# DDSIF: A New Approach for Cooperative Decentralized Tracking

J. Capitán[a], L. Merino[b], F. Caballero[a], A. Ollero[a,c]

[a] *University of Seville, Camino de los Descubrimientos s/n, 41092, Sevilla (Spain)*
[b] *School of Engineering, Pablo de Olavide University, Carretera Utrera km1, 41013, Sevilla (Spain)*
[c] *Center for Advanced Aerospace Technology (CATEC), Seville, Spain*

---

**Abstract**

This paper presents a decentralized data fusion approach to perform cooperative perception with data gathered from heterogeneous sensors, which can be static or carried by robots. Particularly, a Decentralized Delayed-State Information Filter (DDSIF) is described, where full-state trajectories (that is, delayed states) are considered to fuse the information. This approach allows obtaining an estimation equal to that provided by a centralized system and reduces the impact of communications delays and latency into the estimation. The sparseness of the information matrix maintains the communication overhead at a reasonable level. The method is applied to cooperative tracking and some results in disaster management scenarios are shown. In this kind of scenarios the target might move in both open field and indoor areas, so fusion of data provided by heterogeneous sensors is beneficial. The paper also shows experimental results with real data and integrating several sources of information.

*Email addresses:* `jescap@cartuja.us.es` (J. Capitán), `lmercab@upo.es` (L. Merino), `fcaballero@us.es` (F. Caballero), `aollero@cartuja.us.es` (A. Ollero)

*Key words:* cooperative robotics, decentralized data fusion, tracking, sensor networks

## 1. Introduction

Robotic application scenarios have evolved in the last decades from very simple and controlled environments to real-world dynamic applications. In this sense, the cooperation among robots and heterogeneous sensors embedded in the environment for different tasks, like surveillance in urban scenarios [1] or disaster management [2], holds as a very important issue. Real scenarios involve dynamic environments and varying conditions for perception. The robustness and reliability of autonomous perception in these scenarios are critical. In most cases, a single autonomous entity (e.g. a robot or a static surveillance camera) is not able to acquire all the information required for the application because of the characteristic of the particular task or the harmful conditions (e.g. loss of visibility), and thus, the cooperation of several of these entities is relevant.

Therefore, the goal would be to develop a data fusion framework that allows to combine information provided by a wide variety of heterogeneous sensors, as cameras, laser range-finders, and other sensors. A potential solution would be a centralized scheme, in which each sensor just sends all its measurements to a central node where the data fusion is performed. However, this architecture presents some disadvantages that make it unsuitable in real-world applications. These drawbacks include (i) high bandwidth requirements, especially for transmission of high-frequency motion data, (ii) limited range, since each sensor should be within communication range of the central node, and (iii) robustness issues, because a failure in the central node implies that the whole system fails.

The approach should be scalable, robust to communication failures and delays, and work properly under limited bandwidth. Undoubtedly, decentralized approaches can cope with these requirements better than centralized ones [3]. In them, each node of the network employs only information from local sensors and shares its estimations with its neighbors without any knowledge of the full sensor network topology (which will change dynamically if mobile robots are considered) or broadcast facilities. Thus, the need for a central node is eliminated and, as only local communications are performed, scalability is achieved. Moreover, the different agents of the multi-robot platform are allowed to work more independently without the need to keep continuously communication range with a central node.

This paper considers the use of a delayed-state Information Filter (IF) to solve the decentralized cooperative perception problem. As it will be described, the filter considers the trajectory of the state; that is, it maintains information about past states. The main contribution is that using this full-state trajectory (not just the latest state) the nodes can recover the same information than in a centralized version, at the cost of higher message sizes. The paper shows how using a conventional decentralized IF (without past states), the exact solution that would be obtained in a central node fusing all the measurements at their right time steps, cannot be achieved. In the case of dynamic states, some information is proved to be missed when the fusion is carried out by using just the latest state.

This previous concept is applied efficiently to the Gaussian case thanks to the IF. The proposed delayed-state IF keeps a constant computational complexity when the trajectory grows. In addition, the sparse structure of the information matrix, whose size grows linearly with the trajectory, is used in order to keep the communication requirements bounded.

Another advantage of this proposal is the possibility to cope with latency in the network, as past information can be fused. Moreover, the approach deals naturally with data that arrive out of order. Also, information about the state trajectory becomes quite important for the multi-target case, in which data association turns out to be a key issue. Since information from the past is maintained, this technique would allow to cope with previous wrong associations. Once a wrong association is detected in a past time step, the trajectory could be recalculated forward from then.

The paper is organized as follows; Section II discusses some issues related to decentralized fusion. Section III describes the overall decentralized data fusion framework and details the use of state trajectories in the fusion process. Section IV is devoted to present some experimental results. Finally, Section V gives some conclusions and future work.

### *1.1. Related work*

Fusion of data gathered from a network of heterogeneous sensors is a highly relevant problem in robotics that has been widely addressed in the literature. Most of those works are based on Bayesian approaches, where the sensors are modeled like uncertain sources. There are other possible approaches. For instance, some authors employ Dempster-Shafer theory of evidence [4, 5] for information fusion (see for instance [6], where the authors present a multi-robot map-building approach based on evidential reasoning).

Also, there are approaches based on possibility theory [7], built over the arithmetic of fuzzy sets, as for instance in [8], where the authors employ it for cooperative localization and ball position estimation in Robocup. There are also approaches that work with probabilistic beliefs but employ Consensus Theory to combine them, by using what is called an opinion pool. These approaches were largely ignored within the multi-robot research until recently [9, 10]. This kind of techniques tries to deal also with the issue of disagreement (when two or more robots have inconsistent estimations).

This paper deals with Bayesian information fusion. The simplest way to solve the problem is by fusing all the information from the network in a central entity. In [11], for instance, a centralized Extended Kalman Filter is proposed to perform cooperative tracking. Measurements provided by cameras and a wireless sensor network are sent to a central node where the filter is running.

Nevertheless, in many works such as [3, 12, 13] the advantages of a decentralized scheme are highlighted. These previous works propose decentralized data fusion approaches where active sensor networks share information by means of Bayesian filters. The idea of the Channel Filters in order to fuse the information in a consistent manner is considered in all of them. In [12], the decentralized data fusion algorithm is also used to control a group of robots maximizing locally the expected information. Even though there is no explicit negotiation, the exchange of information among the members may influence others. This concept is introduced as coordinated control.

On the contrary, in [14] the *Covariance Intersection* algorithm is presented. This conservative fusion rule allows to achieve a consistent estimation without the need for Channel Filters when no assumptions can be made about the network topology. Moreover, Uhlmann [15] presents the *Covariance Union* method, which tries to deal with disagreement in a Gaussian decentralized fusion setup.

The main issues and problems with decentralized information fusion can also be traced back to the work [16], where the Information Filter (IF, dual of the Kalman Filter) is used as the main tool for data fusion for process plant monitoring. The IF has very nice characteristics for decentralization, and for instance it has been used for decentralized mapping with aerial vehicles in [17, 18]. These works demonstrate that, for the case of static states (for instance, in mapping applications), the decentralized implementation of the IF allows to obtain locally a final estimation that is the same as that obtained by a centralized node with access to all the information. In particular, this is applied in [17] by means of the project ANSER, where a team of UAVs was developed in order to perform decentralized tracking and SLAM.

In the case of dynamic states, for instance in tracking applications, it was noticed in [19, 20] that if only information about the current estimation is exchanged, information will be missed with respect to a centralized estimation. The problem is due to the fact that there is some information not taken into account when performing the prediction steps in each fusing node. In both approaches, delayed-state information is considered to tackle this problem. To the best of the author's knowledge, [20] is the closest work to the approach presented in this paper. It also shows how using delayed information can be even used to overcome the problems of rumor propagation in decentralized systems. However, only results in simulation are shown. [19] adds a protocol that enables to selectively communicate maximally informative measurements. Hence, there is no need to send all the delayed information every time.

Furthermore, in [21] is shown how the exact centralized solution can be obtained with a single IF just when the measurements arrive in order. Thus, when a measurement arrives, some predictions are made backwards to calculate the previous state and add the update information at its right time. Once this information is incorporated, the state can be predicted forward again. However, when the information can arrive out of order, they propose to keep a history of the previous states to recover the centralized solution.

Delayed-state filters have been increasingly used by the SLAM community, as in [22, 23], but mainly due to the sparseness characteristics of the IF for Markov processes with high dimensional states. In [22], the Sparse Extended Information Filter is introduced. When the links between the robot and the features are bounded, a sparse information matrix can be maintained by this filter. Moreover, in [23] it is demonstrated that the information matrix (for the SLAM problem) is exactly sparse in the delayed-state framework. In [24], the authors take into account both advantages, easy decentralization and sparseness, at the same time for decentralized mapping of sensor nodes based on signal strength.

In particular, this paper is an extension of the previous work [25]. In this paper, all the details for the algorithms in [25] are addressed and a wider range of experimental results are shown too.

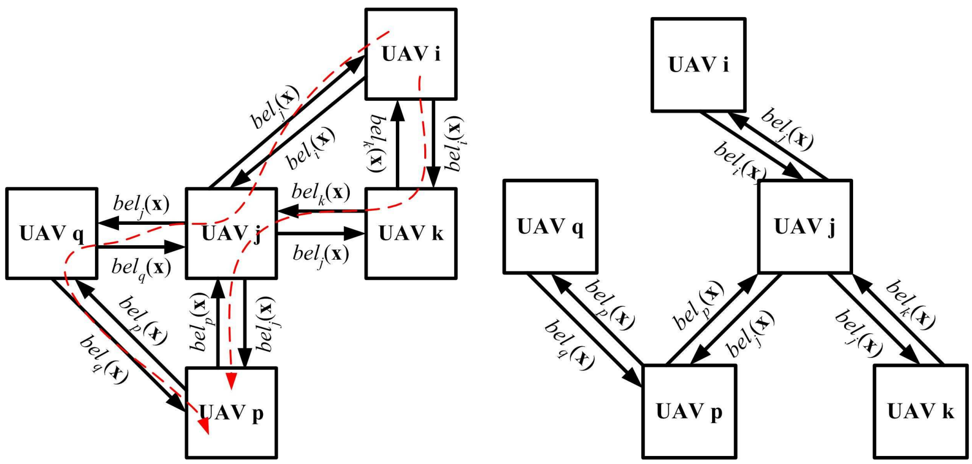
## 2. Decentralized Data Fusion

A decentralized data fusion approach is characterized by the following [18]:

- Each node only accesses measurements from its local sensors and obtain a local estimate.
- Each node communicates with its neighbors its local estimate, and receives estimates from its neighbors.
- There is no broadcast facility, in the sense that cannot be ensured that the information sent will reach all the network nodes.

