DISTRIBUTED SELF LOCALISATION OF SENSOR NETWORKS USING PARTICLE METHODS

Nikolas Kantas, Sumeetpal S. Singh

Signal Processing Group,
Department of Engineering
University of Cambridge, UK
{nk234,sss40}@cam.ac.uk

Arnaud Doucet

Department of Computer Science,
Department of Statistics
University of British Columbia, CA
arnaud@cs.ubc.ca

ABSTRACT

We describe how a completely decentralized version of Recursive Maximum Likelihood (RML) can be implemented in dynamic graphical models through the propagation of suitable messages that are exchanged between neighbouring nodes of the graph. The resulting algorithm can be interpreted as a generalization of the celebrated belief propagation algorithm to compute likelihood gradients. This algorithm is applied to solve the sensor localisation problem for distributed trackers forming a sensor networks. An implementation is given for dynamic nonlinear model without loops using Sequential Monte Carlo (SMC) or particle

1. INTRODUCTION

Consider a randomly deployed sensor network that performs target tracking. Recently there has been a motivation towards the adoption distributed and decentralised approaches and the use of ad-hoc sensor networks. In such networks nodes do not communicate with a central processing unit, but instead collaborate by exchanging appropriate messages between neighbouring nodes.

We shall be considering the problem of each node obtaining the coordinates of other nodes in the network relative to some frame of reference without equipping each node with an expensive GPS system. This is known as sensor self-localisation. We approach the localisation problem as each node having to "learn" the position or coordinates of its neighbours relative to itself.

The specific application we consider is the case where each node is a tracker and is the origin of its own coordinate system. In a target tracking scenario it is important for each sensor-tracker node to be able to translate the relative frame of reference of its neighbour to its own in order to utilise the measurements from the rest of the network to enhance tracking. This is essentially equivalent to knowing with precision the coordinates of its neighbours w.r.t. itself.

We consider sensors that only take bearing measurements from a maneuvering target. We shall not be using any nodes of known coordinates or beacons to assist in solving the localisation problem as was done in methods described in [6]. Moreover as in [8] tracking is done concurrently with localisation so that we do not need any training data and can deal with drifts in the estimated localisation parameters.

We propose a general framework for the sensor self localisation problem and solve it in a completely decentralised fashion using dynamic graphical models and message passing. We cast the problem as a static parameter estimation problem for dynamic graphical models and use Recursive Maximum Likelihood (RML). The unknown static parameters are learnt on-line using a stochastic gradient approach. These methods computed distributively are generic, can be applied to any distributed parameter estimation problem without making any linearity or gaussianity assumptions whatsoever. This motivates the use of Sequential Monte Carlo (SMC) techniques, aka particle filters.

As in [4] we use belief propagation to learn the parameters at each node. Our approach differs from [4] in that nodes do not take range or some other measurement on neighbouring nodes, but only have available measurements from the target being tracked by all nodes. Belief propagation ideas have been widely used to perform statistical inference in undirected and directed graphs using message passing [9]. The novelty of our paper relies on a fully decentralized calculation of the log-likelihood gradients for RML on graphs. In this respect, it can be interpreted as a generalization of belief propagation. Received messages at a node shall contain the sufficient statistics sent from the rest of the network, in order for that node to implement a RML algorithm.

The framework and derived methodology can be extended to solve the distributed sensor registration problem as well. This is the problem of each node estimating the biases in the measurements of other nodes [8, 11]. This is more important when in case of sensor failure or addition in the network, each new sensor position and bias is "registered "in the network in a completely decentralised fashion.

2. PROBLEM FORMULATION

We consider a sensor network deployed for the tracking of targets. Let the set of nodes of the network be indexed by the finite set $\mathcal{V}$ while the connectivity of the network is specified by the set of edges $\mathcal{E}$. We will deal with an undirected graphical model $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Thus, two nodes i and j communicate provided the edge $e = (i, j)$ (or (j, i)) belongs to $\mathcal{E}$.

The state of a node v is a random variable X_v that represents the state of the target being tracked. The state of the target X_v would comprise of the target's location and velocities as measured with respect the local coordinate frame of node v. At time n let the state of the target at node v be $X_{v,n}$. The only assumption on the target dynamic model is

$$X_{v,n+1} | \, X_{v,n} = x_v \sim f_v \left(\cdot | \, x_v \right)$$

A local measurement $Y_{v,n}$ of the target's state is generated based on the sensing capabilities of the node. Let the measurement be generated according to the probability density function

$$Y_{v,n} | \, X_{v,n} = x_v \sim g_{\vartheta_v^*} \left(\cdot | \, x_v \right)$$

where ϑ_v^* is the unknown parameter of sensor node v.

