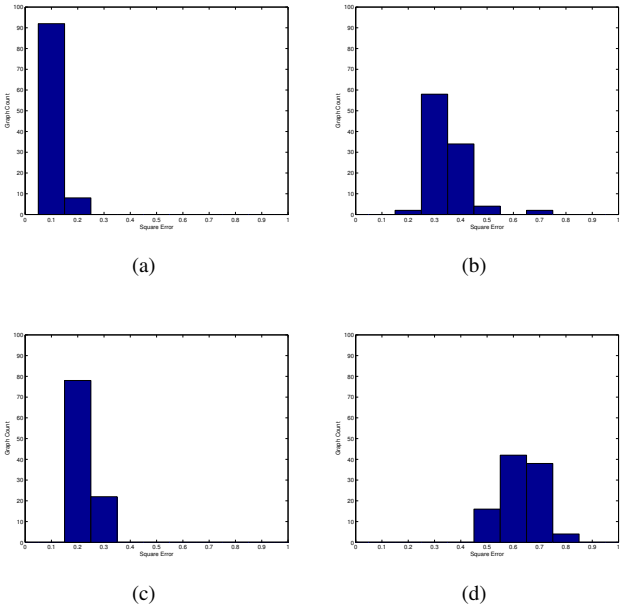
(a) (b) (c) (d)

Fig. 5. Incremental Belief of the topology of a 12 node graph using the simulator with 8000 observations: a) initially b) after 1 iteration, c) after 2 iterations, d) after 3 iterations (the true graph). Dotted lines indicate incorrect transitions.



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

Fig. 3. Histogram of Hamming error per edge using the simulator with 8000 observations on 100 randomly produced graphs: a) 12 nodes and 4 agents, b) 12 nodes and 10 agents, c) 20 nodes and 4 agents d) 20 nodes and 10 agents



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

Fig. 4. Histogram of squared error using the simulator with 8000 observations on 100 randomly produced graphs: a) 12 nodes and 4 agents, b) 12 nodes and 10 agents, c) 20 nodes and 4 agents d) 20 nodes and 10 agents

we can also explicitly model fluctuations in the number of people using a source or sink node, but this is outside the scope of this paper. Finally, we are in the process of testing the algorithm on a real sensor network. An attempt to illustrate the technique using hardware will bring up issues such as tolerance to sensor noise, and the possible use and incorporation into the algorithm of agents' signatures.

REFERENCES

[1] N. Bulusu, D. Estrin, L. Girod, and J. Heidemann, "Scalable coordination for wireless sensor networks: self-configuring localization systems," in *Sixth International Symposium on Communication Theory and Applications (ISCTA-01)*, Ambleside, Lake District, UK, July 2001.

[2] N. Correal and N. Patwari, "Wireless sensor networks: Challenges and opportunities," in *MPRG/Virgina Tech Wireless Symposium*, 2001. [Online]. Available: citeseer.ist.psu.edu/correal01wireless.html

[3] S. Capkun, M. Hamdi, and J.-P. Hubaux, "GPS-free positioning in mobile ad-hoc networks," in *HICSS*, 2001. [Online]. Available: citeseer.ist.psu.edu/capkun01gpsfree.html

[4] N. Patwari, A. Hero, M. Perkins, N. Correal, and R. O'Dea, "Relative location estimation in wireless sensor networks," in *IEEE Transactions on Signal Processing*, ser. 8, vol. 51, Aug 2003, pp. 2137–2148.

[5] Y. Xu, J. Heidemann, and D. Estrin, "Geography-informed energy conservation for ad hoc routing," in *ACM/IEEE International Conference on Mobile Computing and Networking*. Rome, Italy: USC/Information Sciences Institute, July 2001, pp. 70–84.

[6] M. Batalin, G. Sukhatme, and M. Hattig, "Mobile robot navigation using a sensor network," in *IEEE International Conference*

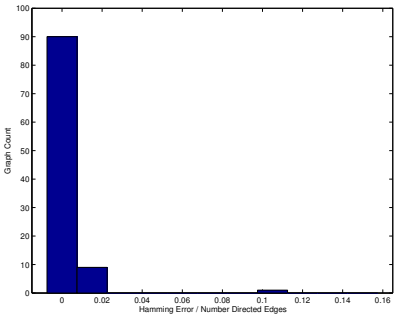


Fig. 6. Histogram of Hamming error per edge using the simulator on 8000 observations with added error on 100 randomly produced 12 node graphs with 4 agents. Error was modeled by removing 5 percent of the correct observations and adding 5 percent spurious observations.

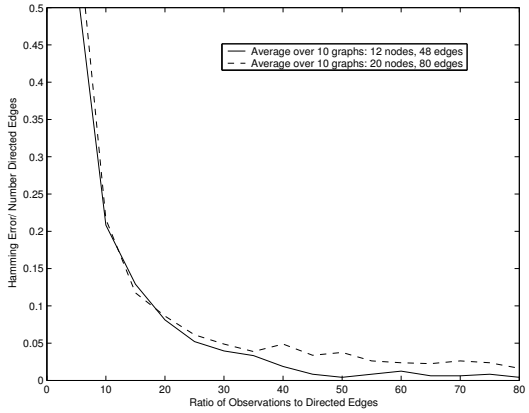


Fig. 7. Hamming error per edge as a function of the ratio of observations to true (directed) edges using the simulator with 4 agents.

