Cooperative Localization for Multiple Soccer Agents Using Factor Graphs and Sequential Monte Carlo

GUILHERME C. G. FERNANDES\textsuperscript{1,2}, (Member, IEEE), STIVEN S. DIAS\textsuperscript{2}, (Member, IEEE), MARCOS R. O. A. MAXIMO\textsuperscript{3}, (Member, IEEE), AND MARCELO G. S. BRUNO\textsuperscript{1}, (Senior Member, IEEE)

\textsuperscript{1}Instituto Tecnológico de Aeronáutica, São José dos Campos 12228-900, Brazil
\textsuperscript{2}Embraer S.A., São José dos Campos 12227-901, Brazil
\textsuperscript{3}Autonomous Computational Systems Lab (LAB-SCA), Computer Science Division, Instituto Tecnológico de Aeronáutica, São José dos Campos 12228-900, Brazil

Corresponding author: Guilherme C. G. Fernandes (gcosta@ita.br)

This work was supported in part by the São Paulo Research Foundation (FAPESP) under Grant 2018/26191-0. The work of Guilherme C. G. Fernandes was supported by the Fundação Casimiro Montenegro Filho.

ABSTRACT
This paper addresses the cooperative localization problem for a multiagent system in the framework of belief propagation. In particular, we consider the RoboCup 3D Soccer Simulation scenario, in which the networked agents are able to obtain simulated measurements of the distance and bearing to both known landmarks and teammates as well as the direction of arrival (DOA) of messages received from allies around the field. There are, however, severe communication restrictions between the agents, which limit the size and periodicity of the information that can be exchanged between them. We factorize the joint probability density function of the state of the robots conditioned on all measurements in the network in order to derive the corresponding factor graph representation of the cooperative localization problem. Then we apply the sum-product-algorithm (SPA) and introduce suitable implementations thereof using hybrid Gaussian-Mixture Model (GMM) / Sequential Monte Carlo (SMC) representations of the individual messages that are passed at each network location. Simulated results show that the cooperative estimates for position and orientation converge faster and present smaller errors when compared to the non-cooperative estimates in situations where agents do not observe landmarks for a long period.

INDEX TERMS
Cooperative localization, distributed estimation, factor graphs, message passing, RoboCup Soccer 3D.

LIST OF ACRONYMS
DOA Direction of arrival
FG Factor graph
FoV Field-of-view
GMM Gaussian-Mixture Model
LS Least squares
ML Maximum likelihood
PL Predict-landmark
PLH Predict-landmark-hearing
PLHA Predict-landmark-hearing-allies
RMSE Root mean square error
SMC Sequential Monte Carlo
SoS System-of-systems
SPA Sum-product algorithm
TOA Time of arrival

I. INTRODUCTION
In several modern engineering applications, multiple agents with sensing, processing, communication and actuating means cooperate towards a common objective. Specifically, the autonomous agents – nodes of an underlying communication network – rely on message passing schemes to build a shared internal representation of the external observed scenario and make decisions on how to proceed to accomplish their mission goals under the perceived scenario circumstances [1].

Multiagent applications usually demand different system-of-systems (SoS) capabilities [2], for instance, agents may be
required to track objects of interest (such as targets in surveillance applications) or continuously map their surrounding environment. In particular, the ability to estimate and share their own positions is an essential perception capability that agents must have in typical military and civilian applications. In this context, fully-distributed cooperative localization methods, e.g. in [3]–[15], are of special interest in practical applications since they often scale with the number of network nodes and do not rely on either a data fusion center or a central system administrator [5] – a single point of failure in conventional, centralized architectures.

In this paper, we focus on the cooperative localization problem considering a sensing environment based on the RoboCup 3D Soccer Simulation domain [16], in which individual humanoid soccer players are able to indirectly observe their corresponding locations with respect to (w.r.t.) known and uniquely identified landmarks, e.g. soccer field corners or goalposts, and w.r.t. allied players. Moreover, each individual player obtains information about its desired displacement from its omnidirectional walking engine.

More specifically, players can measure the direction of arrival (DOA) of incoming messages transmitted by their allies scattered across the soccer field as well as the range and bearing to both allies and landmarks that are detected and unambiguously identified in their cameras’ field-of-view (FoV). RoboCup rules (refer to [16]) impose particular rates at which each robot takes new measurements and specify how often one agent may broadcast information to its teammates.

