Inferring a Probability Distribution Function for the Pose of a Sensor Network using a Mobile Robot

David Meger, Dimitri Marinakis, Ioannis Rekleitis, and Gregory Dudek

***Abstract*— In this paper we present an approach for localizing a sensor network augmented with a mobile robot which is capable of providing inter-sensor pose estimates through its odometry measurements. We present a stochastic algorithm that samples efficiently from the PDF for the pose of the sensor network by employing Rao-Blackwellization and a proposal scheme which exploits the sequential nature of odometry measurements. Our algorithm automatically tunes itself to the problem instance and includes a principled stopping mechanism based on convergence analysis. We demonstrate the favourable performance of our approach compared to that of established methods *via* simulations and experiments on hardware.**

I. INTRODUCTION

In this paper we present an approach for probabilistically localizing a network made up of both static and mobile components based on relative inter-component pose estimates. This problem of self-localization in sensor networks is recognized as a key requirement for many network applications [1] and is considered an important step in the overall goal of developing self-adapting and self-configuring networks.

We consider the case where our network is augmented with a mobile robot which is capable of providing inter-sensor pose estimates through odometry measurements. In this scenario, the robot's motion through the network facilitates localization by explicitly transferring positional information between sensor locations. By maintaining an ongoing estimate of the robot's location, the position of any sensor it interacts with can be probabilistically estimated, (and updated), given the appropriate motion and measurement models.

Our approach is to use a Markov Chain Monte Carlo (MCMC)-based algorithm that allows us to sample from the probability distribution function (PDF) for the pose of a sensor network. We overcome the often-prohibitive computational effort required by MCMC approaches by employing the following techniques: 1) we use a unique, odometry-specific proposal distribution that exploits the conditional dependencies present in our problem domain; 2) we apply Rao-Blackwellization to effectively reduce the dimensionality of our sample space; 3) we automatically tune the parameters of our proposal technique to achieve desired acceptance rates; and 4) we employ convergence analysis based on the Gelman-Rubin statistic [2] as a stopping mechanism that informs us when the samples we have gathered closely represent the underlying pose distribution. Note that few existing methods for sensor network localization compute the full probability distribution for the node positions, due to the computational burden it entails. Existing approaches to sensor network localization and related problems generally estimate only the maximum likelihood configuration and use a Gaussian assumption in the computation of confidence intervals.

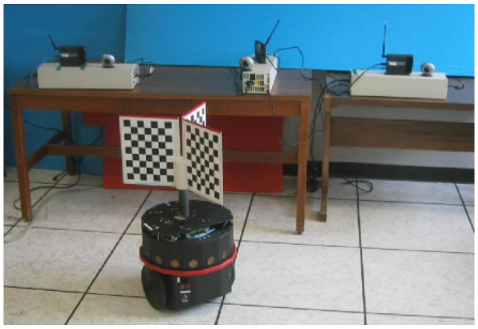


Fig. 1. The components of the camera sensor network used in the experimental results of the paper.

The problem we consider in this work is similar to the simultaneous localization and mapping (SLAM) issue in traditional mobile robot research, but there are some key differences. Our sensors, which correspond to landmarks in the SLAM problem, are uniquely identifiable, so there is no correspondence ambiguity. Additionally, we can assume that the mobile robot will operate, for the most part, within the confines of a sensor-network deployed region and will ultimately visit the local area of each network component many times. Most importantly, in sensor network self-localization, the initial mapping effort is a one-time task, the results of which will likely be used for the lifetime of the network. Therefore the use of computationally sophisticated techniques are appropriate.

One potential application of our work is in the construction of a *smart house* in which actuators and sensors respond to the behaviour of the resident in order to help them with their daily activities [3]. For example, such a technology could allow aging seniors more independence and therefore a better quality of life. Furthermore, it could allow them to stay longer in their own residence than might otherwise be possible, alleviating pressure on public institutions. A mobile robot combined with a sensor network in such a system could aid the resident in many tasks, but must deal with navigation and localization.

D. Meger is with the department of Computer Science, University of British Columbia, Vancouver, British Columbia, Canada, dpmeger@cs.ubc.ca

D. Marinakis, I. Rekleitis, and G. Dudek are members of the Centre for Intelligent Machines, McGill University, Montreal, Quebec, Canada, {dmarinak,yiannis,dudek}@cim.mcgill.ca

All the authors worked at the Centre for Intelligent Machines, McGill University during this research.

The contribution of this work is an efficient, MCMC-based, global localization technique to directly recover a representation for the PDF for the pose of a mobile-robot augmented sensor network. Our approach uses a novel, self-tuning proposal method in conjunction with Rao-Blackwellization to achieve fast mixing rates and employs a principled stopping mechanism that detects when our approach has adequately characterized the underlying distribution.

In the remainder of this paper, we first provide some background on related work and then give a formal definition of the problem we are addressing. We then discuss the details of our approach to sensor network localization and assess its performance against those of established methods *via* simulations and real world experiments.

II. Related Work

Traditional sensor network self-localization efforts have focused on estimating distances between sensors. Techniques include the use of the received communication signal strength in radio networks [4], or time-of-arrival ranging using ultrasound [5]. These techniques typically have limited accuracy and the localization algorithms must be able to handle some degree of noise in the range data [6]. While much of the research conducted on sensor networks is based on developing distributed, computationally efficient algorithms appropriate for networks of low-powered sensor platforms, recently there has been a shift towards more complex approaches incorporating advanced probabilistic techniques and graphical models, *e.g.* [7]. The traditional sensor network assumption of homogenous systems of resource-impoverished nodes is making way for tiered architectures that incorporate network components of significant computational sophistication [8].