When a decentralized data fusion approach is considered, some relevant issues must be taken into account carefully. These issues are highly important in the sense that they could lead to inconsistent estimations. If the estimated covariances are higher than the actual ones, the estimation is considered to be consistent. In the case of inconsistent estimations, the filter may end up diverging.

Firstly, decentralized information fusion raises the problem of rumor propagation (or double counting of information). This problem consists of incorporating locally the same received information more than once. Actually, this would reduce the covariance of the estimation artificially, what could lead to inconsistent estimations. Therefore, when non-independent sources of information are fused, their correlation (common information) must be removed in order to assure consistent outcomes [14].




(a) (b)

Figure 1: (a) Rumor propagation example due to multi-path communication. (b) Tree-like network solution.

If a pair of nodes maintains a communication link, they both will fuse all the information received from each other. However, the information previously incorporated from the other node should be discarded. For instance, a multi-path network could lead to a situation where a node receives a message which has been previously received by an alternative path (see Fig. 1a).

Thus, the common information between two nodes (information previously shared by them) should be removed before fusing in order to avoid rumor propagation, which can lead to non-consistent estimations (due to the lost of independence in the sources) [16, 17]. In the literature, some typical solutions to cope with this problem can be found. The simplest one is to force a tree topology in the network, as it is depicted in Figure 1. In case of considering fixed network topologies such as tree-like structures, this problem can be overcome by means of channel filters that maintain the common information through a communication channel along time [3, 12]. These fixed topologies can be too rigid for fleets of mobile robots though. Gaussian filters also provide analytical solutions for fusion under unknown common information by using the *covariance intersection* (CI) algorithm [14], which leads to conservative estimations. Furthermore, the use of delayed states allows the filter to avoid common information due to common prediction functions, which is not considered by the canonical IF.

A second important issue is the loss of information when a decentralized approach is used instead of a centralized one. In the ideal case, all the decentralized estimations of the system should converge to the centralized solution, which is considered to be optimal. This can be easily achieved when the states are static. For dynamic states though, further constraints are required. In this case, the marginal belief of the last time step is not enough to recover the centralized solution [19], since some information can be missed during the prediction steps if measurements are not incorporated at its right moment and order. As it will be shown in detail in following sections, a filter which considers delayed states can be used to cope with this problem. Thus, past information which is received later due to communication delays could be added correctly into the filter.

### *2.1. Bayesian Decentralized Data Fusion*

In a Bayesian setup, the objective is to estimate a degree of belief $bel(\mathbf{X})$ of the state $\mathbf{X}$ of the environment by using all the measurements gathered by the sensors on the $M$ robots of a fleet[1], $\mathbf{z}^t = [{}_1\mathbf{z}^{tT}, \ldots, {}_M\mathbf{z}^{tT}]^T$. This belief is the conditional probability distribution of the state given the real data, $p(\mathbf{X}|\mathbf{z}^t)$ . Assuming that the data gathered by the different robots at any time instant $t$ are *conditionally independent* given the state at that instant $\mathbf{X}_t$ (a typical assumption for data fusion that requires that the state carries enough information to model the measurement process; as it will be seen, this assumption is adequate for the experiments shown in this paper), and the usual Markovian assumptions, the Bayes filter to compute the belief state $bel(\mathbf{X}_t)$ is given by:

$$p(\mathbf{X}_t|\mathbf{z}^t) = \underbrace{\eta' \prod_{j=1}^{M(\tau)} p({}_j\mathbf{z}_t|\mathbf{X}_t)}_{update} \underbrace{\int p(\mathbf{X}_t|\mathbf{X}_{t-1}) p(\mathbf{X}_{t-1}|\mathbf{z}^{t-1}) d\mathbf{X}_{t-1}}_{prediction} \tag{1}$$

with $M(\tau)$ the number of observations obtained at time $\tau$, and $\eta'$ a normalization constant.

The belief state $bel(\mathbf{X}^t)$ for the state trajectory (from time 0 up to time $t$) can also be derived:

$$p(\mathbf{X}^t|\mathbf{z}^t) = \eta'' p(\mathbf{X}_0) \prod_{\tau=1}^{\tau=t} \Big[ \prod_{j=1}^{M(\tau)} p({}_j\mathbf{z}_\tau|\mathbf{X}_\tau) \Big] p(\mathbf{X}_\tau|\mathbf{X}_{\tau-1}) \tag{2}$$

where $p(\mathbf{X}_0)$ is the prior. In these centralized filters, accessing to all the information provided by the team at any moment is required. In a decentralized approach, however, each robot employs only its local data ${}_i\mathbf{z}^t$ and then *shares*

[1] Capital letters indicate random quantities, and lower case letters realizations of these quantities. A subindex indicates information at time $t$, while a super index indicate up to time $t$. The prefix refers to the robot's index.

its belief with its neighbors. The received information from other teammates is locally fused in order to improve the local perception of the world. The belief state $bel_i(\mathbf{X}_t)$ for robot $i$ is:

$$bel_i(\mathbf{X}_t) = p(\mathbf{X}_t|_i\mathbf{z}^t) = \eta_i^{'} p(_i\mathbf{z}_t|\mathbf{X}_t) \int p(\mathbf{X}_t|\mathbf{X}_{t-1}) p(\mathbf{X}_{t-1}|_i\mathbf{z}^{t-1}) d\mathbf{X}_{t-1} \quad (3)$$

Considering the full trajectory it results in:

$$bel_i(\mathbf{X}^t) = \eta_i^{''} p(\mathbf{X}_0) \prod_{\tau=1}^{\tau=t} p(_i\mathbf{z}_\tau|\mathbf{X}_\tau) p(\mathbf{X}_\tau|\mathbf{X}_{\tau-1}) \quad (4)$$

Comparing equations (3) and (1), the relation between the complete belief and the local ones is given by:

$$bel(\mathbf{X}_t) = \eta \prod_{i=1}^{M} \frac{bel_i(\mathbf{X}_t)}{\int p(\mathbf{X}_t|\mathbf{X}_{t-1}) bel_i(\mathbf{X}_{t-1}) d\mathbf{X}_{t-1}} \int p(\mathbf{X}_t|\mathbf{X}_{t-1}) bel(\mathbf{X}_{t-1}) d\mathbf{X}_{t-1} \quad (5)$$

If the predicted belief is represented by $\hat{bel}(\mathbf{X}_t) = \int p(\mathbf{X}_t|\mathbf{X}_{t-1}) bel(\mathbf{X}_{t-1}) d\mathbf{X}_{t-1}$, the same equation can be written as:

$$bel(\mathbf{X}_t) = \eta \prod_{i=1}^{M} \frac{bel_i(\mathbf{X}_t)}{\hat{bel}_i(\mathbf{X}_t)} \hat{bel}(\mathbf{X}_t) \quad (6)$$

Figure 2 describes equation (6) in logarithmic form. This equation produces the same output than a centralized version only if each robot sends its belief any time they update it with new data. Otherwise, information will be missed and, clearly, the result will be different than the belief state that would be computed in a centralized system that received all data at any time [19, 26, 27].

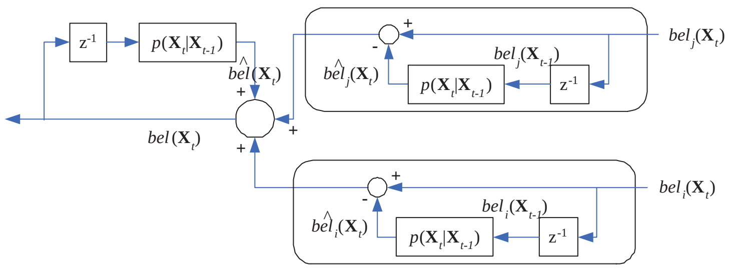



Figure 2: An scheme of the fusion procedure of equation (5) or (6), in logarithmic form. The block $\mathbf{z}^{-1}$ represents a time delay. The predicted belief for each robot is subtracted from the received belief to obtain the likelihood, which is then added to the centralized predicted belief. In the dynamic case, delays in the transmission or missing information will lead to errors with respect to the optimal centralized estimation.

The problem is that, when the state is dynamic, the predicted belief state at any given time depends on all the past observations, so the predicted belief for a node with access to all the information is not the same as the predicted belief for each individual robot. Moreover, the importance of these differences is strongly related to the prediction model and the number of prediction steps carried out in the local nodes between consecutive communications [26].

As noted in [19], in a dynamic state the belief state over the full state trajectory up to time $t$, $bel(\mathbf{X}^t)$, is required to obtain the exact solution. Therefore, comparing (4) and (2), it is possible to obtain the global belief from the local ones:

$$bel(\mathbf{X}^t) = \eta p(\mathbf{X}_0^t) \prod_{i=1}^{M} \frac{bel_i(\mathbf{X}^t)}{p(\mathbf{X}_0^t)} \tag{7}$$

where $p(\mathbf{X}_0^t) = p(\mathbf{X}_0) \prod_{\tau=1}^{\tau=t} p(\mathbf{X}_\tau|\mathbf{X}_{\tau-1})$. Then, if a node of the network receives all the beliefs from the other nodes, the fusion operation consists of combining all the local beliefs after removing the common information they share (the prior over the trajectory $p(\mathbf{X}_0^t)$). Applying this equation, the centralized belief can be exactly recovered.