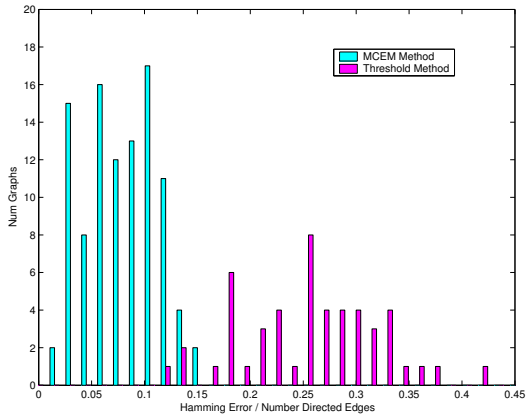


Fig. 8. Histogram of Hamming error per edge using both the threshold method described by Ellis, Makris and Black in [10] and our MCEM method with 8000 observations produced by 10 agents on 100 randomly produced 20 node graphs.

on Robotics and Automation, vol. 1, New Orleans, Louisiana, April 26-May 1 2004, pp. 636 – 641. [Online]. Available: http://cres.usc.edu/cgi-bin/print_pub_details.pl?pubid=367

[7] S. Poduri and G. S. Sukhatme, "Constrained coverage for mobile sensor networks," in *IEEE International Conference on Robotics and Automation*, New Orleans, LA, May 2004, pp. 165–172. [Online]. Available: http://cres.usc.edu/cgi-bin/print_pub_details.pl?pubid=409

[8] M. Batalin and G. S. Sukhatme, "Coverage, exploration and deployment by a mobile robot and communication network," *Telecommunication Systems Journal, Special Issue on Wireless Sensor Networks*, vol. 26, no. 2, pp. 181–196, 2004. [Online]. Available: http://cres.usc.edu/cgi-bin/print_pub_details.pl?pubid=369

[9] O. Javed, Z. Rasheed, K. Shafique, and M. Shan, "Tracking across multiple cameras with disjoint views," in *The Ninth IEEE International Conference on Computer Vision*, Nice, France, 2003.

[10] T. Ellis, D. Makris, and J. Black, "Learning a multicamera topology," in *Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, Nice, France, October 2003, pp. 165–171.

[11] D. Makris, T. Ellis, and J. Black, "Bridging the gaps between cameras," in *IEEE Conference on Computer Vision and Pattern Recognition CVPR 2004*, Washington DC, June 2004.

[12] Y. Bar-Shalom, Ed., *Multitarget multisensor tracking: advanced applications*. Artech House, 1992, vol. II.

[13] J. Liu, J. Liu, J. Reich, P. Cheung, and F. Zhao, "Distributed group management for track initiation and maintenance in target localization applications," in *Proceedings of 2nd International Workshop on Information Processing in Sensor Networks (IPSN)*, April 2003.

[14] M. Rosencrantz, G. Gordon, and S. Thrun, "Locating moving entities in indoor environments with teams of mobile robots," in *Second international joint conference on Autonomous agents and multiagent systems*, Melbourne, Australia, 2003, pp. 233–240.

[15] J. Shin, L. J. Guibas, and F. Zhao, "A distributed algorithm for managing multi-target identities in wireless ad-hoc sensor networks," in *Information Processing in Sensor Networks Second International Workshop, IPSN 2003*, Palo Alto, CA, April 2003, pp. 223–238.

[16] C. Rasmussen and G. Hager, "Probabilistic data association methods for tracking multiple and compound visual objects," in *IEEE Trans. Pattern Analysis and Machine Intelligence*, 2001.

[17] T. Yu and Y. Wu, "Collaborative tracking of multiple targets," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR 2004)*, Washington, DC, June 2004.

[18] H. Pasula, S. Russell, M. Ostland, and Y. Ritov, "Tracking many objects with many sensors," in *IJCAI-99*, Stockholm, 1999.

[19] A. Dempster, N. Laird, and D. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society*, 1977.

[20] S. Thrun, D. Fox, and W. Burgard, "A probabilistic approach to concurrent mapping and localization for mobile robots," *Machine Learning and Autonomous Robots (joint issue)*, 1998.

[21] J. Vermaak, N. Lawrence, and P. Prez, "Variational inference for visual tracking," in *Computer Vision and Pattern Recognition*. IEEE Computer Society Press, 2003.

[22] G. Wei and M. Tanner, "A monte-carlo implementation of the EM algorithm and the poor man's data augmentation algorithms," *Journal of the American Statistical Association*, vol. 85(411), pp. 699–704, 1990.

[23] F. Dellaert, S. Seitz, C. Thorpe, and S. Thurn, "EM, MCMC, and chain flipping for structure from motion with unknown correspondence," *Machine Learning, special issue on Markov chain Monte Carlo method*, vol. 5, pp. 45–7, 2003.

[24] D. Tregouet, S. Escolano, L. Tiret, A. Mallet, and J. L. Golmard, "A new algorithm for haplotype-based association analysis:the stochastic-EM algorithm," *Annals of Human Genetics*, vol. 68, no. 165, March 2004.

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

[26] J. Jonasson and O. Schramn, "On the cover time of planar graphs," in *Electronic Communications in Probability*, 2000, pp. 85–90.