Since all nodes are engaged in tracking the same target, it is possible to utilise the measurement of all nodes to enhance the tracking performance. In a *fully coupled* implementation, each node will use the measurements of all other nodes too in its update of the Bayes recursion, i.e. it propagates the filtering distribution

$$\pi_{v,n}(x_v) dx_v = P_{\vartheta^*} (X_{v,n} \in dx_v | \, Y_1, \ldots, Y_n)$$

where Y_n denotes the vector of stacked observations $[Y_{v,n}]_{v\in\mathcal{V}}$ and ϑ^* denotes the vector of stacked parameters $[\vartheta_v^*]_{v\in\mathcal{V}}$. Similarly the prediction distribution will be

$$\pi_{v,n|n-1}(x_v)dx_v = P_{\vartheta^*}(\, X_{v,n} \in dx_v|\, Y_1, \ldots, Y_{n-1})$$

It is clear that a fully coupled implementation is advantageous since nodes whose observations are poor, due to distance or sensing capabilities, can benefit from other sensors with better quality observations.

It is assumed that the true parameter θ^* governing the evolution of the processes $\{X_n\}_{n\geq 0}$ and $\{Y_n\}_{n\geq 1}$ is not known and it to be estimated. For a given sequence of T observations, the batch ML estimate of θ^* is the solution to the maximization of the log likelihood function,

$$L_\theta(Y_{1:T}) = \log P_\theta(Y_1, \ldots, Y_T) = \sum_{n=1}^{T} \log P_\theta(\, Y_n|\, Y_1, \ldots, Y_{n-1})$$

where for $n=1$ the conditional density is understood to be $P_\theta(Y_1)$.

Our aim is to solve θ^* recursively using the sequence of observations $\{Y_n\}_{n\geq 1}$. Recursive Maximum Likelihood (RML) consists of identifying the static parameter $\theta^* = \arg\max L(\theta)$ by using a stochastic gradient algorithm where at time n the parameter estimate θ_n is given by

$$\begin{aligned}\theta_n &= \theta_{n-1} + \gamma_n \nabla \log(P(Y_n|Y_{1:n-1}))\\ &= \theta_{n-1} + \gamma_n \nabla \log(\int g_{\theta_{n-1}}(Y_n|x_n) \times p_{\theta_{1:n-1}}(x_n|Y_{1:n-1})dx_n),\end{aligned} \tag{1}$$

where $\{\gamma_n\}$ is a sequence of step-sizes such that $\sum_n \gamma_n = \infty$ and $\sum_n {\gamma_n}^2 < \infty$.

3. TRANSLATING RELATIVE FRAME OF REFERENCE BETWEEN NODES

For sake of clarity, we make the following assumption which we stress **is not** essential for the framework we propose.

Assumption 1 *All nodes maintain a 2D-cartesian coordinate system and maintain as the state of the target its position and velocity in the relevant directions. Also, each node regards itself as located at the origin of its own coordinate system.*

We refer to a particular node v and would like to use the measurements of the rest of the nodes to compute $\{\pi_{v,n}\}$. Obviously the measurement likelihood depends on the coordinate of each node w.r.t. node v. We denote then $\theta^*_{i\to j}$ the coordinate transformation from node i to j, i.e. $\theta^*_{i\to j}$ is the origin of node i in the coordinate frame of node j. We use then $\theta^* = [\theta^*_{i\to j}]_{(i,j)\in\mathcal{E}}$ as a static registration parameter instead of a measurement bias and we have $\theta^*_{i\to i} = 0$ because of Assumption 1.