If (1) is considered, another possibility is to communicate to a central node only the likelihood $p({}_j\mathbf{z}_t|\mathbf{X}_t)$ at a given instant. In this case, the problem is that the transmission of information cannot be delayed (otherwise, information is lost with respect to the fully centralized filter). Besides, with this method the use of a robot as *data mule* is lost: one robot that collects the evidence from a group of local neighbors will communicate it to other robots that could be initially disconnected from the first ones. Moreover, if the connection between two robots is lost, it will lose information that would have been available in future transmissions in the case that the robots had sent their complete beliefs.

Another advantage of using delayed states is that the belief states can be received asynchronously. Each robot can accumulate evidence, and send it whenever it is possible. However, as the state grows over time, the size of the message needed to communicate its belief also does. For the normal operation of the robots, only the state trajectory over a time interval is needed, so these belief trajectories can be bounded. However, the trajectories should be longer than the maximum expected delay in the network in order not to miss any information about past measurements.

In decentralized systems, not only does each robot receives from its neighbors, but also sends information to them. In this case, the fusion equation is slightly different. If robot $i$ received information from $j$, its belief would be updated as it follows:

$$bel_i(\mathbf{X}^t) \leftarrow \eta \frac{bel_i(\mathbf{X}^t) bel_j(\mathbf{X}^t)}{bel_{ij}(\mathbf{X}^t)} \tag{8}$$

where $bel_{ij}(\mathbf{X}^t)$ represents the common information between the robots (i.e., the common prior mentioned above but also information previously exchanged between the robots). This common information can be maintained by a separate filter called channel filter [26], which is basically in charge of predicting the common information up to time $t$. Every time a node $i$ sends or receives information to/from another node $j$, its common information must be updated as follows (assuming beliefs in logarithmic form):

$$bel_{ij}(\mathbf{X}^t) \leftarrow bel_{ij}(\mathbf{X}^t) + \underbrace{bel_j(\mathbf{X}^t) - bel_{ij}(\mathbf{X}^t)}_{j \to i} + \underbrace{bel_i(\mathbf{X}^t) - bel_{ij}(\mathbf{X}^t)}_{i \to j} \tag{9}$$

where the new information received or transmitted is added to the previous common information.

The previous channel filter equations can only be applied if the belief network topology is tree-shaped, that is, if there is a unique path between any pair of providers and receivers [28]. If there are loops in the information channels, each robot cannot determine locally if the received data were previously added. Thus, the same information could be counted twice, what can lead to overconfident estimations.

As a conclusion, all the previous equations have not, in general, an analytic solution. Next section will present how, for Gaussian filters, there is an analytic solution, which, employing delayed states, is able, in theory, to obtain the same results than a centralized node for the case of dynamic states.

## 3. Decentralized Delayed-State Information Filter

### *3.1. Delayed-State Information Filter*

In the particular case of Gaussian distributions, there are analytical solutions to the previous filters, the well-known Kalman Filter (KF). The Information Filter is a more natural approach for decentralized estimation. The IF corresponds to the dual implementation of the KF. The constraints for the application of both filters are the same [29]: Markovian processes, linear prediction and measurement functions, Gaussian noises and initial Gaussian priors. Whereas the KF represents the distribution using its first $\boldsymbol{\mu}$ and second $\boldsymbol{\Sigma}$ order moments, the IF employs the so-called *canonical representation*. The fundamental elements are the *information vector* $\boldsymbol{\xi} = \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}$ and the *information matrix* $\boldsymbol{\Omega} = \boldsymbol{\Sigma}^{-1}$. Prediction and updating equations for the (standard) IF can also be derived from the usual KF. In the case of non-linear prediction or measurement, first order linearization leads to the Extended Information Filter (EIF). For more details, see [27, 29].

The IF presents some advantages and drawbacks when compared to the KF. One of the advantages of the canonical representation is that it can consider complete uncertainty seamlessly in the filter, by setting $\boldsymbol{\Omega}_t = \mathbf{0}$. Furthermore, the prediction and updating steps are dual in the KF and IF, in the sense that the prediction is more complicated in the IF than in the KF, but, on the other hand, the update steps are easier. Moreover, the additive nature of its updating step is what makes the IF interesting for decentralized applications.

The information form also presents some interesting properties when the full-state trajectory $bel(\mathbf{X}^t)$ is considered which allow to run the filter efficiently. If the assumptions for the IF hold, it can be seen that the joint distribution over the full state is also Gaussian. The IF considering delayed states can be derived from the general equation (4)(see [27]). In order to consider a more general case, the EIF equations can be also used to describe the full-state trajectory filter. The following system is considered:

$$\mathbf{X}_t = \mathbf{f}_t(\mathbf{X}_{t-1}) + \boldsymbol{\nu}_t \tag{10}$$

$$\mathbf{Z}_t = \mathbf{g}_t(\mathbf{X}_t) + \boldsymbol{\varepsilon}_t \tag{11}$$

where $\boldsymbol{\nu}_t$ and $\boldsymbol{\varepsilon}_t$ are additive Gaussian noises. In general, $\mathbf{f}_t$ and $\mathbf{g}_t$ could be non-linear functions, so a linearization would be required. Defining the matrices $\mathbf{A}_t$ and $\mathbf{M}_t$ as $\mathbf{A}_t = \nabla\mathbf{f}_t(\boldsymbol{\mu}_{t-1})$ and $\mathbf{M}_t = \nabla\mathbf{g}_t(\bar{\boldsymbol{\mu}}_t)$, and knowing the information matrix and vector up to time $t-1$, $\boldsymbol{\Omega}^{t-1}$ and $\boldsymbol{\xi}^{t-1}$, the prediction steps are:

$$\begin{aligned}\bar{\boldsymbol{\Omega}}^t = &\begin{pmatrix} \mathbf{0} & \mathbf{0}^T & \mathbf{0}^T \\ \mathbf{0} & \boldsymbol{\Omega}_{(t-1)(t-1)} & \cdots \\ \mathbf{0} & \vdots & \ddots \end{pmatrix} + \\ &+ \begin{pmatrix} \mathbf{R}_t^{-1} & -\mathbf{R}_t^{-1}\mathbf{A}_t & \mathbf{0}^T \\ -\mathbf{A}_t^T\mathbf{R}_t^{-1} & \mathbf{A}_t^T\mathbf{R}_t^{-1}\mathbf{A}_t & \mathbf{0}^T \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{pmatrix}\end{aligned} \tag{12}$$

**Algorithm 1** $(\boldsymbol{\xi}^t, \boldsymbol{\Omega}^t) \leftarrow \mathrm{EIF}(\boldsymbol{\xi}^{t-1}, \boldsymbol{\Omega}^{t-1}, \mathbf{z}_t)$

$\bar{\boldsymbol{\Omega}}^t = \textbf{Add\_Block\_Matrix}(\ \boldsymbol{\Omega}^{t-1}) + \begin{pmatrix} \begin{pmatrix} \mathbf{I} \\ -\mathbf{A}_t^T \end{pmatrix} \mathbf{R}_t^{-1} \begin{pmatrix} \mathbf{I} & -\mathbf{A}_t \end{pmatrix} & \mathbf{0}^T \\ \mathbf{0} & \mathbf{0} \end{pmatrix}$

$\bar{\boldsymbol{\xi}}^t = \textbf{Add\_Block\_Vector}(\boldsymbol{\xi}^{t-1}) + \begin{pmatrix} \mathbf{R}_t^{-1}(\mathbf{f}_t(\boldsymbol{\mu}_{t-1}) - \mathbf{A}_t\boldsymbol{\mu}_{t-1}) \\ -\mathbf{A}_t^T\mathbf{R}_t^{-1}(\mathbf{f}_t(\boldsymbol{\mu}_{t-1}) - \mathbf{A}_t\boldsymbol{\mu}_{t-1}) \\ \mathbf{0} \end{pmatrix}$

$\boldsymbol{\Omega}^t = \bar{\boldsymbol{\Omega}}^t + \begin{pmatrix} \mathbf{M}_t^T\mathbf{S}_t^{-1}\mathbf{M}_t & \mathbf{0}^T \\ \mathbf{0} & \mathbf{0} \end{pmatrix}$

$\boldsymbol{\xi}^t = \bar{\boldsymbol{\xi}}_t + \begin{pmatrix} \mathbf{M}_t^T\mathbf{S}_t^{-1}(\mathbf{z}_t - \mathbf{g}_t(\bar{\boldsymbol{\mu}}_t) + \mathbf{M}_t\bar{\boldsymbol{\mu}}_t) \\ \mathbf{0} \end{pmatrix}$

$$\bar{\boldsymbol{\xi}}^t = \begin{pmatrix} \mathbf{0} \\ \boldsymbol{\xi}_{t-1} \\ \boldsymbol{\xi}^{t-2} \end{pmatrix} + \begin{pmatrix} \mathbf{R}_t^{-1}(\mathbf{f}_t(\boldsymbol{\mu}_{t-1}) - \mathbf{A}_t\boldsymbol{\mu}_{t-1}) \\ -\mathbf{A}_t^T\mathbf{R}_t^{-1}(\mathbf{f}_t(\boldsymbol{\mu}_{t-1}) - \mathbf{A}_t\boldsymbol{\mu}_{t-1}) \\ \mathbf{0} \end{pmatrix} \tag{13}$$

And, if one measurement is received, the updating equations are:

$$\boldsymbol{\Omega}^t = \bar{\boldsymbol{\Omega}}^t + \begin{pmatrix} \mathbf{M}_t^T\mathbf{S}_t^{-1}\mathbf{M}_t & \mathbf{0}^T & \mathbf{0}^T \\ \mathbf{0} & \mathbf{0} & \mathbf{0}^T \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{pmatrix} \tag{14}$$

$$\boldsymbol{\xi}^t = \bar{\boldsymbol{\xi}}^t + \begin{pmatrix} \mathbf{M}_t^T\mathbf{S}_t^{-1}(\mathbf{z}_t - \mathbf{g}_t(\bar{\boldsymbol{\mu}}_t) + \mathbf{M}_t\bar{\boldsymbol{\mu}}_t) \\ \mathbf{0} \end{pmatrix} \tag{15}$$

where $\mathbf{R}_t$, $\mathbf{S}_t$ are the corresponding covariances of the additive noises for the prediction and measurement models (10) and (11) respectively. Further

