Distributed target tracking under realistic network conditions

Christian Nastasi [1], Andrea Cavallaro [2]

[1] *Scuola Superiore Sant Anna, Piazza Martiri della Libertà, 56127 Pisa (Italy)*
c.nastasi@sssup.it

[2] *Queen Mary University of London, Mile End Road, E1 4NS London (United Kingdom)*
andrea.cavallaro@eecs.qmul.ac.uk

***Abstract*—Distributed target tracking has been a widely studied problem for different applications, more recently considering distributed particle filter techniques for Wireless Sensor Networks (WSNs). However, the adaptation of distributed tracking filters to sensor networks with limited field-of-view and large raw-data management constraints under realistic networking conditions has not yet received enough attention. In this work, we extend the distributed particle filter formulation and enable its operation in realistic scenarios by introducing a next-hop selection mechanism for the aggregation chain and a target hand-over strategy that is capable of handling detection misses and target losses. The filter is tested using a network simulation environment over an area of 6000 m^2 and networks of up to 1000 sensors.**

I. Introduction

Unlike traditional wireless sensor networks (WSNs) that rely on scalar sensor measurements, e.g. temperature or luminosity, Wireless Multimedia Sensor Networks (WMSNs) manipulate more complex information producing vectorial data, e.g. video. The extension of WSN algorithms such as distributed tracking to WMSNs is particularly challenging because of the complexity of raw vectorial-data with respect to raw scalar-data under limited resource conditions.

Multi-sensor tracking aims at estimating the target state, x, given a set of measurements (observations) obtained from N sensor-nodes, each producing a local measurement z_n, with $n = 1...N$. The global measurement vector is $Z = (z_1, z_2, ..., z_N)$. Early approaches for multi-sensor tracking use a *centralized* filter, where all the measurement z_n are sent to a central node that holds the vector Z and executes a classic single-node tracker. This approach is not scalable and it is not feasible in bandwidth-constrained wireless networks, where the physical communication link is potentially shared by all the nodes in the network. A solution to this problem is the use of a decentralized or distributed approach [1]. In *decentralized* tracking, the network is partitioned into clusters according to some sensing relationship between the nodes [2]. For each cluster, a cluster-head is responsible of performing centralized tracking. This solution is effective when nodes observing the same target belong to the same cluster. In *distributed* tracking, the estimation is cooperatively performed through collaboration protocols. The network has a common "distributed" knowledge of the posterior probability function $f(x_k|Z_{1:k-1})$, where k represents the sampling index. This can be achieved through two different information exchange paradigms, namely consensus and token passing. The *consensus-based approach* is equivalent to a data dissemination mechanisms, but is more efficient in terms of communication bandwidth [3]. *Token passing* is a sequential aggregation mechanism: each node receives a partial aggregate, creates a new aggregate with its local information and sends the new aggregate to the next node.

In this work we present a token passing approach for distributed tracking that enables the applicability of a distributed particle filter (DPF) [4], [5] to WMSNs. The extension of a DPF to deal with WMSN in *realistic* deployment scenarios requires several issues to be addressed. In particular, the proposed algorithm deals with the limited field of view (FOV) of the sensors, gaps between sensors (cameras) and introduces active nodes to allow the observation of a target only from a portion of the network. Moreover, we introduce a next-hop selection mechanism for the aggregation chain and a target hand-over strategy that is capable of handling detection misses and target losses. Experimental results using a network simulator for WMSNs [6] on an area of $6000m^2$ and sensor configurations of up to 1000 sensors show the effectiveness of the algorithm in dealing with realistic network conditions.

The remainder of the paper is organized as follows. In Section II we review the state of the art for distributed particle filters, while in Section III we present the proposed algorithm and the extensions that are necessary to deal with WMSN issues. Section IV discusses results obtained by implementing the algorithm on network simulator and, finally, we draw our conclusion in Section V.