Complex probabilistic approaches are especially appropriate when variants of the localization problem are considered in which the network includes one or more mobile components. In these cases, as we have mentioned earlier, the resulting problem formulation often bears many similarities to those posed in the framework of the SLAM problem. The extended Kalman filter, as pioneered by Smith *et al.* [9], is a common approach to SLAM. Sampling-based methods have also been considered, such as FastSLAM [10]. However, several authors have noted that the filtering approach, which maintains only the most recent pose of the mobile agent, is prone to errors, and have instead estimated the entire set of previous poses. For example, Dellaert and Kaess [11], [12], apply many of the Kalman filter assumptions in the context of smoothing rather than filtering.

Other methods employ hybrid online and offline, global correction methods to the SLAM problem. Scan-Matching for Alignment [13] and its later practical implementation, known as Local Registration and Global Correlation, [14] are two examples of such approaches.

Recently there have been a few efforts that have employed a global approach to estimating the distribution for sensor network locations using computationally expensive techniques [15], [16]. These efforts return a PDF which inherently includes confidence estimates for the pose estimates, but they are not necessarily practical for networks actuated by a mobile robot. The use of a robot's motion to facilitate localization has been recently examined, *e.g.* [17], however, the focus has been on real-time, filtering techniques.

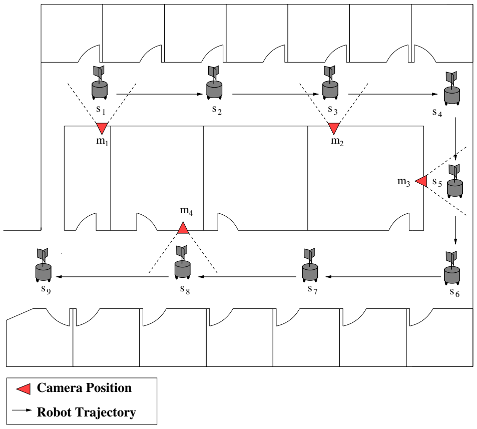


Fig. 2. The calibration and mapping scenario described in this paper. The robot moves through the environment, calibrating each sensor and estimating both its own position as well as the positions of each sensor in a common coordinate frame.

Some closely related work has recently presented the incorporation of MCMC-based approaches to mobile robot aided sensor network self localization [18]. This work focused on the use of MCMC as a corrective tool for faster, filtering-based inference techniques. The approach can be considered a variant of single-component Metropolis-Hastings, a common technique for constructing a Markov chain in high dimensional state spaces [19]. Although additional methods are used to improve the mixing rate of the chain, the algorithm presented in [18] is too expensive computationally to be generally applicable and is designed to be used as a complement to other methods.

III. Problem Definition

Sensor network localization means estimating the pose of each sensor node m_i, to build a map $m^n = [m_1 m_2 ... m_n]$. We assume non-overlapping sensor fields, which means that a mobile agent is required to move between sensors within the network to estimate their relative locations; see Figure 2. Node positions can be measured relative to the position of the robot at a given time, s_t, which is the most recent component of the robot's path, $s^t = [s_1 s_2 ... s_t]$, and so both quantities must be estimated simultaneously. The measurements available are the position of a sensor relative to the robot at time t, denoted z_t and the position of the robot at time t relative to its position at time $t-1$, denoted u_t.

This problem can be modeled as the probabilistic inference of the map and the robot poses conditioned on the observations, as represented by the underlying directed graphical model. The posterior distribution, $p(m^n, s^t | z^t, u^t)$, can be factored into the product of many local conditional distributions, by exploiting the conditional independencies as is common practice for probabilistic graphical models.

For the sensor network localization problem involving a mobile agent, there are two classes of local conditional likelihoods: $p(s_t|u_t, s_{t-1})$ which is known as the motion, or odometry model of the robot; and $p(z_t|s_t, m_i)$, the measurement model which relates the poses of the sensor nodes to that of the robot. These two distributions can be determined empirically or by physical modeling for a particular instantiation. In the remainder of the paper we will present an efficient MCMC global inference technique that directly samples from the posterior distribution, $p(m^n, s^t|z^t, u^t)$, based on models of these two distributions.

IV. Probabilistic Sensor Network Self-Localization using MCMC

Our global inference approach uses MCMC to generate a number of samples for sensor node locations according to our probability model. These are then used as a particle-based representation of the underlying probability distribution function. Conceptually, we form a graph $< V, E >$, where V is the set of vertices and E the set of connecting edges. The vertices of this graph are the robot positions over time s^t and the sensor locations m^n. The edges, or constraints, are the odometry measurements u^t connecting consecutive robot positions and the measured relative positions z^t between the robot poses and network components. Using a model characterizing the error in the measurements, we can calculate the density of any particular configuration $x = (m^n, s^t)$ through the application of Bayes law:

$$p(x|z^t, u^t) = \frac{p(z^t, u^t|x)p(x)}{p(z^t, u^t)} \quad (1)$$

$$p(x|z^t, u^t) \propto p(z^t, u^t|x)p(x) \quad (2)$$

We assume that the prior, $p(x) = p(m^n, s^t)$, is uniform, so the relative likelihood of a particular configuration can be evaluated by factoring $p(z^t, u^t|x)$, into the product of the likelihoods of all constraints (edges) given our motion and measurement models. In this manner we can evaluate the relative density of our target distribution given a configuration:

$$\begin{aligned}\pi(x) &= p(z^t, u^t|x) &(3)\\ &= \prod_k p(z_k|x) \prod_k p(u_k|x) &(4)\\ &= \prod_k p(z_k|s_k, \theta_k) \prod_k p(u_k|s_k, s_{k-1}) &(5)\end{aligned}$$

where θ_k indicates the sensor node $m_i, i \in \{1 : n\}$ observed by (or observing) the robot at time step k.

Given this ability to calculate the relative density of our target distribution at a specific point, we can employ the well known Metropolis-Hastings (M-H) algorithm [20] to generate representative samples. M-H constructs a Markov chain by accepting proposed transitions from the current state x_i to a new state x_j based on their relative likelihoods and that of the proposal function. In theory this approach can be used to characterize any distribution given only the ability to calculate the target density and a reasonable proposal scheme. In our application of the M-H algorithm to sensor network localization, we use a proposal function $Q(x'|x)$, which we will define below, that generates a new state x' given the current state x. The proposal x' is then either accepted or rejected with probability α, where α is calculated as:

$$\alpha = \min\left(1, \frac{\pi(x')Q(x|x')}{\pi(x)Q(x'|x)}\right) \quad (6)$$

A. Odometry-Specific Proposal Scheme

In order to improve the rate at which the Markov chain approaches an equilibrium distribution, *i.e.* the mixing rate, we employ a proposal scheme which exploits domain knowledge regarding the sequential nature of our odometry measurements. We use the information that a change in position early in the odometry path of the robot effects its position from that point forward in time. To model this behaviour, the current state $x = (s^t, m^n)$ in the chain is altered to produce a new proposed state x' through the following procedure:

1) A pose s_j is selected (as described shortly).
2) The initial $j - 1$ robot poses, $[s_1, \ldots, s_{j-1}]$, are kept the same as in x.
3) The position of robot pose s_j is altered by the addition of zero-mean, normally distributed noise with a covariance Σ_j.
4) The effect of the change in s_j is propagated forward to change the locations of all following robot poses, $[s_{j+1}, \ldots, s_t]$. That is, the successive odometry constraints are kept rigid.

The above steps are repeated as new samples are required. In order to obtain a balanced sampling, iterated rounds of random selection without replacement are performed to select poses. This proposal scheme temporarily blocks correlated components of the state space together during proposals in order to facilitate more rapid mixing. Blocking is not an uncommon technique when correlation is present among features in the target distribution and has been shown effective in the past; *e.g.* see [21].

While this proposal method applies to the odometry portion of our state space, there remains the component made up of the sensor positions. In the next section, we will describe the Rao-Blackwellization (RB) process that approximately marginalizes out this factor of the joint distribution allowing for a further improvement in mixing. Instead of using RB, however, one may alternate the proposal scheme described above for the robot poses s^t with one that proposes alterations individually to each of the sensor positions $m_1, \ldots, m_n$ in turn. Here the ordering in which the proposals to the sensors are made will not effect the outcome since the processes are independent. In the next paragraph, we will briefly describe such a proposal scheme for the sensors m^n as it will aid the description of the RB step.

For sensor node m_i, we generate a proposal distribution $Q_i(x'|x)$ that is based on constraints existing between sensor m_i and any of the robot poses. Specifically, let θ_k indicate the sensor node $m_i, i \in \{1 : n\}$ observed by (or observing) the robot at time step k , as defined earlier. Now let S_i represent the set of those poses such that if $s_k \in S_i$ then $\theta_k = m_i$. Now let Z_i represent the corresponding set of constraints (or measurements) providing pose estimates between sensor m_i and each of the robot poses $s \in S_i$. A linear approximation is then applied to the measurement model, yielding a separate Gaussian distribution for m_i given each pose $s \in S_i$ and its corresponding measurement $z \in Z_i$. The product of these separate distributions is calculated and used as the proposal distribution Q_i for the pose of m_i. A sample is then drawn: $(x', y', \theta') \sim Q_i$ as a potential new location for m_i and is accepted or not based on the equation:

$$\alpha = \min\left(1, \frac{p(m_i'|S_i, Z_i)Q_i(m_i|S_i, Z_i)}{p(m_i|S_i, Z_i)Q_i(m_i'|S_i, Z_i)}\right) \quad (7)$$

(from Equation 6) where m_i' represents the newly proposed location for m_i. Note that in the case that no approximation is necessary in this step, for example if the measurement model is already Gaussian, then the acceptance ratio is always one and this mixing approach is equivalent to a Gibbs-style proposal.

B. Rao-Blackwellization

Rao-Blackwellization is a powerful technique for state estimation, as demonstrated by Montemerlo *et al.* for the SLAM problem [10]. By applying a similar approach to network localization, we can greatly accelerate the mixing rate of our chain. Although our method is only guaranteed to produce exactly representative samples for certain classes of models, a good approximation to the real distribution is obtained in most circumstances, and the results considerably exceed the efforts achieved by standard filtering techniques under the same circumstances.