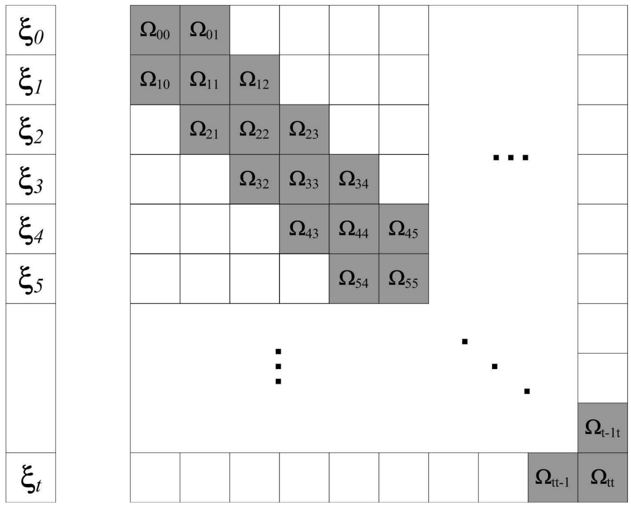



Figure 3: Structure of the information matrix for the full trajectory. The information matrix is a block tridiagonal symmetric matrix, due to the Markov structure of the process.

details can be seen in [27]. Then, the delayed-state EIF is summarized in Algorithm 1, where **Add_Block_Matrix** adds a block row and a block column to the previous information matrix and **Add_Block_Vector** adds a block row to the previous information vector.

Evidently, the state grows along time. In the general case of an information matrix, for a $N$-dimensional state, the storage required is $O(N^2)$. However, in this case, as it can be seen from the prediction and updating equations, the matrix structure is block tridiagonal and symmetric (see Fig. 3) at any time, and thus the storage required is $O(N)$ (where $N$ is the number of time steps). In general, if the measurements only depend on part of the state, the matrix will be sparse. Also, the computational complexity of the algorithm itself is $O(1)$, as the prediction and updating computations at each time instant only involve the previous block. These considerations allow the proposed approach to cope with the delayed states more efficiently than the classical KF does.

### *3.1.1. State Reduction*

In certain situations, the length of the estimated trajectory should be limited, for instance due to storage or bandwidth restrictions. Therefore, a method for reducing the state whenever the size of the trajectory grows over a given threshold is required.

In order to do this, the removed part of the trajectory should be marginalized out. The marginal of a multivariate Gaussian in the canonical form can be computed in closed form [23]. Moreover, due to the structure of the information matrix for this case, the computations required only involve local block matrix operations (see Fig. 4). In addition, this marginalization operation maintains the block tridiagonal structure of the matrix. In general, if the information at time $t$ is eliminated, the only blocks affected are those linked to it (that is, $t-1$ and $t+1$), following:

$$\begin{aligned}
\boldsymbol{\Omega}_{t-1t-1} \leftarrow \boldsymbol{\Omega}_{t-1t-1} - \boldsymbol{\Omega}_{tt-1}^{T}\boldsymbol{\Omega}_{tt}^{-1}\boldsymbol{\Omega}_{tt-1} \\
\boldsymbol{\xi}_{t-1} \leftarrow \boldsymbol{\xi}_{t-1} - \boldsymbol{\Omega}_{tt-1}^{T}\boldsymbol{\Omega}_{tt}^{-1}\boldsymbol{\xi}_{t} \\
\boldsymbol{\Omega}_{t+1t+1} \leftarrow \boldsymbol{\Omega}_{t+1t+1} - \boldsymbol{\Omega}_{t+1t}\boldsymbol{\Omega}_{tt}^{-1}\boldsymbol{\Omega}_{t+1t}^{T} \\
\boldsymbol{\xi}_{t+1} = \boldsymbol{\xi}_{t+1} - \boldsymbol{\Omega}_{t+1t}^{T}\boldsymbol{\Omega}_{tt}^{-1}\boldsymbol{\xi}_{t} \\
\boldsymbol{\Omega}_{t+1t-1} \leftarrow -\boldsymbol{\Omega}_{t+1t}\boldsymbol{\Omega}_{tt}^{-1}\boldsymbol{\Omega}_{tt-1}
\end{aligned} \tag{16}$$

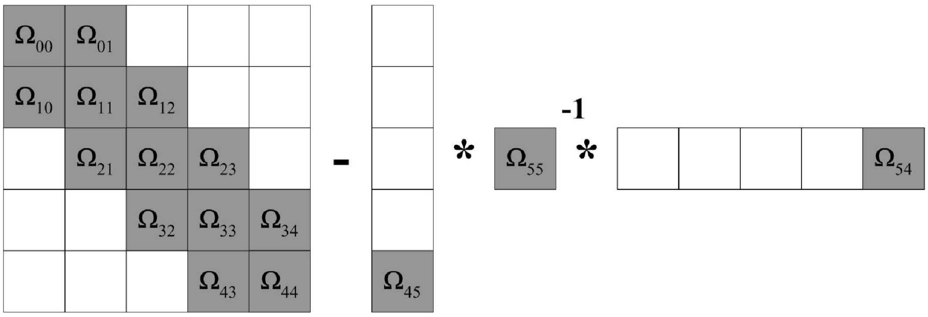



Figure 4: Marginalization of the removed point of the trajectory. Due to the structure of the information matrix, the marginalization only involves local block operations.

### *3.2. Decentralized Information Filter*

The proposed EIF can be easily extended to the multi-robot case, considering a decentralized approach. In this case, each robot will run locally Algorithm 1, updating its full-trajectory state with the information obtained from its sensors. When a robot $i$ is within communication range with other robot $j$, they can share their beliefs, represented by their information vectors ${}_i\boldsymbol{\xi}^t$ and ${}_j\boldsymbol{\xi}^t$, and matrices ${}_i\boldsymbol{\Omega}^t$ and ${}_j\boldsymbol{\Omega}^t$. For Gaussian distributions, equation (8) leads to a quite simple fusion rule:

$$ {}_i\boldsymbol{\Omega}^t \leftarrow {}_i\boldsymbol{\Omega}^t + {}_j\boldsymbol{\Omega}^t - {}_{ij}\boldsymbol{\Omega}^t \tag{17}$$

$$ {}_i\boldsymbol{\xi}^t \leftarrow {}_i\boldsymbol{\xi}^t + {}_j\boldsymbol{\xi}^t - {}_{ij}\boldsymbol{\xi}^t \tag{18}$$

which only requires using a separate EIF to maintain ${}_{ij}\boldsymbol{\Omega}^t$ and ${}_{ij}\boldsymbol{\xi}^t$ (which represent the common information exchanged between $i$ and $j$ in the past). It is important to remark that, if the IF constraints are fulfilled, using this fusion equation and considering delayed states, the local estimator can obtain an estimation that is equal to that obtained by a centralized system. Nevertheless, note that in the case of considering an EIF, local and centralized estimations are no longer exactly the same. This is because the Jacobians calculated for each one could be evaluated for different points ($\boldsymbol{\mu}_t$) at certain time steps.

The common information can be locally estimated assuming a tree-shaped network topology (no cycles or duplicated paths of information). However, this fixed network topology is a constraint too strong on the potential communication links among the (mobile) robots. If there are no assumptions about the network topology, prior to combining the beliefs, unknown common information should be removed. If not, non-consistent estimations could be obtained due to the fact of adding several times the same information. Another option is to employ a conservative fusion rule, which ensures that the system does not become overconfident even in presence of duplicated information. As mentioned previously, for the case of the IF, there is an analytic solution for this, given by the Covariance Intersection algorithm [14]. Therefore, the conservative rule to combine the local belief of a robot $i$ with that received from another robot $j$ is given by:

$$ {}_i\boldsymbol{\Omega}^t \leftarrow \omega({}_i\boldsymbol{\Omega}^t) + (1-\omega)({}_j\boldsymbol{\Omega}^t) \tag{19}$$

$$ {}_i\boldsymbol{\xi}^t \leftarrow \omega({}_i\boldsymbol{\xi}^t) + (1-\omega)({}_j\boldsymbol{\xi}^t) \tag{20}$$

for $\omega \in [0\ 1]$. It can be seen that the estimation is consistent (in the sense that no overconfident estimations are done) for any $\omega$. The value of $\omega$ can be selected following some criteria, such as maximizing the obtained determinant of ${}_i\boldsymbol{\Omega}^t$ (minimizing the entropy of the final distribution). The option chosen by the authors is to use $\omega$ as a fixed weight that setup the system confidence in its own estimation and the neighbor's ones.

Although employing the CI formula avoids the need to maintain an estimation of the common information transmitted to the neighbor systems, as these fusion rules are conservative, some information is lost with respect to the purely centralized case.

Finally, figure 5 depicts the scheme that follows the approach proposed here. The DDSIF for a local agent with its corresponding functional blocks is shown.

### *3.2.1. Synchronization of the trajectories*

Special care has to be taken considering synchronization issues when combining different trajectories. The trajectories are represented at discrete time intervals. The combination formula will work provided that the differences in these intervals are bounded. Therefore, trajectories should be adjusted so that the state space is the same in both cases. Fig. 6 depicts an example of the method.

In this method, first, the newest time steps are predicted by using equations (12) and (13), and the eldest ones are marginalized out (16) until trajectories are adjusted. Thus, in the example, $T'_0$ must be removed and $T'_4$ predicted. Then, each time step is matched with the closest one of the other trajectory. Furthermore, matchings are just allowed if the time difference is lower than a certain threshold. No matched time steps must be also marginalized out before fusing ($T_2$ in the example).