The communication limitation imposed by the rules is a major concern, since the agents only transmit unidirectionally, at pre-allocated time slots and a limited number of bytes per slot. An algorithm for such a scenario cannot require multiple information exchange between robots at the time instants where mutual measurements are assimilated.

A. RELATED WORK

Estimation techniques are usually categorized as Bayesian and non-Bayesian. Two frequently employed non-Bayesian algorithms, least squares (LS) and maximum likelihood (ML), as well as their distributed implementation, are reviewed in [4]. For an extensive review of non-Bayesian localization techniques for wireless sensor networks, see also [17].

The main Bayesian techniques for sequential state estimation are based on the standard Kalman filter [18]–[20], and Sequential Monte Carlo (SMC) filters, also known as particle filters [21], [22]. Concerning algorithms for robot self-localization, they often rely on cameras to detect landmarks and use them as references for global positioning. In robot soccer, the most commonly used landmarks are goalposts and field lines, especially corners and intersections [23]. Reference [24] presents a survey of Kalman-based filters for robot vision applications – e.g. localization, motion control and tracking – whereas seminal work on particle filter based algorithms for robotics is presented in [25], [26]. Dellaert et al. [26] remark that, in general, the particle filter approach is more suitable for robot global positioning in scenarios where there can be ambiguities, such as robot soccer, due to the Kalman filter’s limitations on propagating a multimodal probability density function. An application of Monte Carlo localization in robot soccer is shown in [27].

The approaches mentioned in the previous paragraph are non-cooperative, i.e. the scenario consists of a single robot taking measurements from landmarks in its surroundings and previous information on the map is known by the robot. Collaborative multi-robot localization was proposed in [9]–[15], emphasizing the need of decentralized approaches in face of the downsides of a fusion center (introduction of a single point of failure in the system), as well as the high processing and communication costs. Fox et al. [9] propose a method where each robot must keep a factorial representation, that is, its own local belief. In particular, those beliefs are stored as sets of particles, following the lead from previous works in Monte Carlo, single robot localization. During localization, detections that a robot makes about other ones are used to introduce additional probabilistic constraints, which are transmitted by the observer robot to the observed robot so that the latter can update its self belief.

An EKF-based approach is proposed in [10], whose main challenge is the need of all-to-all communication at all time instants to maintain local estimates of the cross-correlations and coupling between agents. Scenarios with communication restrictions were explored in [11], where persistent connectivity between the robots is relaxed, but they still need full duplex communication when updating their beliefs. A different approach based on consensus is proposed in [12], where agents are required to iteratively exchange information between themselves whenever a mutual measurement is taken.

A belief propagation framework for formulating the cooperative localization problem is proposed in [5] and explored in [13]–[15]. Each robot can be naturally represented as one node of a factor graph where a local belief is stored and propagated to the other nodes according to a message passing scheme defined by the sum-product algorithm (SPA) (see [28]–[31] for further details). Different implementations arise from this framework. For example, a parametric message passing scheme is proposed in [13], where the functions that specify the observation model are linearized and parameterized as Gaussian functions, and the computation of complex integrals in the SPA is reduced to simpler operations on their distribution parameters. On the other hand, a terrain-aided navigation is presented in [14], where the integrals are computed using an SMC method. The resulting algorithm has the advantage of representing the underlying probability distributions more precisely, which can be an advantage in scenarios where there can be ambiguities, at the cost, however, of being computationally heavier. The factor graph formulation is also used in [15] to model and address specific...
problems of time of arrival (TOA) based systems, such as clock synchronization among others.

Other cooperative approaches have arisen in robot soccer in the sense that the team jointly tracks objects in the field, such as the ball, and then an individual agent uses the common view to improve its self-estimate. In particular, [32] presents a cooperative localization estimator based on least squares, which stores a batch of observations and then optimizes over the possible states. Reference [33] presents a real-time algorithm for tracking the ball using Bayesian sensor fusion and [34] presents an online and decentralized approach using particle filters to jointly perform object tracking and cooperative localization.

B. CONTRIBUTIONS AND PAPER OUTLINE

The main contribution of this paper is to propose a cooperative self-localization algorithm in an environment with communication restrictions, where the relative measurements are taken with cameras and DOA sensors as defined in [16]. Differently from [12], [15], those measurements do not require clock synchronization between the agents, although vision-based localization imposes its own challenges, such as the ambiguity the environment may present [9] and the fact the robots’ vision can be blocked by other robots [23]. The way the camera and other sensor data are preprocessed to obtain useful information, either from landmarks or other teammates, is not in the scope of this paper.

