Multiple-model Algorithms for Distributed Tracking of a Maneuvering Target

C. Fantacci, G. Battistelli, L. Chisci
DSI, Università di Firenze, Florence, Italy
{claudio.fantacci, giorgio.battistelli, luigi.chisci}@unifi.it

A. Farina, A. Graziano
SELEX Sistemi Integrati, Rome, Italy
{afarina, agraziano}@selex-si.com

Abstract—The paper deals with distributed tracking of a maneuvering target by means of a network of heterogeneous sensors and communication nodes. To effectively cope with target maneuvers, multiple-model filtering is adopted after being extended to a fully distributed processing framework by means of suitable consensus techniques. Novel Distributed first-order Generalized Pseudo Bayesian (DGPB) and Distributed Interacting Multiple Model (DIMM) algorithms are presented. Simulation experiments on critical tracking case studies involving a highly maneuvering target and sensor networks characterized by weak connectivity and target observability properties demonstrate the effectiveness of the proposed distributed multiple-model filters.

Index Terms—Distributed target tracking; sensor networks; nonlinear filtering; multiple-model.

I. INTRODUCTION

The recent advances on Wireless Sensor Network technology make possible to realize distributed surveillance systems by deploying a large number of low-cost wireless position sensor units in the area of interest. This clearly opens up interesting problems on how to efficiently exploit the distributed information spread through the network so that each node can satisfactorily track the target (or targets) of interest even in critical situations of weak network connectivity and/or target observability. Considerable work has been carried out on distributed (parameter or state) estimation following the so called consensus approach [1]–[4]. In particular, effective consensus-based Kalman filters have been devised. It is well known, however, that a single-model Kalman (or nonlinear) filter is ineffective for tracking a strongly maneuvering target and that multiple-model filters [5] are by far preferable for this task. The objective of this paper is, in fact, to develop consensus-based multiple-model filters to be used for tracking highly maneuvering targets with networks consisting of heterogeneous sensors and communication nodes, i.e. nodes that can only process and exchange data but cannot sense data from the environment. A first step in this direction is provided by [6] that proposes a consensus-based distributed multiple-mode Unscented Kalman Filter (UKF) algorithm for jump Markov nonlinear systems. The present paper provides novel contributions with respect to [6] in that a different consensus approach, allowing to get satisfactory performance with a remarkably smaller number of consensus steps, is followed and, further, the presence of communication nodes in the network is also taken into account.

II. PROBLEM FORMULATION

A. Network model

The type of network being considered in this work is schematized in fig. 1. As it can be seen from the figure, the network consists of two types of nodes: communication (COM) nodes have only processing and communication capabilities, i.e. they can process local data as well as exchange data with the neighboring nodes, while sensor (SEN) nodes have also sensing capabilities, i.e. they can sense data from the environment. Notice that, since COM nodes do not provide any additional information, their presence is needed only to improve network connectivity.

Besides this classification between SEN and COM nodes, the network of interest is characterized by the following assumptions: (1) it has no central fusion node; (2) no node is aware of the network topology, i.e. the number of nodes and their connections. From a mathematical point of view, the network can be described in terms of a directed graph \( G = (\mathcal{N}, \mathcal{A}) \) where \( \mathcal{N} \) is the set of nodes and \( \mathcal{A} \subseteq \mathcal{N} \times \mathcal{N} \) the set of arcs, representing links (connections). In particular, \((i, j)\) belongs to \( \mathcal{A} \) if node \( i \) can transmit data to node \( j \). The set of nodes is partitioned into \( \mathcal{N} = \mathcal{S} \cup \mathcal{C} \) where \( \mathcal{S} \) and \( \mathcal{C} \) denote the disjoint subsets of sensor and, respectively, communication nodes. For each node \( i \in \mathcal{N} \), \( \mathcal{N}^i = \{j \in \mathcal{N} : (j, i) \in \mathcal{A}\} \) denotes the set of neighbors of node \( i \). By definition, \((i, i) \in \mathcal{A}\) for any node \( i \in \mathcal{N} \) and, hence, \( i \in \mathcal{N}^i \) for all \( i \). The number of nodes will be denoted as \(|\mathcal{N}|\), the cardinality of \( \mathcal{N} \).

B. Distributed tracking

The attention in this paper will be devoted to the distributed estimation of the state of a dynamical system [7], [8]. This problem, referred to henceforth as Distributed State
Estimation (DSE), has a variety of applications such as for instance distributed tracking, which consists of estimating the kinematic state vector of a mobile target given measurements from multiple non co-located position sensors.

For the subsequent developments, the DSE problem can be formulated as follows. Given the network (graph) \( \mathcal{G} = (\mathcal{N}, \mathcal{A}) \), where \( \mathcal{N} \) is partitioned into sensor nodes \( \mathcal{S} \) and communication nodes \( \mathcal{C} \), the aim is to estimate the state \( x_t \), at time \( t \), of the dynamical system

\[
x_{t+1} = f(x_t) + w_t
\]

given measurements

\[
y^i_t = h^i(x_t) + v^i_t, \quad i \in \mathcal{S}
\]

provided by the SEN nodes. It is assumed that \( w_t \) and \( v^i_t \), for all \( i \in \mathcal{S} \), are mutually uncorrelated white noises with zero mean and covariances \( Q > 0 \) and \( R^i > 0 \), \( i \in \mathcal{S} \), respectively. More precisely, the objective of DSE is that each node \( i \in \mathcal{N} \), fusing local information and information from adjacent nodes, compute an estimate \( \hat{x}^i_{t|t} \) of \( x_t \) with mean squared estimation error as close as possible to the one that would be obtained with a centralized filter that processes the measurements from all sensors.