Finally, notice that the algorithm cannot allow crossed matchings such as the one labeled as *WRONG* in Fig. 6 This kind of wrong matchings could

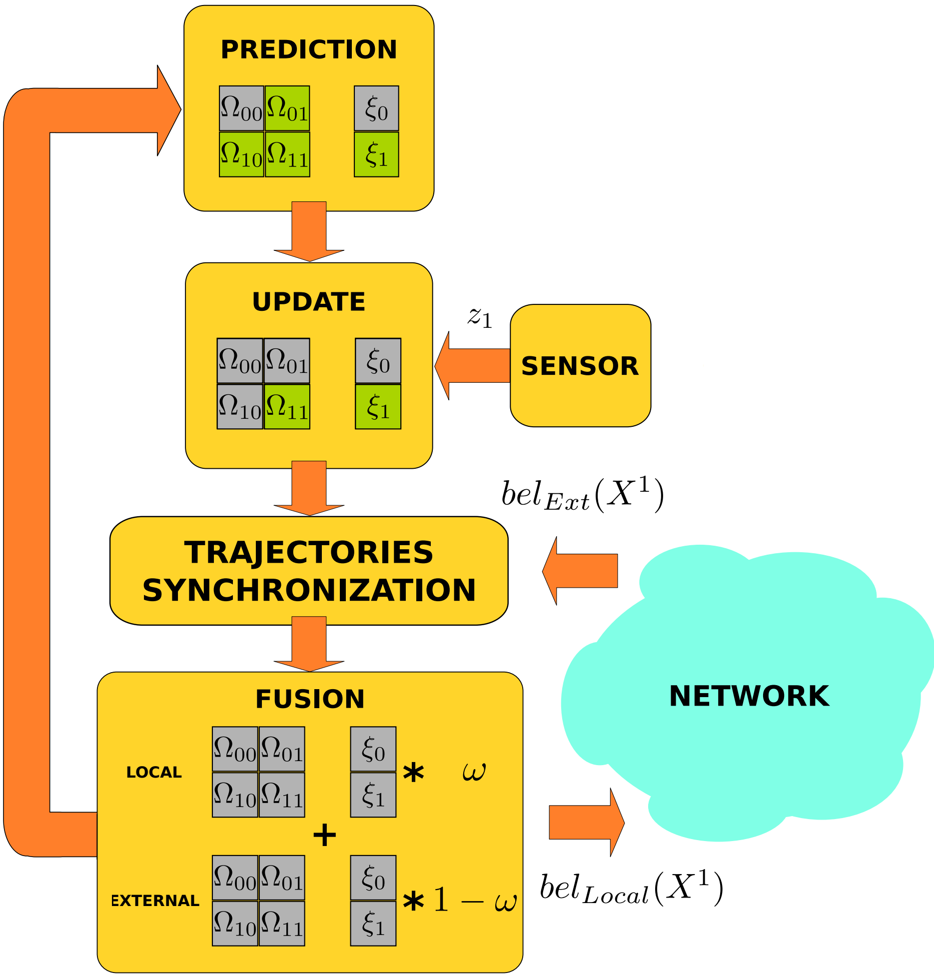



Figure 5: Flow chart of the DDSIF proposed for a local agent. For simplicity, only an example with trajectories of two time instants is shown. In the prediction and update stages, the blocks of the information matrix and vector that are modified are coloured green.

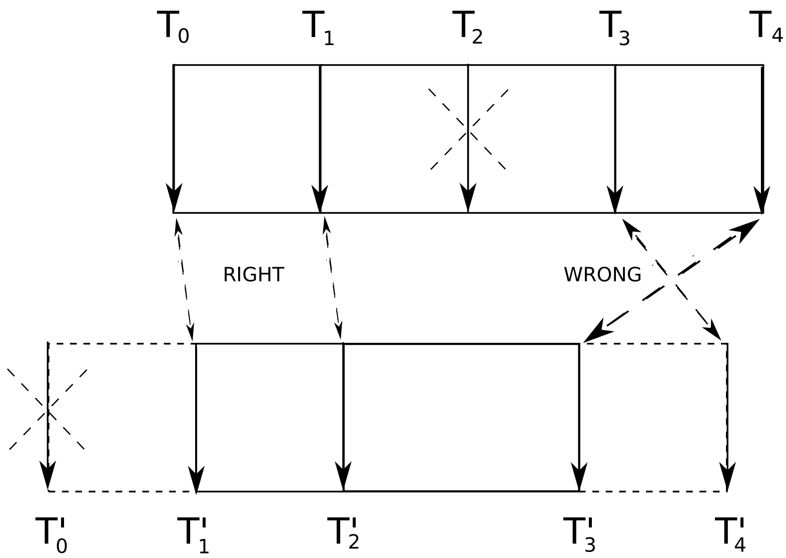



Figure 6: Example of the method to synchronize two trajectories.

result in fatal errors in the estimations.

## 4. Experimental Results

### 4.1. Simulation Example

In this section, some simulations in Matlab are shown. These simple examples were simulated in order to show the concept of missing information when full trajectories are not considered in the estimation of the state. Very similar examples are shown in [21] for the same purposes.

The simulations consist of two agents with sensors tracking a moving vehicle, which is able to move along the $X$ axis (see figure 7). The state to estimate is composed by the position (m) and the velocity (m/s) of the vehicle at every time step:

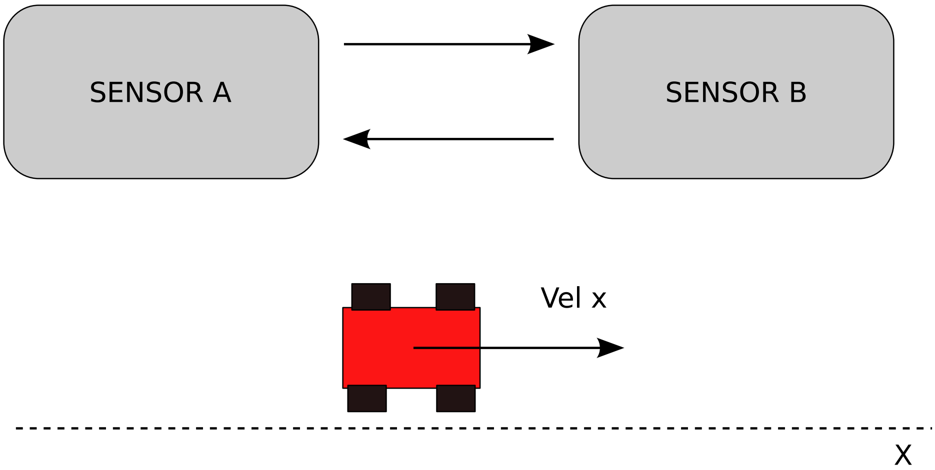



Figure 7: Example simulated in Matlab.

$$\mathbf{X}_t = \begin{pmatrix} X & V_x \end{pmatrix}^T \tag{21}$$

Each agent has a noisy sensor that can measure the vehicle's position directly, so the update model used is linear:

$$\mathbf{z}_t = \begin{pmatrix} 1 & 0 \end{pmatrix} \cdot \mathbf{X}_t + \boldsymbol{\varepsilon}_t \tag{22}$$

with the noise variance $\mathbf{S}_t = 1m^2$. The simulated vehicle always starts at $X = 5m$ and moves with a constant velocity $(8m/s)$. As a generic motion model of the target, a discrete version of the continuous white noise acceleration model or second-order kinematic model is used [30, 31]. In this model, the velocity is assumed to be affected by an acceleration modeled as a white noise of zero mean and with power spectral density $\mathbf{q}$. The discretized version of this linear motion model is characterized by:

| Agent 1 | | Agent 2 | |
|---|---|---|---|
| Measurement | Time Step | Measurement | Time Step |
| 46.18 | 5 | 165.91 | 20 |

Table 1: Simulated measurements for the experiment 1

| Agent 1 | | Agent 2 | |
|---|---|---|---|
| Measurement | Time Step | Measurement | Time Step |
| 46.18 | 5 | 84.41 | 10 |
| 205.63 | 25 | 125.12 | 15 |
| | | 165.91 | 20 |

Table 2: Simulated measurements for the experiment 2

$$\mathbf{A}_t = \begin{pmatrix} 1 & \Delta t \\ 0 & 1 \end{pmatrix} \tag{23}$$

and

$$\mathbf{R}_t = \begin{pmatrix} \frac{1}{3}\Delta t^3 & \frac{1}{2}\Delta t^2 \\ \frac{1}{2}\Delta t^2 & \Delta t \end{pmatrix} \mathbf{q} \tag{24}$$

where $\Delta t = 1s$ and $\mathbf{q} = 0.05m^2/s^3$.

Two similar experiments with the same two agents tracking the vehicle were performed. The measurements gathered locally by each agent's sensors are summarized in the tables 1 and 2.

For every experiment three different approaches were run: (i) A centralized IF, (ii) a decentralized IF without considering delayed states (that is, the proposed algorithm but considering only the last state), and (iii) the proposed delayed-state IF. Moreover, since there were just two agents, a channel filter was able to compute exactly the common information between the two perception entities. Thus, no CI rules were needed during the performance.