Instead of sampling from $p(s^t, m^n|z^t, u^t)$ directly, we sample from the factor $p(s^t|z^t, u^t)$. This is accomplished by approximating $p(m^n|s^t, z^t, u^t)$ with a closed form formula and marginalizing this factor out. We use a product of Gaussians similar to the proposal function Q described in the previous section. The product of Gaussians is obtained by linearizing the measurement model, yielding separate distributions for m_i given each pose $s \in S_i$ and its corresponding measurement $z \in Z_i$. We then take the product of these separate distributions, q_i, as an approximation to $p(m_i|s^t, z^t, u^t)$. We can now approximate $p(s^t|z^t, u^t)$ as follows:

$$\begin{aligned} p(s^t|u^t, z^t) &= \int p(s^t|u^t)p(m^n|s^t, u^t, z^t)dm^n \quad (8) \\ &= p(s^t|u^t)\prod_{i=1}^{n}\int p(m_i|s^t, u^t, z^t)dm_i \\ &\approx p(s^t|u^t)\prod_{i=1}^{n}\int q_i(m_i|S_i, Z_i)dm_i \end{aligned}$$

where S_i and Z_i are as defined in the previous section. Given samples drawn from the approximation of $p(s^t, |z^t, u^t)$, we can characterize the target distribution $p(s^t, m^n|z^t, u^t)$ using either the approximation to $p(m^n|s^t, z^t, u^t)$, or the real distribution; the second requiring the use of a technique such as importance sampling. This process of improving the accuracy of an estimator by marginalizing out variables is a technique referred to as Rao-Blackwellization [22].

C. Automatic Tuning

In keeping with accepted practice, we ensure adequate mixing by running the chain for an initial tuning period during which the proposal parameter values for each component of the state space, (specifically, each robot pose s_t), are automatically adjusted to approach a desired mixing ratio, L. This tuning period is divided into a number of smaller time windows $[t_1, t_2 \ldots]$ in which the chain is run for some fixed number of proposals. The proportion of proposals accepted for each component, j, is then calculated for the current time window t, and this value is compared to the target mixing ratio L and adjusted accordingly using an exponential averaging scheme for use in the next time window.

D. Stopping Mechanism

One of the key issues when employing MCMC is determining how long it takes for the chain to approach the target distribution. In practice, the initial portion of the chain is discarded to reduce the correlation of subsequent samples with the starting point and to allow the chain to move into high likelihood, representative configurations. This is known as the burn-in period. Typically, after the burn-in period, samples are drawn periodically from the chain, with a fixed number of proposals between each sample. Given adequate mixing, these samples are then taken as representative of the target distribution.

As an indicator of convergence our approach employs the Gelman-Rubin statistic [2], which is based on a variance analysis of instances of the chain restarted from different initial positions, (which should be over-dispersed with respect to the target distribution). The resulting value, calculated for a single feature, is referred to as a *potential scale reduction factor* (PSRF) and suggests how the estimated variance for the feature under consideration could be improved by additional simulations. The key idea is that if the system has converged, then the samples should exhibit a large degree of agreement regarding the statistics of the problem. Essentially, a PSRF value near one suggests suggests that each of the restarts obtained samples that share similar characteristics and therefore are presumably close to the target distribution. On the other hand, a PSRF value far from one suggests that a full tour of the target distribution has not been obtained. A $PSRF < 1.2$ is sometimes used as a guideline for "approximate convergence" [23].

When attempting to obtain a particle representation for the PDF of the network pose, our technique employs a number of instances of the algorithm described above running in parallel. Each separate instance starts from a different initial configuration and runs independently. After each instance has run for some set number of proposals, we calculate the Gelman-Rubin statistic for a number of indicator features, namely the X and Y co-ordinates of the sensor positions.

We employ the maximum $PSRF$ value obtained from these indicator features as a metric suggesting convergence. Under normal operation of our algorithm, if the calculated value of this metric falls below a threshold value, (*e.g.* 1.2), then simulations are halted, and the samples from each of the parallel chains are combined as the output. Otherwise we resume the simulation, and continue to obtain samples until our metric suggests that we are near convergence. For the sake of convenience and brevity, in the remainder of this paper we will refer to the parallel instances of the algorithm, described above, as *restarts* and our calculated convergence metric as the *PSRF* obtained.

V. Results from Simulations

We investigated the performance of our localization algorithms on data obtained from a realistic simulator. Using this simulator we found evidence demonstrating the superior ability of our MCMC-based localization algorithm to accurately represent network pose distributions in comparison to two other standard state estimation techniques: the Rao-Blackwellized particle filter (RBPF), and the extended Kalman filter (EKF).

Our simulator places N sensor nodes on a two-dimensional plane with a uniform distribution. These nodes are connected *via* potential pathways by selecting a subgraph of the Delaunay triangulation. The motion of a robot is then simulated through this environment as T distinct steps made from the region of one sensor to another. To choose a destination sensor, a quasi-random walk strategy is employed biased towards visiting new nodes. That is, the robot first selects uniformly at random from the *un-visited* neighbours of its current sensor. If there are no un-visited neighbours, a neighbor is re-visited at random.