III. BACKGROUND

A. Covariance intersection

Let us consider a set of \( N \) agents (nodes) and let agent \( i \) be provided with the estimate-covariance pair \( (\hat{x}^i, P^i) \) of the quantity of interest (e.g., but not necessarily, the state vector of a mobile target given measurements distributed tracking (DSE), has a variety of applications such as for instance distributed tracking, which consists of estimating the kinematic state vector of a mobile target given measurements from multiple non co-located position sensors.

provided by the SEN nodes. It is assumed that \( w_t \) and \( v^i_t \), for all \( i \in \mathcal{S} \), are mutually uncorrelated white noises with zero mean and covariances \( Q > 0 \) and \( R^i > 0 \), \( i \in \mathcal{S} \), respectively. More precisely, the objective of DSE is that each node \( i \in \mathcal{N} \), fusing local information and information from adjacent nodes, compute an estimate \( \hat{x}^i_{t|t} \) of \( x_t \) with mean squared estimation error as close as possible to the one that would be obtained with a centralized filter that processes the measurements from all sensors.

III. BACKGROUND

A. Covariance intersection

Let us consider a set of \( N \) agents (nodes) and let agent \( i \) be provided with the estimate-covariance pair \( (\hat{x}^i, P^i) \) of the quantity of interest (e.g., but not necessarily, the state vector of the target dynamical system). The objective is to appropriately fuse all \( \{(\hat{x}^i, P^i): i = 1, 2, \ldots, N\} \) into \( (\hat{x}, P) \).

Julier and Ullmann [9], [10] introduced the covariance intersection fusion mechanism. Covariance intersection works on the agent information pairs \( (\Omega^i = (P^i)^{-1}, q^i = (P^i)^{-1}\hat{x}^i) \) providing the fused information pair \( (\Omega, q) \) according to the following convex combinations:

\[
\begin{align*}
\Omega &= \sum_{i=1}^{N} \pi^i \Omega^i, \\
q &= \sum_{i=1}^{N} \pi^i q^i
\end{align*}
\]

\( \sum_{i=1}^{N} \pi^i = 1, \quad \pi^i \geq 0, \quad i = 1, \ldots, N \)

In [9] it has been proved that the covariance intersection fusion (3) guarantees the following consistency condition

\[
P \geq \mathbb{E} \left[(x - \hat{x})(x - \hat{x})^\top\right]
\]

with \( P = \Omega^{-1}, \quad \hat{x} = \Omega^{-1} q \) and \( \mathbb{E} [\cdot] \) denoting expectation, provided that all the estimate-covariance pairs \( (\hat{x}^i, P^i) \) are consistent. As it will be seen later, some DSE algorithms will conveniently use the equivalent information filter form which, instead of the estimate \( \hat{x}^i_{t|t-1} \) and covariance \( P^i_{t|t-1} \), propagates the information (inverse covariance) matrices

\[
\Omega^i_{t|t-1} \triangleq P^i_{t|t-1}^{-1}, \quad \Omega^i_{t|t} \triangleq P^i_{t|t}^{-1}
\]

and the vectors

\[ q^i_{t|t-1} \triangleq P^i_{t|t-1}^{-1} \hat{x}^i_{t|t-1}, \quad q^i_{t|t} \triangleq P^i_{t|t}^{-1} \hat{x}^i_{t|t} \]

that will be referred to as information vectors.

B. Consensus on information

Consensus [1]–[4] is a well known and established technique for distributed averaging over networks and is widely used in distributed parameter/state estimation algorithms. The prototypical consensus problem can be defined as follows. Let node \( i \in \mathcal{N} \) be provided with an estimate \( \hat{\theta}^i \) of a given quantity \( \theta \in \mathbb{R}^{n_\theta} \). The objective is to develop an algorithm that computes in a distributed way, in each node, the average

\[
\hat{\theta} = \frac{1}{|\mathcal{N}|} \sum_{i \in \mathcal{N}} \hat{\theta}^i.
\]

To this end, let \( \hat{\theta}^i_0 = \hat{\theta}^i \), then consensus algorithms take the following general iterative form:

\[
\hat{\theta}^i_{t+1} = \sum_{j \in \mathcal{N}^i} \pi^{i,j} \hat{\theta}^j_t, \quad \forall i \in \mathcal{N}
\]

where the consensus weights must satisfy the conditions

\[
\pi^{i,j} \geq 0 \quad \forall i, j \in \mathcal{N}; \quad \sum_{j \in \mathcal{N}^i} \pi^{i,j} = 1 \quad \forall i \in \mathcal{N}.
\]

Notice from (8)-(9) that at a given consensus step the estimate in any node is computed as a convex combination of the estimates of the neighbors at the previous consensus step. In other words, the iteration (8) is nothing but a regional average computed in node \( i \), the objective of consensus being convergence of such regional averages to the collective average (7). Important convergence properties, depending on the consensus weights, can be found in [2], [3].