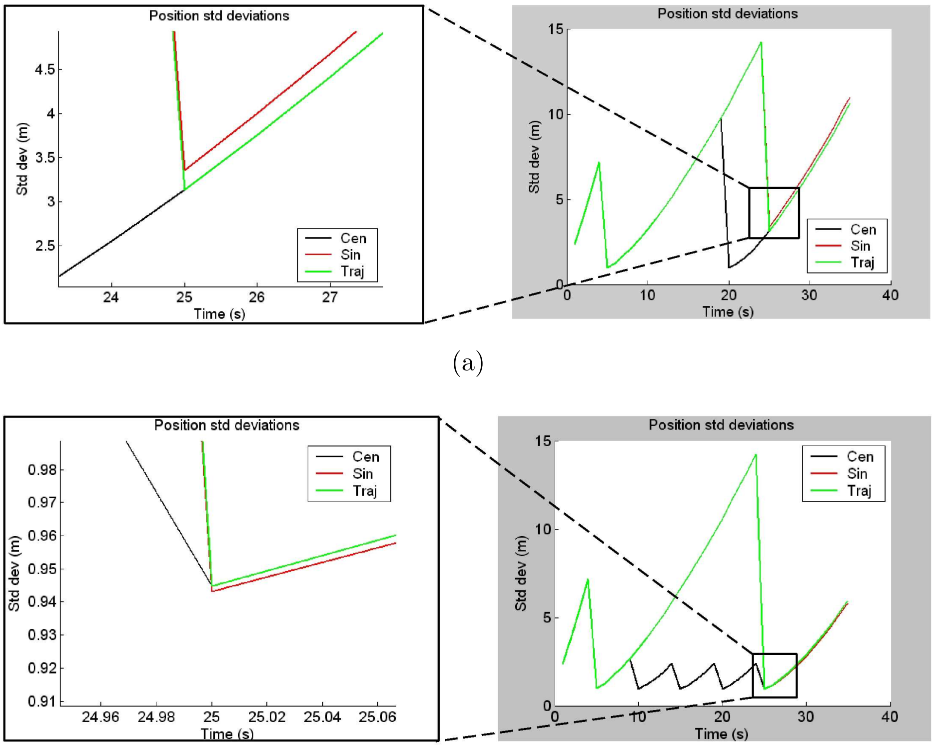
The filter initialization was the same for all the simulations:

$$\begin{aligned} \mu_0 &= \begin{pmatrix} 5 \\ 10 \end{pmatrix} \\ \boldsymbol{\Sigma}_0 &= \begin{pmatrix} 2.5 & 0 \\ 0 & 3 \end{pmatrix} \end{aligned} \tag{25}$$

The values shown in the tables 1 and 2 were obtained simulating the previous models of the vehicle and the sensors for 25 seconds. In both experiments, at the time step 25 the agent 1 fused information received from the other one. The measurements of the sensors were taken at previous time steps in order to show what happens when the fusion is not done every time a local measurement is updated. Thus, between the initialization (which is the same for both agents) and the fusion step, there are several local predictions and updates that are not transmitted until the end of the experiment, what leads to some differences between the tested approaches.

Figure 8 depicts the results of the simulations. Regarding the vehicle's position, standard deviations of the three approaches are compared for the experiment 1 and 2. Just the results for the agent 1 are shown. In both experiments, a zoom has been made at the time step 25 in order to highlight the differences after the fusion.

The specific values of experiment 1 try to show how even in such a simple




(a)

(b)

Figure 8: Standard deviations for the experiment 1 (a) and the experiment 2 (b). Black solid line is the centralized estimation, the red solid line is the decentralized estimation without considering delayed states and the green solid line stands for the proposed delayed-state IF. On the left, both experiments are zoomed at the fusion instant to show that the delayed-state IF is superimposed on the centralized solution.

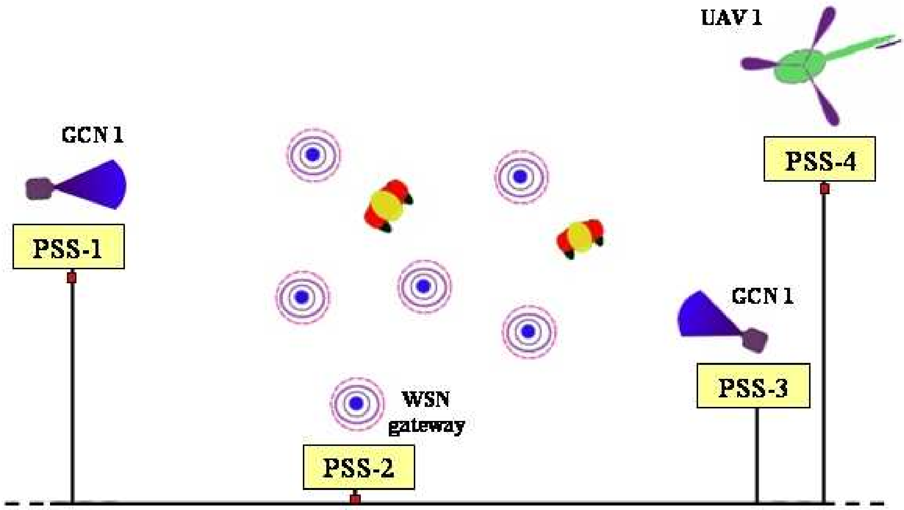



Figure 9: AWARE experiments setup.

example, the centralized solution is not recovered with the decentralized filter which does not consider the full-state trajectories. However, the delayed-state filter is able to recover this centralized estimation. Besides, experiment 2 is included to show another remarkable result: the estimation without delayed states can become overconfident (its variance is smaller than the centralized one after the fusion). This fact indicates that, without considering the trajectories, the estimation could become inconsistent in some cases.

### *4.2. Real Experiments*

In order to test the decentralized perception scheme presented above, a real tracking application is considered in this Section.

The work described in this Section has been developed within the European Commission project AWARE, or Autonomous self-deploying and operation of Wireless sensor-actuator networks cooperating with UAVs [2]. The

(a) (b)

Figure 10: (a) UAVs flying over the experiments area during the AWARE 2009 General Experiments. (b) Ground cameras used to gather visual information during those experiments.

project considers the development of a whole platform considering UAVs, sensors and actuators which can self-deploy. The project deals with issues related to the cooperation among different UAVs and Ground Sensor Networks, such as the systems shown in Fig. 10.

Cooperative tracking is one of the functionalities of the AWARE project, and the presented algorithms were applied within the framework of this project in order to track real fire-fighters in outdoor disaster management scenarios.

The AWARE platform is composed by different Perception Subsystems

(PSS) cooperating among them in order to achieve common objectives (see Fig. 9). Each PSS runs a perception process which is in charge of updating the subsystem status. Thus, this process incorporates the local information obtained by its sensors as well as the information provided by other PSSs (neighbours' beliefs).

Experimental results obtained during real field experiments integrating three sources of information (two cameras and a wireless sensor network) are presented. The information provided by these sensors have been used to track the position of a person moving into the experiments area by means of the decentralized data fusion approach.

The cameras were used to detect the person into the field of view, providing bearing-only information about the position. This measurements can be obtained through a non-linear pin-hole model which will be described in detail. Both cameras were fixed, with known intrinsic and extrinsic calibration parameters. In addition, the person being tracked carried a wireless sensor node that was used by the sensor network to provide positioning information based on the Received Signal Strength Information (RSSI) by means of an approach similar to [24]. Since the provided information consisted of poses referenced to a global frame, it was used to initialize the track.

A C++ implementation of the decentralized data fusion scheme called Perception Subsystem (PSS) has been used to locally process the data gathered by each sensor. These processes incorporate the local information obtained by their sensors as well as the information provided by other PSSs (neighbours' beliefs), cooperating among them in order to achieve common objectives. Then, three PSSs were launched during the experiments: camera

1, camera 2 and wireless sensor network.

The state estimated and shared between PSSs consists of the 3D position and velocity of the person to track, both in the global coordinate system:

$$\mathbf{X}_t = \begin{pmatrix} X & Y & Z & V_x & V_y & V_z \end{pmatrix}^T \tag{26}$$

The prediction model used is the one detailed in the simulation section 4.1, with $\Delta t = 0.2s$ and $\mathbf{q} = 0.05m^2/s^3$.

When dealing with heterogeneous PSSs, different measurement models should be used for each of them. In this case, those models in [11] have been also applied here.

Since the wireless sensor network provides 3D measurements referenced to the global coordinate system, its measurement model is straightforward:

$$\mathbf{z}_t^{wsn} = \begin{pmatrix} X_t & Y_t & Z_t \end{pmatrix}^T + \boldsymbol{\varepsilon}_t^{wsn} \tag{27}$$

On the other hand, each camera provides a measurement composed by the position $(u, v)$ and velocity $(\dot{u}, \dot{v})$ of the target referenced to its image plane, expressed in pixel and pixel/s respectively. These measurements were obtained by means of visual segmentation algorithms that depend on the kind of target.

$$\mathbf{z}_t^{cam} = \begin{pmatrix} u_t^{cam} & v_t^{cam} & \dot{u}_t^{cam} & \dot{v}_t^{cam} \end{pmatrix}^T \tag{28}$$

In order to obtain the measurement model, it is needed to relate objects on the image plane with their positions in the 3D world. Cameras project points in the space into points on the image plane and are usually modeled using the tools of projective geometry [32, 33]. The projection is modeled by

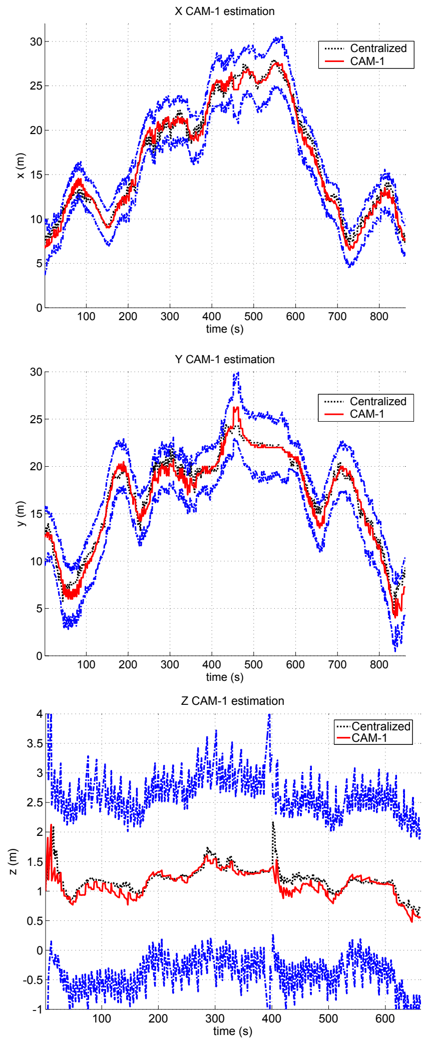



Figure 11: Position estimation of the person using the decentralized approach presented in this paper (red solid line) in camera 1. The estimation provided by a centralized filter is also presented (black dotted line). It can be seen how the estimation is always inside the $3\sigma$ confident interval (blue dashed line)

the *pin-hole* projection model. Following this model, each point in the space, $\mathbf{p} = \begin{pmatrix} X_t & Y_t & Z_t \end{pmatrix}^T$ and its corresponding image pixel $\mathbf{m} = \begin{pmatrix} u_t^{cam} & v_t^{cam} \end{pmatrix}^T$ on the image plane of the camera are related by equation (29), where $\mathbf{p}$ and $\mathbf{m}$ are in homogeneous coordinates:

$$s\mathbf{m}_t = \mathbf{A}_{cal} \begin{pmatrix} \mathbf{Rot}_t & -\mathbf{Rot}_t \mathbf{t}_t \end{pmatrix} \mathbf{p}_t \tag{29}$$

Here, $\mathbf{A}_{cal}$ is the upper triangular intrinsic calibration matrix of the camera. $\mathbf{Rot}_t$ is the rotation matrix from the global reference system to the camera system, and $\mathbf{t}_t$ is the translation camera vector in the global system.