At each step t, the robot first executes a rotation and then performs a translation in the new direction. These two motions are captured as the odometry measurement u_t. At the end of the translation, a new measurement z_t is obtained from the sensor observing the region occupied by the robot. For these experiments, zero-mean, normally distributed noise is added to the odometry measurements for each of the rotation and translation motions, and also for the measurement. We assume known mean μ and covariance Σ for each of the noise signals added to our measurements.

In order to provide benchmarks with which to compare the performance of our localization algorithm, we implemented for comparison purposes, two popular Bayesian filtering approaches: a Rao-Blackwellized particle filter (RBPF), see [24] [25], and also an Extended Kalman Filter (EKF), see [26].

Two different RBPF implementations were used: a 'basic' variant that used the true motion model as the proposal distribution and also a variant that linearized this two step motion (rotation and translation) and incorporated the most recent evidence (also linearized) into a closed form proposal. For both versions, the sensor node distributions were maintained internal to each particle as Gaussians. A full discussion of the comparative performance of these two RBPF variants is outside the scope of this paper. Where particle-filter results are reported, we present data from the variant with the best performance.

MCMC Within-Cloud $M = 5$, $N = 480$ (10 Comparisons)	$\mu = 15.21$ $\sigma = 0.68$	
Algorithm	Hausdorf Distance D_h	Normalized Hausdorf $\|D_h - \mu\|/\sigma$
RBPF ($K = 20000$)	15.50	1.85
RBPF ($K = 10000$)	18.69	6.55
RBPF ($K = 5000$)	22.92	12.74
RBPF ($K = 1000$)	43.75	43.22
EKF	73.05	86.07

TABLE I

Comparison of different algorithms using the Hausdorff distance metric demonstrating improvement with increasing particles for simulator data. 4 step path. 3 sensor network.

Our simulation results indicate that our technique outperforms both the EKF and the RBPF at the task of inferring a network pose distribution. Figure 3 shows the results obtained from the different inference algorithms on the same simulation data for a small scale version of the localization problem under relatively noisy conditions.[1] For the MCMC approach, each restart was initialized with values obtained by running the RBPF with only enough particles to maintain a non-zero probability configuration. As described in section IV, all of the restarts were run until the set of samples produced by each instance had similar statistical characteristics. Although it provides no guarantees, this analysis represents strong evidence for the convergence of the employed chain in this problem instance and suggests that the results obtained should closely reflect the actual distribution suggested by the data and models used by our simulator.

It can be observed qualitatively from Figure 3 that the RBPF when used with $K = 5000$ particles produces a similar distribution to the MCMC algorithm, although the samples are not as homogeneously distributed. [2] Linearization approximations made by the EKF along with its limited expressiveness reduce its accuracy and hence its output is the most different from the MCMC result.

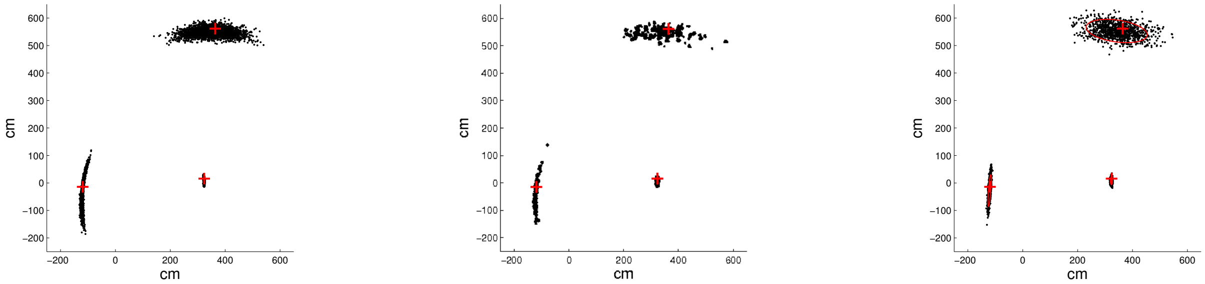
In order to quantitatively compare the distributions obtained from the different approaches, we employed the generalized Hausdorff distance:

$$D_h = k\text{th} \min_{a \in A, b \in B}(||a - b||)$$

where $||a - b||$ is calculated using a L2-norm and the kth largest value is selected based on the $95th$ quantile. In order to correctly interpret a value obtained from this metric we divide the control particle cloud into M disjoint sets of samples, each of size N. We then calculate the mean and standard deviation of the D_h value found between each pair

[1] We used 5 *per cent* rotational and translational error in the motion model, and similar values for the measurement model.

[2] As a final step, in both our RBPF implementation and the MCMC algorithm, the sensor locations are sampled from the closed form approximation to their distribution given the robot poses. This is done for each sample/particle obtained.



(a)

(b)

(c)

Fig. 3. Results obtained on data obtained from the simulator for a robot path of 4 steps through a 3 sensor network for the algorithms: a.) MCMC b.) RBPF ($K = 5000$), and c.) EKF. The crosses indicate the ground truth sensor positions. For the EKF, the samples are drawn from the mean and covariance obtained for the position of the sensors; a three standard deviation uncertainty ellipse is overlaid.