An information-theoretic approach to consensus has been proposed in [4]. Assuming that the information of each agent \( i \in \mathcal{N} \) be represented by a PDF \( (\text{Probability Density Function}) p^i(\cdot) \) on some random vector \( x \in \mathbb{R}^{n_x} \) of interest, the objective is to asymptotically converge to an average (in some sense) PDF \( \bar{p}(\cdot) \) of the PDFs \( p^i(\cdot) \). For example, such PDFs can be obtained via statistical inference or can be the result of some recursive Bayesian estimation algorithm (as will be discussed in the following sections). Here, it is supposed that all the local PDFs belong to the same parametric family \( \mathcal{P} = \{p(\cdot) = \pi(\cdot; \theta), \theta \in \Theta \subset \mathbb{R}^{n_\theta}\} \) where \( \pi(\cdot) \) is a given function and \( \theta \) a parameter vector completely characterizing the PDF. In other words, each local PDF can be written as \( p^i(\cdot) = \pi(\cdot; \theta^i) \) for some vector \( \theta^i \in \Theta \). It has been shown (the interested reader is referred to [4]) that, taking into account the continuous Kullback-Leibler divergence (KLD) or relative entropy between two PDFs \( p^i(\cdot) \) and \( p^j(\cdot) \)

\[
D_{KL}(p^i||p^j) = \int p^i(x) \log \frac{p^i(x)}{p^j(x)} dx,
\]
the Kullback-Leibler average (KLA) \( \bar{p}(\cdot) \) of the PDFs \( p^i(\cdot) \), \( i \in \mathcal{N} \), is defined as follows
\[
\bar{p} = \arg \inf_{p \in \mathcal{P}} \frac{1}{|\mathcal{N}|} \sum_{i \in \mathcal{N}} D_{KL}(p||p^i) \quad (10)
\]

Let us now assume that each local PDF \( p^i(\cdot) \) is Gaussian with mean \( \hat{x}_i \) and (positive definite) covariance \( \Sigma_i \). Then, the KLA of such Gaussian PDFs can be simply obtained by averaging their information matrices and information vectors defined in (5) and (6). An important consequence of this state fusion rule is that the KLA of such Gaussian PDFs can be simply obtained by applying one of the many existing consensus algorithms to the agent information pairs, which corresponds to performing multiple iterations of the covariance intersection fusion rule (compare (3) with (8)).

Let us now assume that, at time \( t \), each node \( i \in \mathcal{N} \) be provided with a local estimate-covariance pair \((\hat{x}^i_{t|t-1}, \Sigma^i_{t|t-1})\). The Consensus on Information (CI) approach to DSE is summarized by the following algorithm to be carried out at each sampling interval \( t \in \mathbb{N} \) approach to DSE is summarized by the following

**CI DSE Algorithm**

1) If \( i \) is a SEN node, sample the measurement \( y^i_t \) and apply to \((\hat{x}^i_{t|t-1}, \Sigma^i_{t|t-1})\) the correction step so as to obtain \((\hat{x}^i_{t|t}, \Sigma^i_{t|t})\). Otherwise, if \( i \) is a COM node, set \( \Omega^i_{t|t-1} = \Sigma^i_{t|t-1} \) and \( x^i_{t|t} = \hat{x}^i_{t|t-1} \).
2) Set \( \Omega^i_{t|t} = (\bar{P}^i_{t|t})^{-1} \) and \( q^i_{t|t} = (\bar{P}^i_{t|t})^{-1} x^i_{t|t}. \)
3) For \( \ell = 0, \ldots, L-1 \) perform the following consensus steps:
   - transmit \( q^i_{\ell|\ell+1} \) and \( \Omega^i_{\ell|\ell+1} \) to all adjacent nodes \( j \in \mathcal{N} \setminus \{i\} \);
   - wait until \( q^j_{\ell|\ell+1} \) and \( \Omega^j_{\ell|\ell+1} \) have been received from all adjacent nodes \( j \in \mathcal{N} \setminus \{i\} \);
   - fuse the quantities \( q^j_{\ell|\ell+1} \) and \( \Omega^j_{\ell|\ell+1} \) according to
     \[
     q^j_{\ell|\ell+1} = \sum_{j \in \mathcal{N}} \pi^{i,j} q^j_{\ell|\ell+1}, \quad \Omega^j_{\ell|\ell+1} = \sum_{j \in \mathcal{N}} \pi^{i,j} \Omega^j_{\ell|\ell+1} \quad (11)
     \]
4) Compute the filtered estimate-covariance pair:
   \[
   \bar{P}^i_{t|t} = [\Omega^i_{t|t} L]^{-1} \quad \bar{x}^i_{t|t} = [\Omega^i_{t|t} L]^{-1} q^i_{t|t}
   \]
5) Apply the prediction step to \((\bar{x}^i_{t|t}, \bar{P}^i_{t|t})\) so as to obtain \((\hat{x}^i_{t+1|t}, \Sigma^i_{t+1|t})\).

It is worth to point out that the above CI algorithm reduces to the well known covariance intersection [9] for a single \( L = 1 \) consensus step. It represents, therefore, a generalization of covariance intersection to multiple \( L > 1 \) consensus steps which can be introduced in order to get closer to the optimal MSE performance of the Centralized Kalman Filter (CKF).