Previous equations imply a non-linear relation between the state and the measurements (due to the homogeneous coordinates). Moreover, if the camera pose is uncertain, it has to be considered when obtaining the corresponding likelihood (and, also in this case, the relation among variables are non-linear). Therefore, if an IF is to be used, a previous linearization is required. In this case, a first order Taylor expansion was used in order to derive an EIF. Although camera poses at each step were assumed to be known, their uncertainties were also considered and propagated through the model Jacobians. In such a way, the noise vector was composed by the additive noises from the measurement itself (they depended on the segmentation algorithm accuracy) and the camera pose uncertainties.

It can be seen that the measurements obtained by the cameras and the wireless sensor networks depend only on the position and velocity of the target (the state), and thus the conditional independence assumption of Section 2.1 is applicable here.

The results of the proposed algorithm are compared with the results obtained by a centralized implementation. In that version, all the measurements were processed off-line by a centralized EIF without considering communication issues or delays. The centralized filter has access to all the information provided by all the sensors instantly, a very important advantage with respect to the decentralized approach.

Thus, Fig. 11 shows the estimated X, Y and Z position of the target provided by the software instance attached to camera 1. It can be seen how the error with respect to the centralized estimation is, in mean, about one meter. In addition, the estimated standard deviation from the filter is coherent with the errors and always inside the $3\sigma$ confident interval.

Note that in this case, thanks to the fusion among the different sources, the bearing only information provided by the cameras can be used to estimate the full target position. This issue would have been hardly addressed with a single camera.

Another important aspect in decentralized approaches is to verify that the estimation carried out by the different software instances converge to a single solution. This is shown in Fig. 12, where the estimated XY trajectory provided by camera 1, camera 2 and wireless sensor network are plot together with the centralized estimation. It can be seen how all estimations converge to the same solution with errors in the order of one meter.

Fig. 13 presents the estimated standard deviation computed by the decentralized approach and the estimated by the centralized filter. The decentralized approach presents more conservative estimations than the centralized filter. The difference between the solutions in this case is explained by different linearization points for the Jacobians in the centralized and decentralized

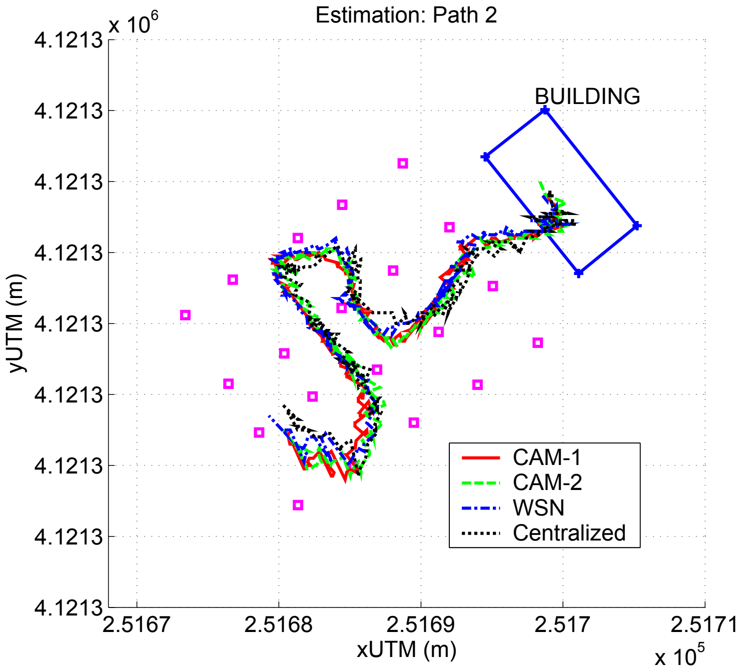



Figure 12: XY estimation provided by the two cameras and the wireless sensor network, and centralized estimation. A sensor network was deployed into the experiments area, pink squares denote the position of each sensor node.

filters; and, mainly, by the use of the *covariance intersection* algorithm. However, it is worth to mention the closeness of both estimations, which differ in no more than half a meter. This fact remarks the consistency and benefits of the proposed approach.

Finally, Fig. 14 shows how the algorithm works when the transmission frequency between the fusion nodes is increased. In these experiments, this transmission period was varied from 1 second to 5 seconds. It can be seen how, after the fusion steps, the estimation is very similar in all cases, due to the use of delayed states. Clearly, between the fusion steps, the differences are higher when the frequency is lower (although after the fusion steps also the past states are recovered, and therefore the same solution as in the central filter is achieved, although with latency). This is an expected result and thus, the transmission frequency is a parameter that should be chosen as a compromise between performance and required bandwidth. Here and in the previous experiments, 5-second trajectories were used so that no information was missed during the performance. Again, some differences can be seen in the cases mainly due to different linearization points in the filters.

To sum up, the experiments showed that the proposed decentralized approach is able to provide estimations with small errors (one meter) with respect to centralized filters and very similar standard deviation estimations (less than half a meter difference), but with the advantage of processing the information in a fully decentralized manner, which basically improves the fault tolerance and scalability of the system.

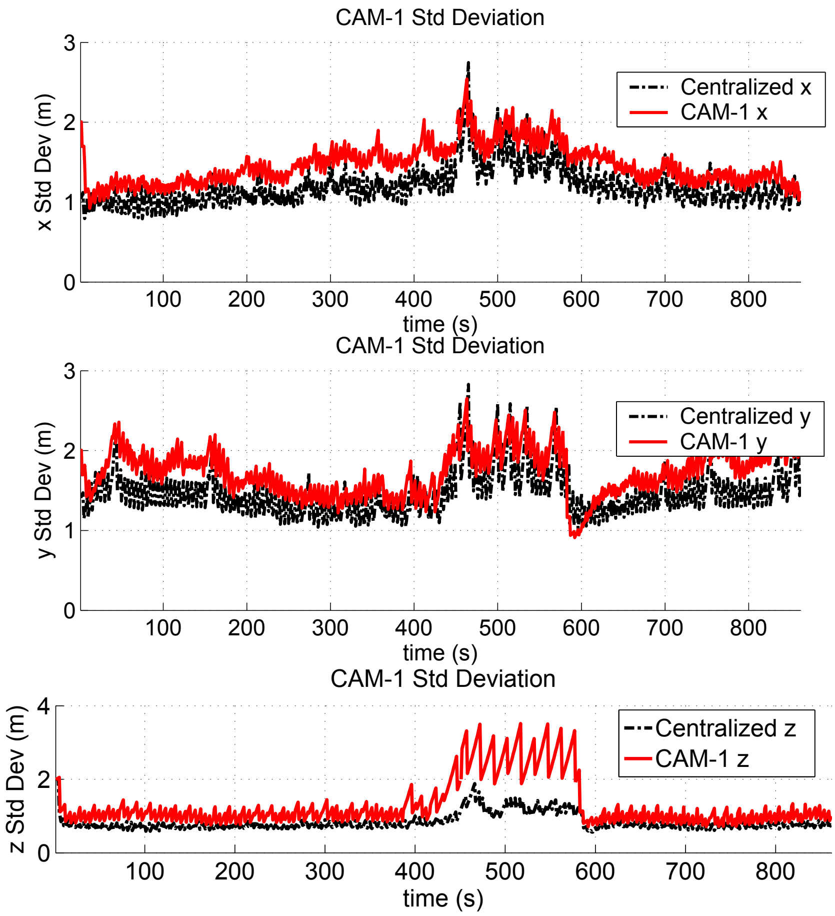



Figure 13: Estimated standard deviation using the decentralized approach (red solid line) versus standard deviation computed by the centralized filter (black dashed line)

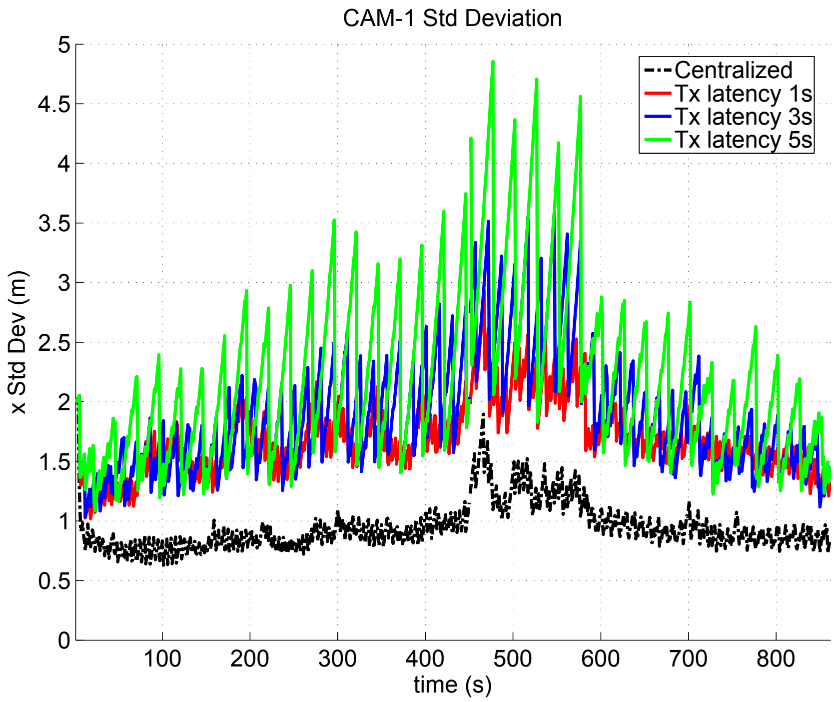



Figure 14: Estimated standard deviation using the decentralized approach varying the transmission frequency between nodes versus standard deviation computed by the centralized filter

## 5. Conclusions and Future Works

The paper presented a decentralized data fusion scheme valid to perform cooperative perception tasks using a set of heterogeneous sensors. An extension of the usual EIF considering delayed states was proposed, which allows to obtain locally the same estimation than a centralized filter, and permits to overcome the usual delays and latency in inter-process communications.