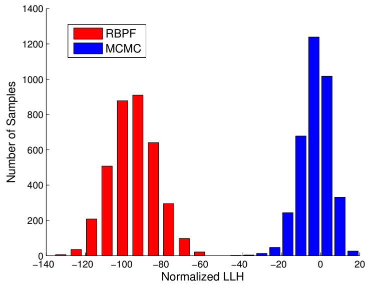


Fig. 5. Histogram comparing the relative log likelihoods (LLH) of the final configuration samples obtained from the MCMC and RBPF ($k = 20000$) techniques for the simulation result shown in Figure 4. The likelihoods were normalized such that ground truth had a log likelihood of zero.

of sample sets. When interpreting a distance value found between the control set and a comparison set, each of size N, we can measure the number of standard deviations between the new distance and the previously computed control value.

Table I shows the distance metric values obtained when the particle clouds from the different algorithms are compared to the MCMC result; the same experiment presented in Figure 3. It can be seen that while the PDF suggested by the EKF is significantly different from the MCMC result, the performance of the RBPF improves as a function of the number of particles used. For this size of a problem, the data obtained from the RBPF with 20,000 particles is not significantly different from that of the MCMC technique.

As the scale of the problem increases, however, it becomes increasingly difficult for the filtering techniques to accurately characterize the distribution. Figure 4 shows an example of results obtained from the different inference algorithms on a moderately sized problem in which the robot visits each of the sensors a number of times. Interestingly, while both the EKF and the RBPF provide good estimates of the maximum likelihood location for the sensors, their uncertainty estimates are extremely poor in comparison to the MCMC approach which, we argue, is portraying the underlying distribution with reasonable accuracy. The EKF is over-confident and the RBPF suffers severely from the particle-depletion problem and shows a lack of diversity. In general, in our simulations, we observed that both the EKF and the RBPF suggest distributions that diverge from that suggested by the MCMC algorithm as the path length of the robot increases and this divergence is usually towards over-confidence. Further insight can be gained by considering the likelihood of the final configurations obtained. Given an adequate burn-in time, the log likelihoods of the final configurations obtained by MCMC approach the same order of magnitude as ground truth, and typically are much higher in likelihood than results obtained from the RBPF, even with a large number of particles; *e.g.* see Figure 5.

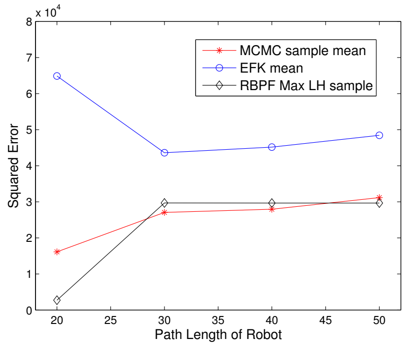
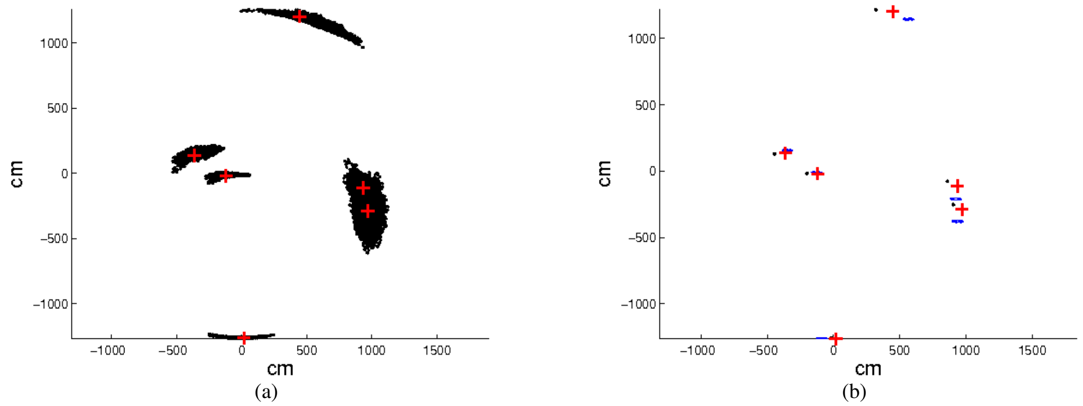


Fig. 6. Squared error of MLE of sensor positions as a function of robot path length through a 6 senor network; (the same simulation presented in Figure 4). The result obtained from the mean of the RBPF samples was similar, but poorer, than the RBPF maximum likelihood sample in this experiment and not presented for improved clarity.

To use the MCMC technique to provide a maximum likelihood estimate (MLE) for the sensor positions, one can consider the sample with maximum likelihood (ML) or the mean of the samples obtained. In our simulations, we observed that the mean of the cloud consistently gave good results, although an estimate obtained from the RBPF on the same problem instance generally had similar accuracy; *e.g.* see Figure 6. The performance of the maximum likelihood MCMC sample had a much higher variance, and while it was occasionally extremely accurate as an estimator, it was overall less consistent.

Figure 7 demonstrates the improved convergence properties of the odometry-based proposal scheme used in conjunction with RB over single-component Metropolis-Hastings. Presumably the application of RB removes some of the correlation between individual components of the state space and allows much larger jumps than would otherwise be



(a) (b)

Fig. 4. Results for data obtained from the simulator for a robot path of 50 steps through a 6 sensor network for the algorithms: a.) MCMC b.) RBPF ($K = 20000$), (black particles) and EKF (blue uncertainty ellipses). The red crosses indicate the actual sensor positions.