**C. Centralized multiple-model algorithms**

Let us now consider state estimation for the jump Markovian system
\[
x_{t+1} = f(m_t, x_t) + w_t \quad (12)
\]
where \( m_t \in \{m^j\}_{j=1}^r \) denotes the modal state at time \( t \) which represents the mode in which the system is operating at time \( t \). It is assumed that the system can operate in \( r \) possible modes \( m^1, m^2, \ldots, m^r \), each mode \( m^j \) being characterized by a mode-matched model with state-transition function \( f(m^j, \cdot) \) and process noise covariance \( Q(m^j) \). Further, mode transitions are modelled by means of a homogeneous Markov chain with suitable constant transition probabilities
\[
p_{jk} = \text{prob} (m_{t+1} = j | m_{t} = k), \quad j, k \in \{1, 2, \ldots, r\} \quad (13)
\]

From now on, the event \( m_t = m^j \) will be more simply denoted as \( m^j_t \). It is well known that a jump Markovian system (12)-(13) can effectively model the motion of a maneuvering target [5, section 11.6]. For instance a nearly-constant velocity model can be used to describe straight-line target motion while coordinated turn models with different angular speeds can describe target maneuvers. In alternative, models with different process noise covariances (small for straight-line motion and larger for target maneuvers) can be used with the same kinematic transition function \( f(\cdot) \). For multimodal systems, the classical single-model filtering approach is clearly inadequate. To achieve better state estimation performance, the multiple model (MM) filtering approach must be adopted. The interested reader is referred to [5, section 11.6] for a thorough treatment of MM filtering. In the sequel, the two most commonly used MM algorithms, i.e. First Order Generalized Pseudo-Bayesian (GPB1) and Interacting Multiple Model (IMM), will be briefly reviewed. Both algorithms run in parallel with mode-matched filters (one for each mode) on the same input measurements.

In this section, it is shown how GPB1 and IMM work in a centralized scenario, wherein the measurements of all sensors are gathered and jointly processed (COM nodes being ignored) in order to carry out the correction step in the mode-matched filters.

**GPB1 Algorithm**

Assume that the state estimate \( \hat{x}_{t-1|t-1} \), the covariance \( P_{t-1|t-1} \) and the mode probabilities \( \mu^j_{t-1} \) \( (j = 1, \ldots, r) \) have
been obtained at time step \( t - 1 \), then the algorithm proceeds as follows at each time step \( t \).

1. **Mode-matched filtering**: starting from the state estimate \( \tilde{x}_{t-1|t-1} \) and its covariance \( P_{t-1|t-1} \), for each \( j \in \{1, \ldots, r\} \), the associated mode-matched filter carries out the prediction and correction steps, using all the measurements \( y_i^t \), \( i \in S \). The output of each model-matched filter is the mode-conditioned state estimate \( \tilde{x}^j_{t|t} \) and the associated covariance \( P^j_{t|t} \). Then, assuming Gaussian noises, the mode likelihoods, corresponding to the \( r \) mode-matched filters, are evaluated as follows:

\[
\Lambda^j_{t} = \mathcal{N}(\nu^j_t; 0, S^j_t), \quad j = 1, \ldots, r
\]

where \( \nu^j_t \triangleq y^t - \mathbf{h}^{j}(\tilde{x}^j_{t-1}) \) is the innovation at time \( t \) of mode \( j \) and \( S^j_t \) the corresponding covariance.

2. **Mode probability update**: assuming that mode transitions are modelled by an homogeneous Markov chain, with jump probabilities defined in (13), the mode probabilities \( \mu^j_t \triangleq \text{prob} \left( m^t \in \{y^t, \tau \leq t-1, i \in S\} \right) \) are evaluated as follows.

\[
\mu^j_t = \frac{1}{c} \Lambda^j_{t} \sum_{k=1}^{r} p_{jk} \mu^k_{t-1}
\]

where \( c = \sum_{j=1}^{r} \Lambda^j_{t} \sum_{k=1}^{r} p_{jk} \mu^k_{t-1} \) is the normalization constant.

3. **State estimate and covariance combination**: the target state estimate and its covariance are obtained as convex combinations in the following way:

\[
\begin{align*}
\tilde{x}_{t|t} &= \sum_{j=1}^{r} \mu^j_t \tilde{x}^j_{t|t} \\
P_{t|t} &= \sum_{j=1}^{r} \mu^j_t \left\{ P^j_{t|t} + \left[ \tilde{x}^j_{t|t} - \tilde{x}_{t|t} \right] \left[ \tilde{x}^j_{t|t} - \tilde{x}_{t|t} \right]^T \right\}
\end{align*}
\]

**IMM Algorithm**

IMM is identical to GPB₁, except for the initialization of the mode-matched filters; in fact each filter is initialized with a different combination of the previous model-conditioned estimates \( \tilde{x}^j_{t-1|t-1} \) and the associated covariances \( P^j_{t-1|t-1} \), the so called mixed initial conditions. Hence, before carrying out the same three steps of GPB₁, two further steps are required by IMM.

1. **Calculation of the mixing probabilities**: in order to calculate the mixed initial conditions, the required mixing probabilities are updated as follows:

\[
\mu^j_{t-1|t-1} \triangleq \frac{p_{jk} \mu^k_{t-1}}{r} \sum_{h=1}^{r} p_{jh} \mu^h_{t-1}
\]

2. **Mixing**: the mixed initial conditions for the mode-matched filters are computed as follows for \( j = 1, 2, \ldots, r \):

\[
\begin{align*}
\tilde{x}^{0j}_{t-1|t-1} &= \sum_{k=1}^{r} \mu^k_{t-1|t-1} \tilde{x}^k_{t-1|t-1} \\
P^{0j}_{t-1|t-1} &= \sum_{k=1}^{r} \mu^k_{t-1|t-1} \left\{ P^k_{t-1|t-1} + \left[ \tilde{x}^k_{t-1|t-1} - \tilde{x}^{0j}_{t-1|t-1} \right] \left[ \tilde{x}^k_{t-1|t-1} - \tilde{x}^{0j}_{t-1|t-1} \right]^T \right\}
\end{align*}
\]

**IV. Distributed multiple-model algorithms**

In the distributed sensor network scenario of section II, the aforementioned multiple-model filtering algorithms need to be modified. The main difficulty arises from the presence of COM nodes. In fact the latter, due to the lack of measurements, cannot compute the likelihoods (14) and consequently update the mode probabilities according to (15). Further, the possible lack of complete observability from a single sensor may make it impossible to reliably estimate the system mode on the sole grounds of local information. To get around these problems, the idea is to carry out consensus even on the mode probabilities. In this way, GPB₁ and IMM filtering can correctly be performed also in the COM nodes, that otherwise should operate with constant measurement-independent modal probabilities.