In addition, methods to match trajectories from different agents and to fuse the information in a conservative way were explained. This is particularly important in decentralized architectures in order to face double counting information.

Simple simulations were proposed in order to illustrate how some information can be missed with respect to the centralized case when a standard IF is used in a decentralized manner. The presented approach was proved to be valid to overcome this problem though.

The decentralized data fusion approach has been implemented and tested with real information as well. In particular, three data sources have been integrated in those tests. The experimental results showed that the proposed approach is able to track the position of a moving object in a fully decentralized manner with small errors with respect to a centralized filter, obtaining similar results in mean (about one meter error) and standard deviation (about half a meter difference).

The proposed method is attempted to be a generic approach. Hence, some approximations were described so that non-linear models could be considered and no constraints about the network topology had to be made. It is important to notice that when the information is fused in a conservative manner or linearized models are used, the centralized solution is no longer reached. In those cases (e.g. the open field experiments proposed in this paper), it could be thought that the use of delayed states would not be so beneficial and would just increase the required bandwidth. However, even in those cases, the use of delayed states leads to more robust estimations. Moreover, considering state trajectories can become a powerful tool for other purposes.

For instance, maintaining delayed states, wrong data associations made in the past could be detected later and fixed by recomputing the trajectory up to the current time step. Besides, for tracking applications, keeping a trajectory of the target would provide more useful information when a prediction of its movement has to be made before planning actions.

In addition, the other advantage of the Delayed-State Information Filter is that the communication bandwidth is increased at a fair rate and the trajectories can be bounded in time by means of the presented algorithms. The length of these trajectories has to be chosen so that a compromise between missing information and required bandwidth is reached.

Future works will consider exploiting the information provided by the trajectory. Techniques such as mutual information [34] could be very useful in order to cope with the track-to-track association problem. Moreover, extending that work to the multi-target case, new algorithms could be developed in order to deal with wrong associations made in the past by using the trajectories. Finally, to demonstrate the scalability of the approach, the authors plan to apply it to a bigger network involving several robots, a fixed camera network of around 20 cameras and a Wireless Sensor Network.

## 6. Acknowledgments

J. Capitán, F. Caballero and A. Ollero are partially supported by the ROBAIR Project (DPI2008-03847) funded by the Spanish Research and Development Program, the CONET Network of Excellence (ICT-224053) funded by the European Commission under FP7 and the AWARE project (IST-2006-33579) funded by the European Commission under FP6.

L. Merino is partially supported by the CONET Network of Excellence (ICT-224053) funded by the European Commission under FP7 and the URUS project (IST-2006-045062) funded by the European Commission under FP6.

## References

[1] A. Sanfeliu, J. Andrade-Cetto, Ubiquitous networking robotics in urban settings, in: Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2006, 2006.

[2] Aware project web site, http://www.aware-project.net.

[3] A. Makarenko, A. Brooks, S. Williams, H. Durrant-Whyte, B. Grocholsky, A decentralized architecture for active sensor networks, in: Proceedings IEEE International Conference on Robotics and Automation, ICRA, Vol. 2, 2004, pp. 1097–1102.

[4] A. P. Dempster, A generalization of bayesian inference, Journal of the Royal Statistical Society. Series B (Methodological) 30 (2) (1968) 205–247.

[5] G. Shafer, A mathematical theory of evidence, Princetion University Press, Princetion, NJ, 1976.

[6] Y. Yanli, A. Minai, M. Polycarpou, Evidential map-building approaches for multi-UAV cooperative search, in: Proc. of the American Control Conference, 2005, pp. 116–121.

[7] L. A. Zadeh, Fuzzy sets as a basis for a theory of possibility, Fuzzy Sets and Systems 100 (1999) 9–34.

[8] K. LeBlanc, A. Saffiotti, Multirobot Object Localization: A Fuzzy Fusion Approach, IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics 39 (5) (2009) 1259–1276.

[9] J. Blanco, J. Gonzalez, J. Fernandez-Madrigal, A consensus-based approach for estimating the observation likelihood of accurate range sensors, in: Proceedings of the IEEE International Conference on Robotics and Automation, 2007, pp. 4032–4037.

[10] A. Pahliani, P. Lima, Cooperative opinion pool: A new method for sensor fusion by a robot team, in: Proc. of International Conference on Intelligent Robots and Systems, 2007.

[11] J. Capitan, D. Mantecon, P. Soriano, A. Ollero, Autonomous perception techniques for urban and industrial fire scenarios, in: Proceedings of IEEE International Workshop on Safety, Security, and Rescue Robotics (SSRR), Rome, Italy, 2007, pp. 1–6.

[12] B. Grocholsky, A. Makarenko, T. Kaupp, H. F. Durrant-Whyte, Lecture notes in Computer Science, Vol. 2634, Springer, 2003, Ch. Scalable Control of Decentralised Sensor Platforms.

[13] B. Grocholsky, Information-Theorectic Control of Multiple Sensor Platforms, Ph.D. thesis, University of Sydney (2002).

[14] S. Julier, J. Uhlmann, A non-divergent estimation algorithm in the presence of unknown correlations, in: Proceedings of the American Control Conference, Vol. 4, 1997, pp. 2369–2373.

[15] J. K. Uhlmann, Covariance consistency methods for fault-tolerant distributed data fusion, Information Fusion (4) (2003) 201–215.

[16] S. Grime, H. F. Durrant-Whyte, Data fusion in decentralized sensor networks, Control Engineering Practice 2 (5) (1994) 849–863.

[17] S. Sukkarieh, E. Nettleton, J.-H. Kim, M. Ridley, A. Goktogan, H. Durrant-Whyte, The ANSER Project: Data Fusion Across Multiple Uninhabited Air Vehicles, The International Journal of Robotics Research 22 (7-8) (2003) 505–539. doi:10.1177/02783649030227005.

[18] E. Nettleton, H. Durrant-Whyte, S. Sukkarieh, A robust architecture for decentralised data fusion, in: Proc. of the International Conference on Advanced Robotics (ICAR), 2003.

[19] M. Rosencrantz, G. Gordon, S. Thrun, Decentralized sensor fusion with distributed particle filters, in: Proc. Conf. Uncertainty in Artificial Intelligence, 2003.

[20] T. Bailey, H. Durrant-Whyte, Decentralised data fusion with delayed states for consistent inference in mobile ad hoc networks, Tech. rep., Australian Centre for Field Robotics, University of Sydney (2007).
URL http://www-personal.acfr.usyd.edu.au/tbailey/papers/delayedstateddf.pdf

[21] E. Nettleton, Decentralised Arquitectures for Tracking and Navigation with Multiple Flight Vehicles, Ph.D. thesis, University of Sydney (2003).

[22] S. Thrun, Y. Liu, D. Koller, A. Ng, Z. Ghahramani, H. Durrant-Whyte, Simultaneous localization and mapping with sparse extended information filters, Internation Journal of Robotics Research 23 (7–8) (2004) 693–716.

[23] R. Eustice, H. Singh, J. Leonard, Exactly Sparse Delayed-State Filters for View-Based SLAM, IEEE Transactions on Robotics 22 (6) (2006) 1100–1114.

[24] F. Caballero, L. Merino, P. Gil, I. Maza, A. Ollero, A Probabilistic Framework for Entire WSN Localization Using a Mobile Robot, Journal of Robotics and Autonomous Systems 56 (10) (2008) 798–806. doi:10.1016/j.robot.2008.06.003.

[25] J. Capitan, L. Merino, F. Caballero, A. Ollero, Delayed-State Information Filter for Cooperative Decentralized Tracking, in: Proceedings of the IEEE International Conference on Robotics and Automation, 2009.

[26] F. Bourgault, H. Durrant-Whyte, Communication in general decentralized filters and the coordinated search strategy, in: Proc. of The 7th Int. Conf. on Information Fusion, 2004, pp. 723–730.

[27] L. Merino, A cooperative perception system for multiple unmanned aerial vehicles. Application to the cooperative detection, localization and monitoring of forest fires, Ph.D. thesis, University of Seville (2007). URL http://www.upo.es/isa/lmercab/tesis.pdf

[28] S. Utete, H. Durrant-Whyte, Reliability in decentralised data fusion networks, in: Proc. of the IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems, 1994, pp. 215–221.

[29] S. Thrun, W. Burgard, D. Fox, Probabilistic Robotics, The MIT Press, 2005.

[30] Y. Bar-Shalom, X. Li, T.Kirubarajan, Estimation with Applications to Tracking and Navigation, Wiley Interscience, 2001.

[31] L. D. Stone, T. L. Corwin, C. A. Barlow, Bayesian Multiple Target Tracking, Artech House, Inc., Norwood, MA, USA, 1999.

[32] R. I. Hartley, A. Zisserman, Multiple View Geometry in Computer Vision, 2nd Edition, Cambridge University Press, 2004.

[33] O. Faugeras, Q.-T. Luong, The Geometry of Multiple Images: The laws that govern the formation of multiple images of a scene and some of their applications, The MIT Press, 2001.

[34] A. Dame, E. Marchand, Entropy-Based Visual Servoing, in: Proceedings of the IEEE International Conference on Robotics and Automation, 2009.