We denote the observation likelihood of node v by $g_v(\,\cdot|\,x_v)$. In a Bayes recursion coupled implementation of two nodes, node i incorporates the observation of adjacent node j (connected by an edge),

$$\begin{aligned}\pi_{i,n+1}(x_{i,n+1}) \quad &\propto g_i\left(Y_{i,n+1}|\, x_{i,n+1}\right) g_j(Y_{j,n+1}|\theta^*_{i\to j} + x_{i,n+1})\\ &\times \int f_i(x_{i,n+1}|x_{i,n})\pi_{i,n}(x_{i,n})dx_{i,n}.\end{aligned}$$

where $\theta^*_{i\to v}$ for non adjacent nodes is defined as follows: for any path that connects nodes i and v then

$$\theta^*_{i\to v} = \theta^*_{i\to j_1} + \theta^*_{j_1\to j_2} + \ldots + \theta^*_{j_{n-1}\to j_n} + \theta^*_{j_n\to v}. \tag{2}$$

It is clear that sharing the observations is only possible with the coordinate transformation variable $\theta^*_{i\to j}$. In this paper, we will use the observations from the entire network to update each node's filtering density.

For simplicity **assume all nodes have the same transition density** f. Since we start with the same prior and all nodes have the same local transition densities, it is obvious that the filtering density shared in the network will be the same for all nodes on it. The coordinate transformations are consistent at each filtering step and hence can be used when one node needs to pass a message to another.

4. LEARNING $\theta^* = [\theta^*_{I\to J}]_{I,J}$ BY DISTRIBUTED RML

We propose a distributed implementation of RML for estimating all the coordinate transformations $\theta^*_{i\to j}$. Since there is one parameter per edge, namely $\theta_{i\to j}$ for the edge $(i,j)\in\mathcal{E}$, a particular node, say i, will take ownership of the parameter and recursively update it as observations are received in the network. This node shall be referred as root node for that edge, since it shall be the node to collect messages from the rest of the network in order to iterate $\theta_{i\to j,n}$. The messages received by each node i from its neighbours should be sufficient to iterate the parameter $\theta_{i\to j,n}$ and perform the update step of the Bayesian recursion yielding the filtering density $\pi_{i,n}$. Since the prediction step can be performed locally at each node, we shall be able to estimate $\theta^*_{i\to j}$ while propagating $\pi_{i,n}$.

We now formulate the RML problem for node 1 as the reference node. Note this is an arbitrary choice since any node in the network can become root node. For a fixed parameter $\theta = [\theta_{i\to j}]_{(i,j)\in\mathcal{E}}$, the recursive log likelihood $\log P_\theta(Y_n|Y_{1:n-1})$, where Y_n denotes the vector of stacked observations $[Y_{v,n}]_{v\in\mathcal{V}}$, is

$$J_1(Y_n;\theta) = \log \int \prod_{v\in\mathcal{V}} g_v(\,Y_{v,n}|\, x_1 + \theta_{1\to v})\pi^\theta_{1,n|n-1}(x_1)dx_1$$

where $\theta_{1\to 1} = 0$, while superscript θ on $\pi^\theta_{1,n|n-1}$ emphasizes the dependency on the vector of coordinate transformation. We will now take the gradient of this quantity with respect to $\theta_{1\to 2}$, i.e., we are assuming that node $(1,2)$ is a valid edge and that node 1 has ownership of parameter $\theta_{1\to 2}$, i.e. node 1 is the root node in this case,

$$\begin{aligned}&\nabla_{\theta_{1\to 2}} J_1(Y_n;\theta) =\\ &\nabla_{\theta_{1\to 2}} \log \textstyle\int \displaystyle\prod_{v\in\mathcal{V}} g_v(\,Y_{v,n}|\, x_1 + \theta_{1\to v})\pi^\theta_{1,n|n-1}(x_1)dx_1\\ &= \{\textstyle\int \displaystyle\prod_{v\in\mathcal{V}} g_v(\,Y_{v,n}|\, x_1 + \theta_{1\to v})\pi^\theta_{1,n|n-1}(x_1)dx_1\}^{-1}\\ &\times\{\textstyle\int [\sum_{v\in\mathcal{V}} \frac{\nabla_{\theta_{1\to 2}} g_v(\,Y_{v,n}|x_1+\theta_{1\to v})}{g_v(\,Y_{v,n}|x_1+\theta_{1\to v})}]\\ &\times \prod_{v\in\mathcal{V}} g_v(\,Y_{v,n}|\, x_1 + \theta_{1\to v})\pi^\theta_{1,n|n-1}(x_1)dx_1\\ &+ \textstyle\int \displaystyle\prod_{v\in\mathcal{V}} g_v(\,Y_{v,n}|\, x_1 + \theta_{1\to v})\nabla_{\theta_{1\to 2}}\pi^\theta_{1,n|n-1}(x_1)dx_1\}\end{aligned} \tag{3}$$