Before proceeding with the details of distributed multiple-model algorithms, it is worth pointing out two major differences between GPB₁ and IMM.

**D1.** The fused state estimate \( \tilde{x}^j_{t|t} \) and associated covariance \( P^j_{t|t} \) are not mandatory for the time propagation of the IMM filter. In fact, the mixing step of the IMM algorithm directly uses the mode-conditioned state estimates \( \tilde{x}^j_{t|t} \) and covariances \( P^j_{t|t} \) to initialize the mode-matched filters, while GPB₁ reinitializes in the same way all mode-matched filters with the fused state estimate and covariance.

**D2.** Whenever a SEN node is unable to guarantee complete local observability, like e.g. range-only and bearing-only sensors, its local IMM filter must perform consensus on mode-conditioned state estimates \( \tilde{x}^j_{t|t} \) and associated covariances \( P^j_{t|t} \) in order to achieve observability [4] and then properly use such mode-conditioned statistics for reinitializing mode-matched filters. Conversely, for the GPB₁ filter this can be accomplished by exploiting consensus on either mode-conditioned or fused estimates-covariances.

As explained in section III, to distribute the available information (essentially state estimates and covariances) over the network, the so called “consensus on information” will be adopted. This type of consensus can be implemented at two different granularity levels of the MM algorithms. It can be applied at the combined information level, i.e. on estimates-covariances resulting from (16), or at the modal information level, i.e. on estimates-covariances resulting from the correction steps of the mode-matched filters. To better suit the logic of each MM algorithms to a distributed scenario, the Consensus on Combined Information (CCI) will be adopted for the distributed GPB₁ filter while the Consensus on Mode-
conditioned Information (CMI) will be adopted for the distributed IMM filter.

For what concerns the consensus on mode probabilities, (10) is applied, but taking into account the discrete-time KLD between a pair of PMFs (Probability Mass Functions) $p^i_j = [\mu^i_j, \ldots, \mu^i_{j+r}]'$ and $p^j_i = [\mu^j_i, \ldots, \mu^j_{i+r}]'$ ($i, j \in N$), defined as follows

$$D_{KL}(p^i_j||p^j_i) = \sum_{k=1}^{r} \mu^i_k \log \frac{\bar{\mu}^i_k}{\bar{\mu}^j_k},$$

where the PMFs $p^i_j$ and $p^j_i$ represent mode PMFs in the nodes $i$ and $j$. Defining $\bar{\mu} = (\bar{\mu}^1, \ldots, \bar{\mu}^r)$ as the KLA of all the PMFs $p^i_j$, it can be proved (the proof is omitted here due to lack of space) that the KLA is given by

$$\bar{\mu}^k = \frac{\left( \prod_{i \in N^j} \mu^i_k \right)^{\frac{1}{k}}}{\sum_{h=1}^{k} \left( \prod_{i \in N^h} \mu^i_h \right)^{\frac{1}{h}}},$$

which is the normalized Geometric Mean. Therefore, it is convenient to write the GM in the logarithmic form $\log \bar{\mu}^k = \frac{1}{k} \sum_{i \in N^k} \log \mu^i_k$ (the normalization constant is omitted for simplicity), and then use the consensus iterations (8). Hence, it follows that, $\forall i \in N$, $\log \bar{\mu}^i_k = \sum_{j \in N^i} \pi^{i,j} \log \mu^i_k$, from which

$$i^i_k \ell_{\ell+1} = \prod_{j \in N^i} (i^i_k)^{\pi^{i,j}} \quad (19)$$

Each node, once all the data from neighbors have been received, carries out a certain number of iterations of (19). Afterwards, it performs a normalization so as to ensure that the probabilities sum up to 1, thus obtaining an approximation of the KLA of the mode PMFs.

A couple of remarks are in order.

Remark 1 - In the CCI case, consensus on mode probabilities must be performed before consensus on information. Hence, two sequential phases of data communication are needed. On the other hand, in the CMI case, consensus on mode probabilities and on information can be carried out in parallel.

Remark 2 - On the basis of Remark 1, in the CCI case the consensus requires two distinct data communication phases, but less data than CMI is exchanged throughout the nodes.

Let us now introduce the two proposed distributed multiple-model state estimation algorithms, i.e. Distributed GBP$_1$ (DGBP$_1$) and Distributed IMM (DIMM). In order not to overburden notation, the algorithm operations in each node will be described by omitting the node superscript in the associated node variables.

A. DGBP$_1$ Algorithm

Assume that the state estimate $\hat{x}_{t-1|t-1}$, the covariance $P_{t-1|t-1}$ and the mode probabilities $\mu_j^t$ (for $j = 1, \ldots, r$) are available at time step $t-1$ in each node, then the algorithm proceeds as follows at each time step $t$.

1. Mode-matched filtering: starting from the state estimate $\hat{x}_{t-1|t-1}$ and its covariance $P_{t-1|t-1}$ for each mode $j \in \{1, \ldots, r\}$, the generic node operates as follows according to its type (SEN/COM):

   - SEN nodes carry out the prediction step, for each mode the corresponding mode-matched model, and then the correction step using local measurements;
   - COM nodes carry out only the prediction step, using for each mode the corresponding mode-matched model.