II. Related work

Distributed Particle Filters (DPFs) for WSNs have been proposed by Coates in [7]. Each node of the network executes a *local Particle Filter (PF)*, a slightly modified version of the classic PF. In particular, each node has a particle set and exchanges some partial information with other nodes according to a specific protocol with the goal of keeping all the particle sets consistent in all the nodes, i.e. an agreement

The work reported in this paper was carried out while the first author was visiting Queen Mary University of London

on the particle set is reached by the nodes at each time step. Although Coates has the merit of formulating the problem of the DPF for the first time, it is unclear what type of information the nodes should exchange.

Solutions based on the definition of a *common posterior distribution* were proposed in [4], [8], [5]. Each node executes the PF by drawing particles from an importance function chosen as the common posterior distribution at the previous step. The goal is therefore to have the same posterior in all the nodes. However, exchanging the particle set among nodes to approximate the posterior probability is not efficient and does not scale with the size of the target's state and the number of particles.

To reduce the amount of data exchanged, i.e. the particles and the weights, independence from the number of particles is obtained by approximating the posterior with a *Gaussian Mixture Model* (GMM). In [4] the authors proposed two approaches based on a two-level hierarchical architecture in which nodes with correlated measurements are organized in sensor cliques, and then clique-heads are organized in clusters observing the same target. Data within a clique are sent towards the clique-head and the distributed algorithm takes place at the cluster level, i.e. among clique-heads. The first solution in [4] is based on a spatially sequential approximation of the posterior with a GMM. The mechanism is based on a chain similar to that in [7], but, rather than the likelihood, the posterior is approximated and transferred. The second solution in [4] uses a parallel approach and each node derives its own GMM for the posterior. Then, all the GMMs are exchanged among the nodes and the k-means algorithm is used to create the final GMM approximation of the posterior. A similar sequential solution is used in [5]. The two have a slightly different formulation of the weight update function and the GMM parameter generation, although the underlying idea is the same. The authors also propose an alternative "look-ahead" technique to design a more accurate proposal distribution for the importance sampling process, making use of a two-step Gaussian sum filter. Finally, the approach presented in [8] is based on consensus algorithms. The problem is to have the same GMM approximation in all the nodes. The global parameters of the GMM are obtained by averaging local statistics of the nodes using a consensus algorithm.

III. The distributed tracking algorithm

The algorithms discussed in the previous section are defined at a high level of abstraction, without considering issues related to their applicability to realistic networking scenarios. To address issues related to the distributed implementation in the context of a realistic network environment and line-of-sight sensors (cameras), in this section we discuss our extensions of the algorithms presented in [4], [5] for realistic WMSNs.

Let us consider for now that all the nodes, N, in the network observe the target, i.e each node has a measurement and is involved in the DPF iteration. A set of P particles and its relative set of weights is held by each node. The nodes' observations are synchronized, i.e. the k-th measurement is

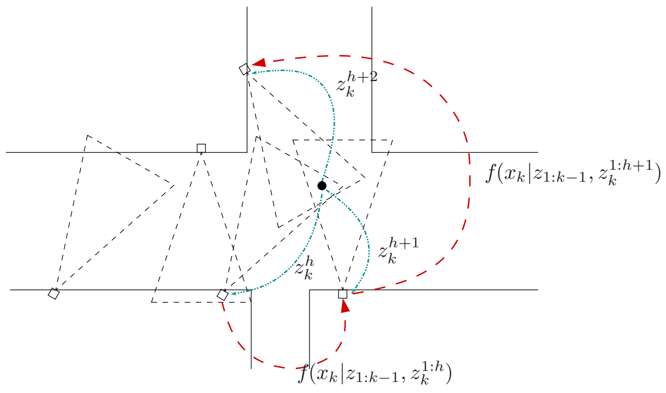

z_k^{h+2}
$f(x_k|z_{1:k-1}, z_k^{1:h+1})$
z_k^{h+1}
z_k^h
$f(x_k|z_{1:k-1}, z_k^{1:h})$

Fig. 1. Example of aggregation sequence using GMM-PP

generated at the same time in each node. The DPF algorithm is based on a sequence of aggregation steps, the *aggregation chain*. The aggregation chain is performed every time a new sample (measurement) is available. Let k indicate a specific instance of the aggregation chain, i.e. a tracking step; whereas h indicates the steps of the aggregation chain, i.e. the *aggregation steps*.