In the context of recursive distributed parameter estimation $\pi^\theta_{1,n|n-1}(x_1)$ and its gradient $\nabla_{\theta_{1\to 2}}\pi^\theta_{1,n|n-1}(x_1)$ should be propagated locally at node 1. It also appears necessary to pass $\prod_{v\in\mathcal{V}\setminus\{1\}} g_v(Y_{v,n}|x_1+\theta_{1\to v})$ and $\sum_{v\in\mathcal{V}} \frac{\nabla_{\theta_{1\to 2}} g_v(\,Y_{v,n}|x_1+\theta_{1\to v})}{g_v(\,Y_{v,n}|x_1+\theta_{1\to v})}$ using appropriate messages from the network to node 1 in order to be able to update $\theta_{i\to j,n}$. It can be shown that these messages are sufficient to propagate the filtering and prediction densities as well as their gradients.

4.1. Propagating the filtering and prediction densities derivatives

For node 1 we would like to implement a recursion for $\pi^\theta_{1,n}(x_1)$ and $\nabla_{\theta_{1\to 2}}\pi^\theta_{1,n}(x_1)$ given that $\pi^\theta_{1,n-1}(x_1')$ and $\nabla_{\theta_{1\to 2}}\pi^\theta_{1,n-1}(x_1')$

are available locally from previous epoch $n-1$. As node 1 is chosen to be the reference or root node just for convenience and is a completely arbitrary choice, the derivation for any other node is identical.

Consider propagating the prediction and filtering densities and their derivatives. We have:

$$\pi^{\theta}_{1,n|n-1}(x_1) = \int f_1(x_1|x'_1)\pi^{\theta}_{1,n-1}(x'_1)dx'_1, \tag{4}$$

and

$$\pi^{\theta}_{1,n}(x_1) = \frac{\prod\limits_{v\in\mathcal{V}} g_v(Y_{v,n}|\, x_1+\theta_{1\to v})\pi^{\theta}_{1,n|n-1}(x_1)}{\int \prod\limits_{v\in\mathcal{V}} g_v(Y_{v,n}|\, x_1+\theta_{1\to v})\pi^{\theta}_{1,n|n-1}(x_1)dx_1} \tag{5}$$

For the derivatives:

$$\nabla_{\theta_{1\to 2}}\pi^{\theta}_{1,n|n-1}(x_1) = \int f_1(x_1|x'_1)\nabla_{\theta_{1\to 2}}\pi^{\theta}_{1,n-1}(x'_1)dx'_1, \tag{6}$$

and

$$\begin{aligned}
&\nabla_{\theta}\pi^{\theta}_{1,n}(x_1) = \{\textstyle\int \prod\limits_{v\in\mathcal{V}} g_v(Y_{v,n}|\, x_1+\theta_{1\to v})\pi^{\theta}_{1,n|n-1}(x_1)dx_1\}^{-1}\\
&\times\{\prod_{v\in\mathcal{V}} g_v(Y_{v,n}|\, x_1+\theta_{1\to v})\nabla_{\theta_{1\to 2}}\pi^{\theta}_{1,n|n-1}(x_1)\\
&+(\sum_{v\in\mathcal{V}} \tfrac{\nabla_{\theta_{1\to 2}} g_v(Y_{v,n}|x_1+\theta_{1\to v})}{g_v(Y_{v,n}|x_1+\theta_{1\to v})})\\
&\times\prod_{v\in\mathcal{V}} g_v(Y_{v,n}|\, x_1+\theta_{1\to v})\pi^{\theta}_{1,n|n-1}(x_1)\\
&-\pi^{\theta}_{1,n}(x_1)[\textstyle\int \prod\limits_{v\in\mathcal{V}} g_v(Y_{v,n}|\, x_1+\theta_{1\to v})\nabla_{\theta_{1\to 2}}\pi^{\theta}_{1,n|n-1}(x_1)dx_1\\
&+\textstyle\int(\sum\limits_{v\in\mathcal{V}} \tfrac{\nabla_{\theta_{1\to 2}} g_v(Y_{v,n}|x_1+\theta_{1\to v})}{g_v(Y_{v,n}|x_1+\theta_{1\to v})})\\
&\times\prod_{v\in\mathcal{V}} g_v(Y_{v,n}|\, x_1+\theta_{1\to v})\pi^{\theta}_{1,n|n-1}(x_1)dx_1]\}.
\end{aligned} \tag{7}$$