The outputs of the $j$th mode-matched filter are the mode-conditioned state estimate $\hat{x}_{t|t}$ and the associated covariance $P_{t|t}^j$. Then,

SEN nodes evaluate the mode likelihoods according to (14);

COM nodes do not evaluate such likelihoods due to the lack of measurements.

2. Mode probability update: each node updates the mode probabilities as follows according to its type (SEN/COM):

   - SEN nodes use (15);
   - COM nodes use $\mu^j_t = \sum_{k=1}^{r} p_j^k \mu^j_{k-1}$.

3. Consensus on mode probabilities: each node carries out, along with its neighbors, consensus on mode probabilities exploiting (19) for a given number of steps.

4. State estimate and covariance combination: each node computes the combined state estimate and its covariance according to (16) and with the mode probabilities resulting from the previous consensus. Then, it transforms the combined estimate-covariance pair into the information pair $(\Omega_{t|t \cap 0}, q_{t|t \cap 0})$.

5. CCI: each node carries out, along with its neighbors, consensus on information (i.e. step 3 of the CI DSE Algorithm) for a given number of consensus steps and then transforms the information pair resulting from consensus into the covariance pair, i.e. $\hat{x}_{t|t}$ and $P_{t|t}$, by inversion of (5) and (6).

B. DIMM Algorithm

Assume that the model-conditioned estimates $\hat{x}_{t-1|t-1}$, the associated covariance $P_{t-1|t-1}$ and the mode probabilities $\mu_j^t$ (for $j = 1, \ldots, r$) are available at time step $t-1$ in each node, then the algorithm proceeds as follows at each time step $t$.

1. Calculation of the mixing probabilities: each node computes the mixing probabilities according to (17).

2. Mixing: each node computes, for each mode-matched filter, the mixed initial conditions according to (18).

3. Mode-matched filtering: starting from the mixed initial conditions $\hat{x}_{t-1|t-1}^j$ and $P_{t-1|t-1}^{j}$ for each mode $j \in \{1, \ldots, r\}$, the generic node operates as follows according to its type (SEN/COM):

   - SEN nodes carry out the prediction step, for each mode the corresponding mode-matched model, and then the correction step using local measurements;
   - COM nodes carry out only the prediction step, using for each mode the corresponding mode-matched model.

The outputs of the $j$th mode-matched filter are the mode-conditioned state estimate $\hat{x}_{t|t}$ and the associated covariance $P_{t|t}^j$. Then,
SEN nodes evaluate the mode likelihoods associated according to (14); COM nodes do not evaluate such likelihoods due to the lack of measurements.

4. Mode probability update: each node updates the mode probabilities as follows according to its type (SEN(COM):

SEN nodes use (15);
COM nodes use μk = \sum_{k=1}^{r} p_{jk} \mu_{k-1}^j.

5. Consensus: each node carries out, along with its neighbors and in parallel, the following two consensus procedures.

5a. Consensus on mode probabilities: i.e. consensus on mode probabilities exploiting (19) for a given number of steps.

5b. CMI, i.e. consensus on mode-matched information: the estimate-covariance pair of each mode-matched filter is transformed into the corresponding information pair, then consensus on information (i.e. step 3 of the CI DSE Algorithm) is applied for a given number of consensus steps to each mode-matched information pair, and finally the mode-matched information pairs resulting from consensus are transformed back to estimate-covariance pair (\hat{x}_{t|t}, P_{t|t}) for j = 1, ..., r.

6. State estimate and covariance combination: each node computes the combined state estimate and its covariance according to (16) and with the mode probabilities resulting from the previous consensus.

V. SIMULATION CASE-STUDIES

To assess performance of the proposed distributed multiple-model algorithms described in section IV, the 2D tracking scenario of fig. 2 is considered. Notice from fig. 2 that the target state is denoted by x = [x, y, \dot{x}, \dot{y}] where (x, y) and (\dot{x}, \dot{y}) represent the target Cartesian position and, respectively, velocity components. The sampling interval is T_s = 5[s] and the total target navigation time is 640[s], corresponding to 129 sampling intervals. Specifically, r = 5 different Coordinated-Turn (CT) models [5] are used in the MM algorithms. All models have the state dynamics

\[
\dot{x}_{t+1} = \begin{bmatrix}
1 & \sin(\omega T_s) & 0 & -\frac{1}{\omega} \cos(\omega T_s) \\
0 & \cos(\omega T_s) & 0 & -\frac{1}{\omega} \sin(\omega T_s) \\
1 & 1-\cos(\omega T_s) & 1 & \frac{1}{\omega} \sin(\omega T_s) \\
0 & \sin(\omega T_s) & 0 & \cos(\omega T_s)
\end{bmatrix} x_t + w_t
\]

\[
Q = \begin{bmatrix}
\frac{1}{2}T_s^4 & \frac{1}{2}T_s^3 & 0 & 0 \\
\frac{1}{2}T_s^3 & \frac{1}{2}T_s^2 & 0 & 0 \\
0 & 0 & \frac{1}{2}T_s^4 & \frac{1}{2}T_s^3 \\
0 & 0 & \frac{1}{2}T_s^3 & T_s^2
\end{bmatrix}
\]