In order to apply DPF to FOV-based sensors (cameras), we have to address the algorithmic constraints introduced by their limited FOV. The "content" of the measurement z_k^n is generated from the detection process performed by node n at time k. If the target is not in the node's FOV, such content cannot be generated. This problem, here referred to as *detection miss*, is not considered in the original algorithm and has an impact in the cooperative protocol. Since the target can exit the FOV of a node, we will introduce a mechanism to manage the *target hand-over*, i.e. when a target that was detected in the previous step is no longer detected in the current one. This mechanism is dependent on the role of the node in the aggregation chain. Note that we are not considering here false positive detections.

Let us refer to the posterior probability obtained at the intermediate steps of the aggregation mechanism as *Partial Posterior* (PP). In case of a GMM approximation, the expression GMM-PP can be used:

$$f_{\mathrm{GMM-PP}}^h \approx f_{\mathrm{PP}}^h = f(x_k|z_{1:k-1}, z_k^{1:h}).$$

Figure 1 shows the token passing mechanism used to exchange the (GMM-) PP.

At *initialization*, P particles $x^{(i)}$ are set equal to the initial target state x_0 and all the weights $w^{(i)}$ are set equal to $1/P$. One of the nodes is selected to be the first node starting the aggregation process.

The mechanism *to select the next node* in the aggregation chain is an important component of the algorithm. The node currently holding the token, n_h, queries the n_{h+1} node that is closer to the (partially) estimated target position $\hat{x}_k^h$. The queried node can refuse if it has already been included in the aggregation process or if it determines that its local measurement is not accurate enough [5]. Unlike the original approach, in our case the nodes are aware of the FOV information of the cameras that could be in the same active set (its logical neighborhood). However, it could be also sufficient to know only which are the cameras of the node's neighborhood. In this case the node shall query all the nodes in the neighborhood without any preference or filter related to the estimated target state. The choice is based on the current node estimation ($\hat{x}_k^h$) and on the knowledge of the FOV of the neighboring nodes. In our implementation, n_{h+1} is selected as the node whose center of the FOV is the closest to the target position.

Another important limitation of the original algorithm is that all the nodes in the network must observe the target at any time, which in case of cameras would mean that the FOVs of all the nodes overlap and that a target only moves inside the overlapping region. To remove this unrealistic assumption we define a subset of *active nodes* that, for a given tracking step, are able to generate a detection of the target.

The **first node**, n_1, starts the iteration for the current tracking step k. The node n_1 has the global posterior probability from time $k-1$ (and does not receive a PP at time k). This could be possible, for instance, if the first node in the current tracking step is also the last one of the previous one. Node n_1 performs the posterior prediction and measurement update steps in order to create the first PP from the previous posterior.

The particle set $\{x_{k-1}^{(i)}, w_{k-1}^{(i)}\}_{i=1}^P$ from the previous tracking step describes the previous posterior probability. This particle set is re-sampled, generating the set

$$\{\bar{x}_{k-1}^{(i)}, \bar{w}_{k-1}^{(i)}\}_{i=1}^P.$$

The prediction step is obtained by drawing a new particle from the state transition function for each re-sampled particle:

$$x_k^{(i)} \sim f(x_k|\bar{x}_{k-1}^{(i)}) \qquad \forall i = 1, \ldots, P\ . \tag{1}$$

The measurement update uses the local measurement z_k^1 to update the weights for the new particle set through the likelihood function,

$$w_k^{(i)} = \frac{f(z_k^1|x_k^{(i)})}{\sum_{j=1}^P f(z_k^1|x_k^{(j)})} \qquad \forall i = 1, \ldots, P\ . \tag{2}$$

The new particle set $\{x_k^{(i)}, w_k^{(i)}\}_{i=1}^P$ represents the PP in the first node. The GMM-PP is then created and sent to the next node. The creation of the GMM-PP is based on unsupervised algorithm that uses expectation maximization and minimum description length order estimation [9].