Note that if $\prod\limits_{v\in\mathcal{V}\setminus\{1\}} g_v(Y_{v,n}|\, x_1+\theta_{1\to v})$ and $\sum\limits_{v\in\mathcal{V}} \frac{\nabla_{\theta_{1\to 2}} g_v(Y_{v,n}|x_1+\theta_{1\to v})}{g_v(Y_{v,n}|x_1+\theta_{1\to v})}$ are available at the root node 1, then the filtering and predictions densities together with their derivatives can be computed locally. These quantities should become available to node 1 by messages received from its neighbours, which in turn receive messages from their neighbours. We aim to define an appropriate message passing scheme so that all possible nodes can act as roots and update one or more parameters associated with its adjacent edges.

4.2. Defining Message Passing in the Sensor Network

We have shown that the sufficient statistics that root node 1 needs to compute $\theta_{1\to 2,n}$ as well as propagating the filtering density are $\prod\limits_{v\in\mathcal{V}\setminus\{1\}} g_v(Y_{v,n}|\, x_1+\theta_{1\to v})$ and $\sum\limits_{v\in\mathcal{V}} \frac{\nabla_{\theta_{1\to 2}} g_v(Y_{v,n}|x_1+\theta_{1\to v})}{g_v(Y_{v,n}|x_1+\theta_{1\to v})}$. Therefore we could define messages from each node in a way that it is not necessary to transmit each $\nabla_{\theta_{1\to 2}} g_v(Y_{v,n}|\, x_1+\theta_{1\to v})$ and $g_v(Y_{v,n}|\, x_1+\theta_{1\to v})$ separately in all different possible paths $\overrightarrow{(v,j_l)}$, $\overrightarrow{(j_l,j_{l-1})}$, $\dots$, $\overrightarrow{(j_2,j_1)}$, $\overrightarrow{(j_1,1)}$.

For this reason we inherit a generalized version of belief propagation (BP) approach as described in [9] where messages are defined $m_{1,i\to j}$ and $m_{2,i\to j}$ from node i to j for all $(i,j)\in\mathcal{E}$

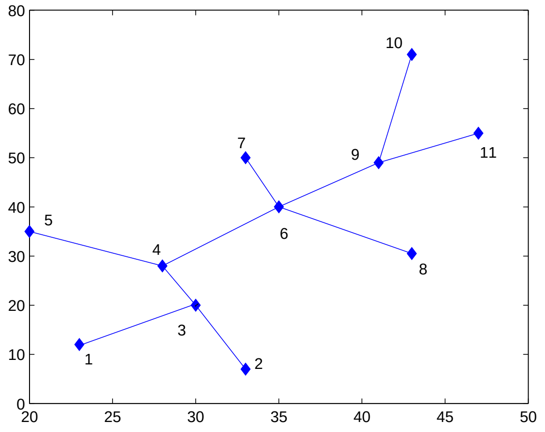

Fig. 1. Sensor Network of the numerical example.

$$m_{1,i\to j}(x_j) = g_i(Y_{i,n}|\, x_j+\theta_{j\to i}) \times \prod_{p\in \mathrm{ne}(i)\setminus\{j\}} m_{1,p\to i}(x_j+\theta_{j\to i}) \tag{8}$$

$$m_{2,i\to j}(x_j) = \frac{\nabla_{\theta_{1\to 2}} g_i(Y_{i,n}|\, x_j+\theta_{j\to i})}{g_i(Y_{i,n}|\, x_j+\theta_{j\to i})} + \sum_{p\in \mathrm{ne}(i)\setminus\{j\}} m_{2,p\to i}(x_j+\theta_{j\to i}) \tag{9}$$

where ne(i) is the set of neighbouring nodes of i in $\mathcal{V}$.