for five different constant angular speeds \( \omega \in \{-0.0157, -0.058, 0.058, 0.0157\} [\text{rad/s}]. Notice, in particular, that, taking the limit for \( \omega \to 0 \), the model corresponding to \( \omega = 0 \) is nothing but the well known Discrete White Noise Acceleration (DWNA) model [5]. The standard deviation of the process noise is taken as \( \sigma_w = 0.1[m/s^2] \) for the DWNA (\( \omega = 0 \)) model and \( \sigma_w = 0.5[m/s^2] \) for the other models. Jump probabilities (13), for the Markov chain, are chosen as follows:

\[
[p_{jk}]_{j,k=1,...,5} = \begin{bmatrix}
0.95 & 0.05 & 0 & 0 & 0 \\
0.05 & 0.9 & 0.05 & 0 & 0 \\
0.05 & 0.9 & 0.05 & 0 & 0 \\
0 & 0.05 & 0.9 & 0.05 & 0 \\
0 & 0 & 0.05 & 0.9 & 0.95
\end{bmatrix}
\]

All the local mode-matched filters exploit the Unscented Kalman Filter (UKF) [11] with cubature weights [12]. Two different types of consensus weights have been considered. Specifically, the Metropolis weights [3, 13] are adopted for consensus on information while for consensus on mode probabilities, the following entropy weights have been used:

\[
\pi_k^i = \frac{\sum_{h \in N_i} \mu_{j;k} \log \mu_{j;k}}{\sum_{h \in N_i} \sum_{k=1}^{r} \mu_{h;k} \log \mu_{h;k}}.
\]

Entropy weights, instead of Metropolis weights, are employed for consensus on mode probabilities because it is necessary to emphasize lower-entropy mode probabilities from SEN nodes which have been measurement-updated, with respect to higher-entropy mode probabilities from COM nodes which, on the other hand, have been only model-predicted and hence have to be penalized in the consensus.

All simulation results have been obtained by averaging over 300 Monte Carlo trials. The initial conditions have been randomly chosen, for each Monte Carlo trial, in a 300 [m]-neighborhood of the true target initial position with associated initial covariance matrix set equal to \( \text{diag} \{10^6, 10^6, 10^6, 10^4 \} \).

A. Range-only and bearing-only sensor networks

A surveillance area of 50 × 50[km] is considered, wherein two different sensor networks are deployed: the first (see fig. 3) consists of 6 COM nodes and 3 bearing-only (DOA) SEN nodes; the second (see fig. 4) is made up of 100 COM, 5 DOA and 5 range-only (TOA) SEN nodes. The following measurement functions characterize the DOA and TOA sensors:

\[
h^i(x) = \begin{cases}
\sqrt{(x-x_i)^2 + (y-y_i)^2}, & \text{DOA sensor} \\
\sqrt{(x-x_i)^2 + (y-y_i)^2}, & \text{TOA sensor}
\end{cases}
\]

where \( (x_i, y_i) \) represents the known position of sensor \( i \) in Cartesian coordinates. The standard deviation of DOA and
TOA measurement noises are taken respectively as $\sigma_{DOA} = 2[\circ]$ and $\sigma_{TOA} = 100[\mathrm{m}]$. The number of consensus steps has been fixed to $L = 4$ for the first network (equal to the network diameter [2]) and $L = 5$ for the second one (chosen as the largest previous integer of the average degree [2]).

The performance of the various multiple-model algorithms is measured by the *Position Root Mean Square Error (PRMSE)*. Notice that averaging is performed over time and Monte Carlo trials for the centralized algorithms (i.e. CGPB$_1$ and CIMM), while further averaging over network nodes is applied for distributed algorithms (i.e. DGPB$_1$ and DIMM). Table I summarizes performance of the various algorithms using the smaller-scale network of fig. 3. As it can be seen, DIMM performs significantly better than DGPB$_1$ and, due to the presence of COM nodes, the distributed case exhibits remarkably worse performance compared to the centralized case, which must be considered as a lower bound for the achievable PRMSE performance. The high values of the PRMSE are due to the large size of the surveillance area and the fact that for DOA sensors a small angle error induces large position errors at far distances. To provide a visual characterization of tracking performance, fig. 5 compares the true and estimated target trajectories (for the various algorithms) in two different cases: one due to the small fraction of sensor nodes in the network. On the other hand it is apparent that, even in presence of so many communication nodes (bringing no information about the target position) and of a highly maneuvering target, distributed MM algorithms still work satisfactorily by distributing the little available information throughout the network by means of consensus. Clearly, centralized algorithms exhibit significantly better performance since they are not affected by the massive presence of COM nodes. Another reason for the high PRMSE of distributed filters is the difficulty of spreading information on mode probabilities throughout the network due to such a high number of COM nodes and low number of consensus steps $L$. Despite these difficulties, the target can still be tracked with reasonable confidence. Fig. 6 shows the differences between a COM and a SEN nodes that are placed almost at opposite points in the network. It is noticeable that the main difficulty in tracking by COM nodes arises when a turn is approaching, especially, in this trajectory, during the last turn. Therefore, one should use a higher number of consensus steps $L$ to allow data to reach two distant points in the network and, hence, to have similar estimated trajectory and mode probabilities. Again, DIMM is shown to work significantly better than DGPB$_1$; in fact, the PRMSE of DIMM is lower than half than the PRMSE of DGPB$_1$.