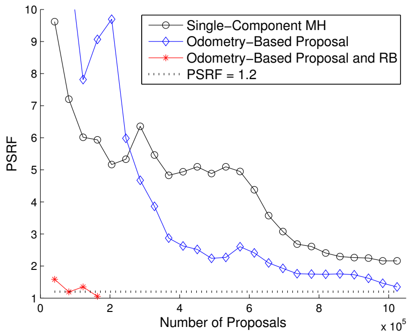


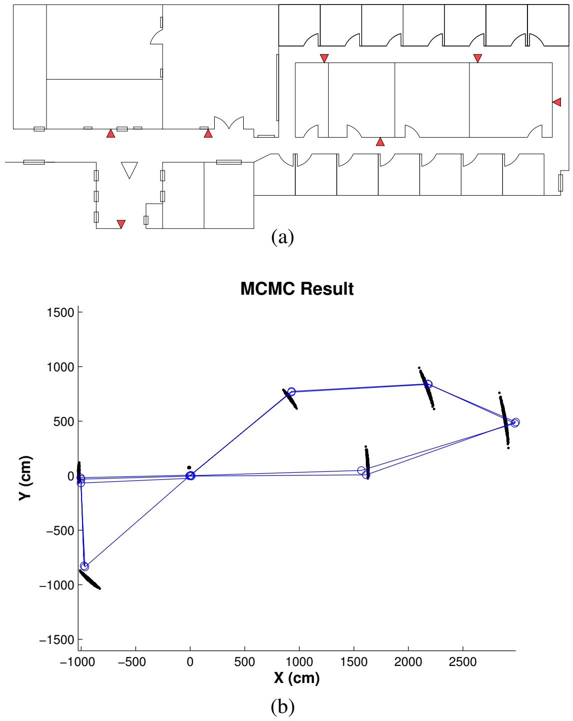
Fig. 7. Example of potential scale reduction factor (PSRF) as a function of computational effort for different variants of the MCMC global inference algorithm. Data are presented based on simulation data gathered from 4 sensor, 12 path length scenario. The PSRF is calculated given 4 restarts of each algorithm.

possible. Supporting this idea is the observation that the automatically tuned sigma values for individual components, (*i.e.* the robot poses s^t), are in general larger when RB is employed than when it is not for the same measurement data.

VI. Experimental Data

We applied our MCMC approach to localization on mapping data gathered from a deployed camera sensor network and a single mobile robot. The target sensor network is located in an office environment, and consists of seven networked cameras. The robot traveled through a pair of loops connected by a long straight hallway with length approximately 50 m as shown in Figure 8(a). A Nomadics Scout robot mounted with a target with six recognizable patterns was used to perform a calibration procedure and obtain position measurements, using the method described in [17].

Due to the size of the environment, and lack of line-of-sight between camera positions, ground truth data could not be collected for this experiment. There are several measures which can be used for qualitative assessment of the estimation accuracy. First, care was taken to return the robot to within a few centimetres of its initial position at the end of the run, which implies the first and last robot positions should agree very closely in any accurate estimate. Also, camera location accuracy can be estimated visually, by comparing to the camera locations recorded on Figure 8(a).



(a)

(b)

Fig. 8. (a) Approximate floor plan showing camera locations during the experiment. (b) The estimated robot path (based on a MLE estimate) and distributions of the camera positions resulting from our approach.

When applied to the data gathered during these experiments our algorithm converged in under 2 hours on a P4, 3.2 GHz machine with 1 GB of RAM. Figure 8 shows the results obtained. This figure includes an approximation to the robot path as a sequence of linear motions based on the ML configuration obtained. Although this MCMC approach is relatively computationally expensive, sensor network calibration can be considered a one-time expense and accurate location and uncertainty results can be utilized for higher level planning and reasoning purposes throughout the lifetime of the system.

The final robot positions can be observed to lie within a meter from the initial position, which is a strong indicator of map accuracy, as the path length is over 200 m in total.

VII. Conclusion

This paper presents an approach to probabilistic sensor network localization that exploits a combination of emplaced sensing nodes and a moving robot. Our approach is capable of providing a representation of the underlying PDF with much greater efficiency and accuracy than other currently available options, and furthermore, provides a principled stopping mechanism for determining when enough computational effort has been expended. This work also demonstrates the limitations of current filtering-based techniques at accurately representing uncertainty in the domain of sensor network localization.

Although aimed at the sensor network domain, this technique can also be employed in most SLAM scenario involving a robot and landmarks with known correspondences. Even for larger scale problems in which the computation required becomes an issue for 'on-platform' implementation, this approach can be run as an off-line batch process. One value might be to provide a measuring-stick for tuning the performance of faster techniques, particularly where their uncertainty measurements are concerned. Additionally, the nature of the MCMC algorithm makes it suitable for extensions to a multi-robot scenario where multiple mobile robots explore the same sensor network collecting information from their interaction with the sensor nodes and with each other.

The work also entails some open problems. One open issue is how the quantity of data collected affects the accuracy with which the PDF can be represented. It appears that one possible optimization step could be to omit some of the constraints in order to quickly arrive at a distribution. Data could then be incrementally included to improve accuracy. This could yield an 'anytime'-style algorithm that could quickly produce usable results which become more refined with additional computation. A related direction would be to alter a standard RBPF such as the one implemented in this work to include a global correction step utilizing our MCMC approach. Incorporating an MCMC step in a RBPF has been considered before, *e.g.* by Doucet *et al.* in [24], and should improve particle diversity and ultimately bring the distribution suggested closer to the target distribution. It would be interesting to see how much computation would be needed to obtain results with reasonable uncertainty estimates.