Equations (3)-(7) can be reproduced for any other root node r and parameter $\theta_{r\to q}$, $(q,r)\in\mathcal{E}$, by obvious substitutions to the subscript indices. In addition we can define for each root node r an appropriate message scheduling, where we choose to use only messages $m_{1,i\to j}(x_j)$ and $m_{1,i\to j}(x_j)$ only for (i,j) directed towards that root node, i.e. start from the outer branches of the graphical model and leading each time to node r. So after the root node receives its messages from its neighbours it will have available: $\prod\limits_{p\in\mathrm{ne}(r)} m_{1,p\to r}(x_1+\theta_{r\to p}) = \prod\limits_{v\in\mathcal{V}\setminus\{r\}} g_v(Y_{v,n}|\, x_r+\theta_{r\to v})$ and $\sum\limits_{p\in\mathrm{ne}(r)} m_{2,p\to r}(x_r+\theta_{r\to p}) = \sum\limits_{v\in\mathcal{V}\setminus\{r\}} \frac{\nabla_{\theta_{r\to q}} g_v(Y_{v,n}|x_r+\theta_{r\to v})}{g_v(Y_{v,n}|x_r+\theta_{r\to v})}$, as desired.

As for standard belief propagation [9], the general resulting algorithm will only be exact when the graph/network admits a tree structure. Applying belief propagation algorithms to non-tree topologies is generally referred as Loopy Belief Propagation (LBP) and can in some cases lead to very good approximations.

5. DISTRIBUTED RML IDEAL ALGORITHM

In this section we demonstrate how the general distributed parameter estimation problem for sensor localisation can be solved using RML. Using the results of Section 4 we estimate $\theta_{r\to q}$ for any edge (r,q) using now an arbitrary root node r. At each iteration n all edges should be updated in a cyclic fashion using a valid root node and hence $\theta=\theta_{i\to j_{ij}}$ will be updated for all $(i,j)\in\mathcal{E}$.

Algorithm:
For each edge $(q,r)\in\mathcal{E}$ assign a valid root node, say r, so as to update parameter $\theta_{r\to q}$
At time n,

Prediction Update for all nodes: For all nodes $v\in\mathcal{V}$ derive the prediction densities $\pi^{\theta}_{v,n|n-1}(x_{v,n})$ and their derivatives $\nabla_{\theta_{r\to q}}\pi^{\theta}_{v,n|n-1}(x_{v,n})$

Propagate messages: After $Y_{v,n}$ is received from all nodes $v \in \mathcal{V}$ propagate all possible messages $m_{1,i\to j}$, $m_{2,i\to j}$ given by (8) and (9) for all edges $(i,j) \in \mathcal{E}$ in the network.

Use messages to compute sufficient statistics: At each node v compute $\prod_{p\in \text{ne}(v)} m_{1,p\to v}(x_v + \theta_{v\to p})$ and, for each root node r, $\sum_{p\in \text{ne}(r)} m_{2,p\to r}(x_r + \theta_{r\to p})$.

Update the parameter $\theta_{r\to q}$: At each root node r set

$$\theta_{r\to q,n+1} = \theta_{r\to q,n} + \gamma_n \nabla_{\theta_{r\to q}} J_r(Y_n; \tilde{\theta}_n),$$

where $\tilde{\theta}_n$ is defined so that $\tilde{\theta}_{r\to q} = \theta_{r\to q,n}, \tilde{\theta}_{l\to e} = \theta_{l\to e,n}$, for all $(l,e) \in \mathcal{E}$ and $\nabla_{\theta_{r\to q}} J(Y_n;\theta)$ is given by an expression similar to (3).

Update Filtering density and derivative: Using incoming messages, update at all nodes the current state belief $\pi^{\theta}_{v,n}(x_{v,n})$ and its derivative $\nabla_{\theta_{r\to q}}\pi^{\theta}_{v,n}(x_{v,n})$ as in (7).

Typically, the step-sizes are selected as $\gamma_n = n^{-\gamma}$, where $\gamma > 0.5$, so that $\sum_n \gamma_n = \infty$ and $\sum_n {\gamma_n}^2 < \infty$.

6. NUMERICAL EXAMPLE

The sensor network in Figure 1 has a tree structure and we apply the ditributed algorithm of the previous section to the self localisation problem using bearings only tracking. At each node v, the target being tracked yields observation $Y_{v,n}$ and obeys the following dynamics

$$X_{v,n} = AX_{v,n-1} + BV_{v,n},$$
$$Y_{v,n} = tan^{-1}(X_{v,n}(1)/X_{v,n}(3)) + W_{v,n}$$

with $V_{v,n} \overset{i.i.d.}{\sim} \mathcal{N}(0,Q)$ and $W_{v,n} \overset{i.i.d.}{\sim} \mathcal{N}(0,R)$.