In case of a detection miss in the first node of the aggregation chain, the algorithm shall not start until a new first node is found. A solution could be to restart the first-node selection mechanism, but this will require several iterations to converge. In the proposed solution the first node with a detection miss selects the next node, according to the standard procedure, to which it forwards the previous GMM-Posterior and a notification of its failure. The receiving node is now in charge of starting the aggregation chain. If also this node fails, the procedure is repeated until a new first node is found. Note that no first nodes are found when there are not any nodes observing the target. We define this condition as *target loss*.

The **intermediate nodes**, n_h with $h = 2, \ldots, N-1$, i.e. all nodes but the first and the last one, receive a PP as input and send a new PP as output. Each n_h receives the (GMM-)PP from the previous aggregation step ($h-1$), updates the PP with its local measurement z_k^{h-1}, creates a GMM approximation of the PP and sends it to the next node n_{h+1}.

The received GMM-PP is used as importance function $g_k(x_k)$ for the importance sampling process. A new particle set is created by drawing P particles from the received GMM-PP:

$$x_k^{(i)} \sim f_{\text{GMM-PP}}(x_k|z_k^{1:h-1}) \qquad \forall i = 1, \ldots, P\ . \tag{3}$$

The relative weights are calculated according to the importance sampling rule

$$\tilde{w}_k^{(i)} = \frac{f(z_k^1|x_k^{(i)}) f_{\text{GMM-PP}}(x_k|z_k^{1:h-1})}{g_k(x_k^{(i)})} \qquad \forall i = 1, \ldots, P$$

$$w_k^{(i)} = \frac{\tilde{w}_k^{(i)}}{\sum_{j=1}^P \tilde{w}_k^{(j)}}\ .$$

However, since the importance function is equal to the incoming GMM-PP, the weight calculation is simplified as in (2):

$$w_k^{(i)} = \frac{f(z_k^h|x_k^{(i)})}{\sum_{j=1}^P f(z_k^h|x_k^{(j)})} \qquad \forall i = 1, \ldots, P\ . \tag{4}$$

The new PP is then approximated with a GMM-PP that is sent to the next node n_{h+1}.

In case of a detection miss in an intermediate nodes, the problem is solved by skipping the local processing (since the detection is not available) and by forwarding the incoming GMM-PP to the next node directly.

The **last node** of the aggregation chain, n_N, reconstructs the global posterior probability and performs the estimation of the target state. The global posterior is reconstructed following the same mechanism described for the intermediate nodes.

In the last node, the PP outcoming from Equations (3) and (4) is also the global posterior,

$$f_{\text{PP}}^N = f(x_k|z_k^{1:N}) = f(x_k|Z_k)\ .$$

The target's state estimation is then possible in n_N as

$$\hat{x}_k = \sum_{i=1}^P w^{(i)} x^{(i)}\ . \tag{5}$$

This last node will act as first node in the next tracking step [5]. Note that, however, the algorithm could be easily modified by selecting a different first node, provided the final global posterior is transferred from $n_{N,k}$ to $n_{1,k+1}$.

A detection miss at the last node does not require particular procedures. The node will produce the estimate considering the incoming GMM-PP as the final GMM-posterior.