### TABLE I: Performance comparison for the bearing-only sensor network of fig. 3

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>PRMSE [m]</th>
<th>PRMSE [m]</th>
</tr>
</thead>
<tbody>
<tr>
<td>CGPB$_1$</td>
<td>1687</td>
<td>DGPB$_1$</td>
</tr>
<tr>
<td>CIMM</td>
<td>1113</td>
<td>DIMM</td>
</tr>
</tbody>
</table>

### TABLE II: Performance comparison for the heterogeneous (DOA-TOA) sensor network of fig. 4

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>PRMSE [m]</th>
<th>PRMSE [m]</th>
</tr>
</thead>
<tbody>
<tr>
<td>CGPB$_1$</td>
<td>178.2</td>
<td>DGPB$_1$</td>
</tr>
<tr>
<td>CIMM</td>
<td>180.2</td>
<td>DIMM</td>
</tr>
</tbody>
</table>

Let us now consider the larger-scale sensor network of fig. 4. The aim, in this case, is to analyze the behavior of a large network with few heterogeneous sensors and many communication nodes. Table II summarizes the performance of the various algorithms in the described scenario. As it can be seen, the estimation errors are higher than in the previous case due to the small fraction of sensor nodes in the network. On the other hand it is apparent that, even in presence of so many communication nodes (bringing no information about the target position) and of a highly maneuvering target, distributed MM algorithms still work satisfactorily by distributing the little available information throughout the network by means of consensus. Clearly, centralized algorithms exhibit significantly better performance since they are not affected by the massive presence of COM nodes. Another reason for the high PRMSE of distributed filters is the difficulty of spreading information on mode probabilities throughout the network due to such a high number of COM nodes and low number of consensus steps $L$. Despite these difficulties, the target can still be tracked with reasonable confidence. Fig. 6 shows the differences between a COM and a SEN nodes that are placed almost at opposite points in the network. It is noticeable that the main difficulty in tracking by COM nodes arises when a turn is approaching, especially, in this trajectory, during the last turn. Therefore, one should use a higher number of consensus steps $L$ to allow data to reach two distant points in the network and, hence, to have similar estimated trajectory and mode probabilities. Again, DIMM is shown to work significantly better than DGPB$_1$; in fact, the PRMSE of DIMM is lower than half than the PRMSE of DGPB$_1$.

### B. Doppler-only measurements networks

Let us now consider networks composed by communication and Doppler-only (DOP) sensor nodes [14], [15]. Doppler-only...
tracking represents a challenging problem for the following reasons.

- The state remains unobservable before receiving and processing at least three Doppler measurements from different sensors [14]; actually, three measurements are sufficient in ideal noise-free conditions and whenever the target motion is deterministic with constant velocity. Hence, for a highly maneuvering target and in presence of measurement noise, more than three measurements are usually needed to achieve observability.
- DOP measurements are typically very accurate (very small measurement noise standard deviation) so that, because of the initial lack of observability, the first corrections (measurement-updates) are weighted too much and, in turn, low-complexity nonlinear filters, such as UKF or the Extended Kalman Filter, turn out to be unreliable and often diverge.
- If the initial target state is unknown (as in most real scenarios), the initialization of the filtering algorithms becomes a serious and tricky challenge as setting high initial covariances on position and velocity amplifies the aforementioned problem.

Despite these intrinsic difficulties, let us anyway consider DOP sensors characterized by the following measurement function:

$$h^i(x) = -\frac{f_c}{c} \frac{(x - x^i) \dot{x} + (y - y^i) \dot{y}}{\sqrt{(x - x^i)^2 + (y - y^i)^2}}$$

where $f_c = 0.9 [GHz]$ is the carrier frequency, $c$ is the speed of light and the assumed standard deviation of the measurement noise is $\sigma_{DOP} = 0.5 [Hz]$. In the simulation experiments, the initial state estimate is taken as $x_0 = [2600, 0, 4000, 0]^T$ (approximately 1.5[km] away from the true target initial position) with associated covariance matrix $\text{diag} \{2.5^5, 10^4, 2.5^5, 10^4\}$. Two Doppler-only sensor networks have been considered: one is identical to the network of fig. 3 but with the 2 DOA sensors replaced by DOP sensors; the other just consists of the same 3 DOP sensors, directly connected without communication nodes. Table III reports the performance of the proposed algorithms in the described Doppler-only tracking scenario. The reason why the PRMSEs are so small is due to high Doppler sensor accuracy and because the initialization is not too far from the true initial target position. As it can be seen, the proposed algorithms are able to deal with Doppler sensors even if the target is highly maneuvering and by exploiting the UKF in each mode-matched filter instead of the computationally more expensive particle filter used in [14], [15].

**VI. CONCLUSION**

The paper has presented novel distributed consensus-based multiple-model filters for jump Markov nonlinear systems over a network of heterogeneous sensors and communication nodes. Target tracking simulation case studies have demonstrated the effectiveness of the proposed algorithms. Future work will address smart initialization issues to cope with situations characterized by no prior knowledge on the target and weak target observability. A further topic of investigation will concern the far more complicated multitarget tracking problem in the same distributed sensor network setting.

**REFERENCES**


**TABLE III: Performance comparison for the Doppler-only tracking scenario:**

<table>
<thead>
<tr>
<th>(a)</th>
<th>(b)</th>
<th>(c)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRMSE [m]</td>
<td>PRMSE [m]</td>
<td>PRMSE [m]</td>
</tr>
<tr>
<td>CGPB1</td>
<td>20.17</td>
<td>DGPB1</td>
</tr>
<tr>
<td>CIMM</td>
<td>19.63</td>
<td>DIMM</td>
</tr>
</tbody>
</table>

Fig. 6: Estimated target trajectories of node 1 (COM, green dash-dotted line) and node 110 (DOA, red dashed line) in the sensor network of fig. 4. The real target trajectory is the blue continuous line.