We shall be implementing the presented algorithm using Sequential Monte Carlo (aka particle filters) to solve the problem. We choose nodes $\{3,4,6,9\}$ as root nodes and update at each iteration their adjacent edges. For practical implementation reasons we choose to use a constant step size $\gamma_n = 10^{-4}$. We also initialise $\theta_{i\to j} = 0$ for all $(i,j) \in \mathcal{E}$. In Figure 2 (a), (b) we show $\theta^*_{i\to j} - \theta_{i\to j,n}$, i.e. the error at each iteration between the true parameter and the value of the current iteration of the localisation parameter for all edges $(i,j) \in \mathcal{E}$, when $L = 3000$ number of particles are used.

In Table 1 we present the total mean squared error (MSE) of all the estimated localisation parameters for each of the edges in the graph after convergence is met for different number of particles used.

Number of Particles	Total MSE
1000	0.1228
3000	0.0437
5000	0.0284

Table 1. Mean squared errors after convergence when using different number of particles

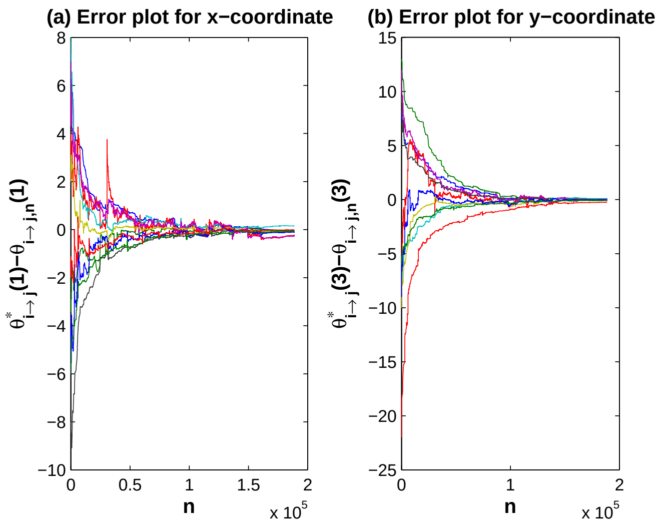

Fig. 2. Error at each iteration between the true parameter and current update of the localisation parameter for all edges of the sensor net

7. REFERENCES

[1] Y. Bar-Shalom, and X. R. Li, Estimation and Tracking: Principles, Techniques, and Software, Artech House Inc.,1993.

[2] T. Brehard, and J.-P. Le Cadre, "Distributed Target Tracking for Nonlinear Systems: Application to Bearings-only Tracking, " In Proc. Fusion (FUSION'05), Vol. 40(2), Philadelphia, USA, 2005.

[3] A. Doucet, J.F.G. de Freitas and N.J. Gordon, Sequential Monte Carlo Methods in Practice. New York: Springer, 2001.

[4] A. T. Ihler and J. W. Fisher and R. L. Moses and A. S. Willsky, "Nonparametric Belief Propagation for Self-Localisation in Sensor Networks, " IPSN'04: Proc. of 3rd IPSN, Berkeley, California, 2004

[5] N. Kantas, S.S. Singh and A. Doucet, "A Distributed Recursive Maximum Likelihood Implementation for Sensor Registration", Proc. International Conference on Information Fusion, Florence 2006

[6] K. Langendoen and N. Reijers "Distributed localization in wireless sensor networks: a quantitative comparison, "Computer Networks, Vol. 43, November 2003

[7] F. LeGland, and L. Mevel, "Recursive Identification in Hidden Markov Models" Proc. of 36th IEEE CDC, pp:3468-3473, 1997.

[8] N. N. Okello, and B. Ristic, "Maximum Likelihood Registration for Dissimilar Sensors, " IEEE Transactions on Aerospace and Electronic Systems, Vol. 39(3), pp:1074–1083, July 2003.

[9] J. Pearl, Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference, Morgan-Kauffman, 1988.

[10] G. Poyadjis, A. Doucet and S.S. Singh, "Maximum Likelihood Parameter Estimation using Particle Methods", Proc. of the American Statistical Association, 2005.

[11] J. Vermaak, S. Maskell, and M. Briers, "Online Sensor Registration, " IEEE Aerospace Conference, Big Sky, MT, 2006 (to appear).