If all the nodes appear to have lost the target, a *target loss* condition happens. This condition is due to the target hand-over and takes place when no first node can be selected from the candidate neighborhood. As soon as the target is lost, the node detecting this condition notifies all the nodes in the network with a broadcast message. This message includes the last known posterior. When the target is visible again, the node currently observing it shall start the first node selection process. Note that, owing to imprecisions of the state estimation, it might be possible that the target was visible by other nodes not originally included in the neighborhood. In this case the tracking will immediately restart.

IV. Experiments

In this section, we evaluate the performance of the distributed particle filter we presented in the previous section with (and without) limited resources. *Without resource limitations*, the target state estimation is immediately produced for each new observation by the distributed tracking algorithm. *With resource limitations*, network delays and packet losses may reduce the number of estimations, compared to the total number of available measurements.

Let K_{tr} be the number of estimations and let an estimation be available after a certain delay $d(k)$. The average estimation delay, $\overline{D}$, is defined as

$$\overline{D} = \frac{1}{K_{tr}} \sum_{k=1}^{K_{tr}} d(k) \ , \tag{6}$$

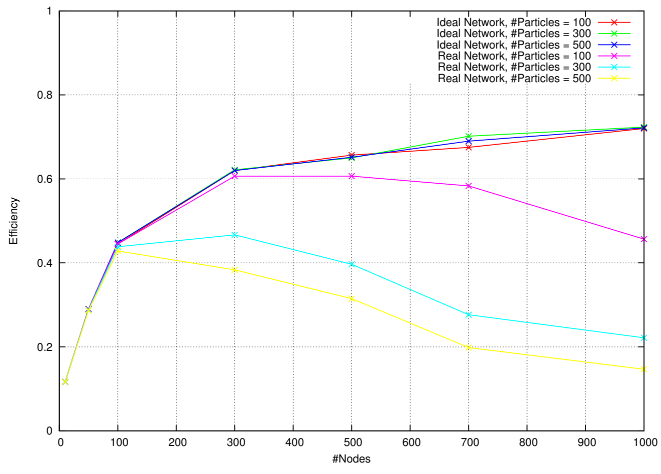
whereas the network estimation efficiency, E, is defined as

$$E = \frac{K_{tr}}{K} \ . \tag{7}$$

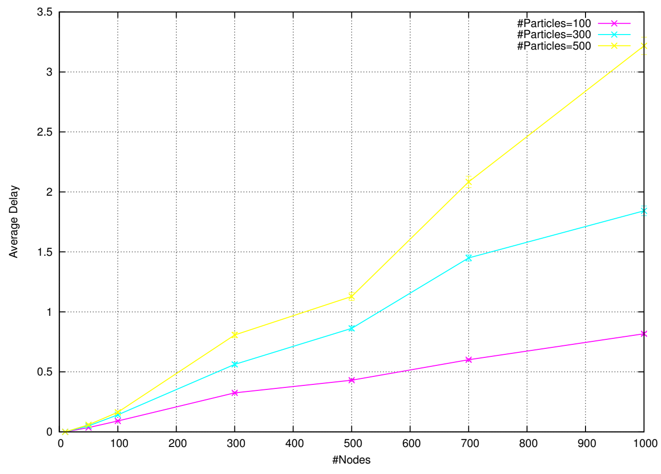
E quantifies the total number of estimations (detected events) over the total number of observations K (all the events).

We define an area of $100m \times 60m$ surveilled by the N cameras having a FOV of $10m \times 6m$ and with a top-down view. The FOV of a camera is calculated considering angles of view of cameras and assuming a distance of $6m$ from the ground plane. The cameras are positioned according to a random uniform distribution. The sampling period is $T_s = 1s$ and there are $K = 600$ observations per run. The network is based on the T-MAC protocol [10], configured with the request-to-send/clear-to-send and acknowledged-transmission mechanisms and with a number of retransmissions set to 10. The bandwidth is $BW = 250$ kbps. The target motion is a linear motion with a random walk that follows a zero-mean normal distribution with a variance of $0.3m$. We reproduce the realistic networking environment using a network simulator engine [?] based on Castalia [11] and Omnet++ [12]. Due to the probabilistic nature of the filter, 100 simulation runs, each of 10 minutes, are generated by varying the number of nodes in the network from $N = 10$ to $N = 1000$. The average values with standard deviation are considered in the analysis.