References

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

[2] A. Gelman and D. B. Rubin, "Inference from iterative simulation using multiple sequences (with discussion)," *Statistical Science*, vol. 7, pp. 457–511, 1992.

[3] M. Bramberger, A. Doblander, A. Maier, B. Rinner, and H. Schwabach, "Distributed embedded smart cameras for surveillance applications," *Computer*, vol. 39, no. 2, pp. 68–75, Feb. 2006.

[4] N. Bulusu, J. Heidemann, and D. Estrin, "Gps-less low cost outdoor localization for very small devices," *IEEE Personal Communications Magazine*, vol. 7, no. 5, pp. 28–34, October 2000.

[5] D. Niculescu and B. Nath, "Ad hoc positioning system (APS) using AoA," in *Proc. of Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 3, San Francisco, CA., 2003, pp. 1734–1743.

[6] D. Moore, J. Leonard, D. Rus, and S. Teller, "Robust distributed network localization with noisy range measurements," in *Proc. of the Second ACM Conference on Embedded Networked Sensor Systems (SenSys '04)*, Baltimore, November 2004.

[7] M. A. Paskin, C. E. Guestrin, and J. McFadden, "A robust architecture for inference in sensor networks," in *In Proc. of the Fourth International Symposium on Information Processing in Sensor Networks 2005 (IPSN-05)*, April 2005, pp. 55–62.

[8] K. Dantu and G. S. Sukhatme, "Rethinking data-fusion based services in tiered sensor networks," in *The Third IEEE Workshop on Embedded Networked Sensors*, 2006.

[9] R. Smith, M. Self, and P. Cheeseman, "Estimating uncertain spatial relationships in robotics," in *Autonomous Robot Vehicles*, I. Cox and G. T. Wilfong, Eds. Springer-Verlag, 1990, pp. 167–193.

[10] M. Montemerlo and S. Thrun, "Simultaneous localization and mapping with unknown data association using fastslam," in *IEEE International Conference on Robotics and Automation*, vol. 2, Taipei, Taiwan, 14-19 Sept. 2003, pp. 1985 – 1991.

[11] F. Dellaert and M. Kaess, "Square Root SAM: Simultaneous location and mapping via square root information smoothing," *International Journal of Robotics Research*, vol. 25, no. 12, pp. 1181–1203, 2006.

[12] M. Kaess, A. Ranganathan, and F. Dellaert, "iSAM: Incremental smoothing and mapping," 2008.

[13] F. Lu and E. Milios, "Optimal global pose estimation for consistent sensor data registration," in *International Conference in Robotics and Automation*, vol. 1. IEEE, 1995, pp. 93–100.

[14] J.-S. Gutmann and K. Konolige, "Incremental mapping of large cyclic environments," in *International Symposium on Computational Intelligence in Robotics and Automation (CIRA'99),*, Monterey, CA, November 1999.

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

[16] D. Marinakis and G. Dudek, "Probabilistic self-localization for sensor networks," in *AAAI National Conference on Artificial Intelligence*, Boston, Massachusetts, July 2006, pp. 976–981.

[17] I. Rekleitis, D. Meger, and G. Dudek, "Simultaneous planning localization, and mapping in a camera sensor network," *Robotics and Autonomous Systems (RAS) Journal, special issue on Planning and Uncertainty in Robotics*, vol. 54, no. 11, pp. 921–932, Nov. 2006.

[18] D. Marinakis, D. Meger, I. Rekleitis, and G. Dudek, "Hybrid inference for sensor network localization using a mobile robot," in *AAAI National Conference on Artificial Intelligence*, Vancouver, Canada, July 2007, pp. 1089–1094.

[19] W. Gilks, S. Richardson, and D. Spiegelhalter, *Markov chain Monte Carlo in practice*. Chapman and Hall, 1996.

[20] W. Hastings, "Monte carlo sampling methods using markov chains and their applications," *Biometrika*, vol. 57, pp. 97–109, 1970.

[21] F. Hamze and N. de Freitas, "From fields to trees: On blocked and collapsed mcmc algorithms for undirected probabilistic graphical models," in *Proc. Uncertainty in Artificial Intelligence*, 2004.

[22] G. Casella and C. P. Robert, "Rao-blackwellisation of sampling schemes," *Biometrika*, vol. 83, no. 1, pp. 81–94, 1996.

[23] S. P. Brooks and A. Gelman, "General methods for monitoring convergence of iterative simulations," *Journal of Computational and Graphical Statistics*, vol. 7, pp. 434–455, 1998.

[24] A. Doucet, N. de Freitas, K. Murphy, and S. Russell, "Rao-blackwellised particle filtering for dynamic bayesian networks," in *In Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*. Stanford, California: Morgan Kaufmann, 2000.

[25] K. Murphy, "Bayesian map learning in dynamic environments," in *In Proceedings of Advances in Neural Information Processing Systems*. Denver, Colorado: MIT Press, 1999, pp. 1015–1021.

[26] P. Maybeck, *Stochastic Models, Estimation and Control*. New York: Academic, 1979, vol. 1.