Figure 2(a) compares under ideal (i.e. no delays) and realistic networking conditions the efficiency of a DPF (0-GMM), when the whole particle set is exchanged across nodes, while varying the number of sensors from 10 to 1000 and the number of particles from 100 to 500. It is possible to notice that in case of ideal networking conditions, the efficiency improves as the number of nodes increases, as deploying more nodes gives a higher degree of coverage of the surveillance



(a)

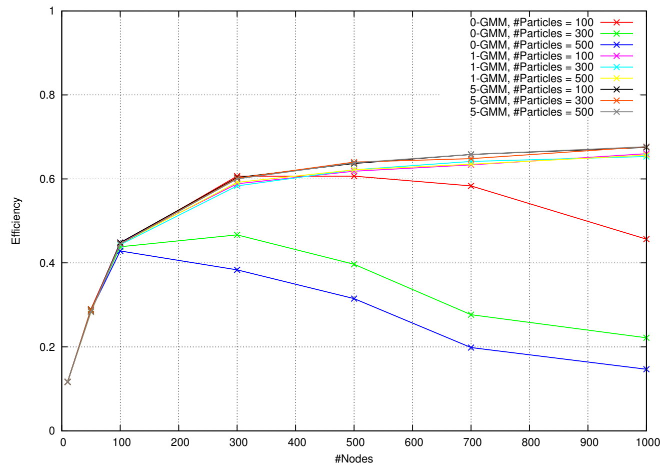


(b)

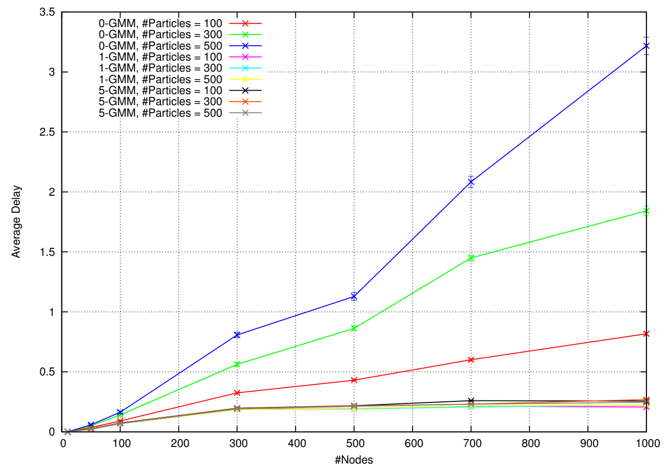
Fig. 2. (a) Efficiency of a DPF (0-GMM) while varying the number of sensors and particles, under ideal and realistic networking conditions. (b) Average delay of a DPF (0-GMM) under realistic networking conditions. Error bars represent standard deviations, whose small values overlap with the X representing the average values

area. However, *when realistic networking conditions are modeled, the performances of the system are sensibly different*. While E is still decreasing in the first part of the curves ($N = 10, 50, 100$), for larger values of N it increases with the number of nodes. This loss of performance is related to network delays, which are clearly visible in Figure 2(b). Because of the limited communication bandwidth, the delay required to completely perform a tracking step increases with N. Indeed, more and more nodes have to be involved in the iteration mechanism of the aggregation process, thus delaying the time when the estimation is ready. The larger the number of particles, the steeper the slope of $\overline{D}$ when increasing the number of sensors. When the delay is larger than the sampling time, some observations are ignored: if a new sample is ready but the network is still completing the aggregation process, that sample is dropped.

Figure 3 shows the impact of using the GMM approximation, as opposed to sending the entire particle set, on



(a)



(b)

Fig. 3. (a) Efficiency and (b) average delay of the methods under analysis under realistic network conditions. Error bars represent standard deviations, whose small values overlap with the X representing the average values

the efficiency E when using the realistic network model. We consider three configurations, namely the DPF with no GMM (0-GMM); the single Gaussian approximation of the DPF (1-GMM); and the approximation with a mixture of five Gaussian components (5-GMM). The GMM approximation considerably reduces the amount of information that the nodes have to exchange. As it can be observed from Figure 3(b), the delay increases much more slowly for 1-GMM and 5-GMM than for 0-GMM. When the GMM approximation are used, the delay remains sufficiently below the sampling time limit ($1s$). It can be also observed that transferring 5 components of the GMM demands clearly more bandwidth (higher delay) than 1 component only. Notice also that the performance of the 1-GMM and 5-GMM are almost comparable to those obtained with 0-GMM under ideal assumptions.

V. Conclusions

We have addressed the problem of distributed target tracking for WMSNs using distributed particle filters and extended the formulation of a sequential algorithm to deal with realistic network scenarios. More specifically, we designed the algorithm to work with sensors with limited field of view, dealing with problems such as detection miss, target handover and target-loss. We demonstrated the proposed algorithm using a network simulator [?]. Simulation results showed the importance of the co-design of distributed tracking algorithms and communication protocols.

References

[1] M. Taj and A. Cavallaro, "Distributed and decentralized multicamera tracking," *Signal Processing Magazine, IEEE*, vol. 28, no. 3, pp. 46 –58, May 2011.

[2] H. Medeiros and J. Park, *Multi-Camera Networks: Concepts and Applications*. Elsevier, 2009, ch. Cluster-Based Object Tracking by Wireless Camera Networks, pp. 539–572.

[3] R. Olfati-saber, J. A. Fax, and R. M. Murray, "Consensus and cooperation in networked multi-agent systems," in *Proceedings of the IEEE*, 2007.

[4] X. Sheng, Y.-H. Hu, and P. Ramanathan, "Distributed particle filter with GMM approximation for multiple targets localization and tracking in wireless sensor network," *IPSN 2005. Fourth International Symposium on Information Processing in Sensor Networks, 2005.*, pp. 181–188, 2005. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1440923

[5] O. Hlinka, P. M. Djuric, and F. Hlawatsch, "Time-space-sequential distributed particle filtering with low-rate communications," in *2009 Conference Record of the Forty-Third Asilomar Conference on Signals, Systems and Computers*. Ieee, Nov. 2009, pp. 196–200. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5470131

[6] C. Nastasi and A. Cavallaro, "Wise-mnet: an experimental environment for wireless multimedia sensor networks," in *Sensor Signal Processing for Defence 2011 (SSPD 2011)*, 2011.

[7] M. Coates, "Distributed particle filters for sensor networks," in *Proceedings of the third international symposium on Information processing in sensor networks - IPSN'04*. New York, USA: ACM Press, 2004. [Online]. Available: http://portal.acm.org/citation.cfm?doid=984622.984637

[8] D. Gu, "Distributed Particle Filter for Target Tracking," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, no. April. Ieee, Apr. 2007, pp. 3856–3861. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4209688

[9] C. A. Bouman, "Cluster: An unsupervised algorithm for modeling Gaussian mixtures," April 1997, available from http://www.ece.purdue.edu/~bouman.

[10] T. van Dam and K. Langendoen, "An adaptive energy-efficient mac protocol for wireless sensor networks," in *Proceedings of the 1st international conference on Embedded networked sensor systems*, ser. SenSys '03. New York, NY, USA: ACM, 2003, pp. 171–180. [Online]. Available: http://doi.acm.org/10.1145/958491.958512

[11] "The Castalia simulation model," http://castalia.npc.nicta.com.au/.

[12] "OMNeT++ Project," http://www.omnetpp.org/, 2010.