We use the factor graph framework for translating our network topology into a belief propagation algorithm, in which the agents cooperatively propagate their marginal posterior distributions (or local beliefs) regarding their individual poses, i.e. the robots’ positions and orientations w.r.t. a two-dimensional Cartesian coordinate system bonded to the soccer field. Unlike previous works, we do not assume that the robots are capable of communicating with any other player in their team at any time instant, especially when assimilating cooperative measurements. Instead, the proposed belief propagation algorithm is designed to abide by the communication constraints imposed by RoboCup rules, where only one agent may transmit a limited amount of data at each time instant reserved for the team communication.

Additionally, we provide a feasible implementation of the SPA based on hybrid Gaussian-Mixture Model (GMM) / Sequential Monte Carlo representations of the propagated beliefs, which are more suited for scenarios where there can be ambiguities [9], [26]. While the local computations are performed using particle methods, the communication cost of the beliefs are substantially reduced by parameterizing them before transmission.

Compared to non-cooperative methods for localization, which rely solely on local data, the incorporation of mutual measurements to the self-localization problem reduces the number of detected landmarks required by a robot to achieve a single modal estimate and, as verified in this paper, leads to a significant improvement in the convergence of the errors and to higher accuracy in scenarios where the robots remain a long time without detecting any landmarks. Compared to other cooperative localization methods, our approach does not require that agents exchange information between themselves when they collect mutual measurements.

1) PAPER OUTLINE

The paper is divided into six sections. Sec. I is this Introduction. In Sec. II, we describe the signal model. In the sequel, we introduce the proposed factorization of the joint posterior distribution of the individual robot poses conditioned on all available network measurements and derive its corresponding factor graph (FG) representation. In Sec. III, we present the message passing schedule for our cyclic FG. In Sec. IV, we detail our implementation of the SPA using the hybrid GMM / SMC approximation. Simulations and comparison of different scenarios are shown in Sec. V and conclusions are presented in Sec. VI.

2) NOTATION

We denote scalars by \( a \) or \( A \), vectors by \( a \) and matrices by \( A \). \( (\cdot)^T \) denotes matrix transpose. We denote random scalars and vectors respectively by \( A \) and \( A \), which can be distinguished from deterministic scalars or matrices in context. The probability density function (p.d.f.) \( p \) evaluated at \( a \) is denoted by \( p(a) \). The p.d.f. of a multivariate real-valued Gaussian distribution parameterized by the mean vector \( \mu \) and the covariance matrix \( \Sigma \) is represented as \( f_{\mathcal{N}}(x; \mu, \Sigma) \).

II. PROBLEM STATEMENT

A. SIGNAL MODEL OF THE ROBOT DYNAMICS AND MEASUREMENTS

We assume a system with \( R \) humanoid robot soccer players (or agents). At discrete-time instant \( n \), the state of agent \( r \) is a three-dimensional, real vector \( x_{r,n} \) containing its \((x, y)\) position in the field and its orientation (heading) w.r.t. the \( x \) axis

\[
x_{r,n} = [x_{r,n} \ y_{r,n} \ \psi_{r,n}]^T.
\]

Each robot is equipped with a camera, with which it may measure the distance and bearing to fixed landmarks in the field and to players of the same team, also known as teammates or allies. Those elements are uniquely identified, i.e. there is no wrong data association once they are detected within the camera’s FoV and subsequently identified by the robot. At time \( n \), each individual measurement taken by agent \( r \) of a landmark \( l \) or an ally \( a \) is denoted respectively as \( o_{r \rightarrow l,n} \) or \( o_{r \rightarrow a,n} \). The sets containing the indices of the measured objects are denoted by \( L_{r,n} \) and \( A_{r,n} \), respectively.

Robots are also able to broadcast messages to their team. In that case, each listening agent obtains a measurement of the DOA of that message, which is a straight line from itself to the broadcaster. The DOA measurement of a broadcaster agent \( b \in B_{r,n} \) is denoted as \( \phi_{r \rightarrow b,n} \). In our setup, broadcasting is restricted to a maximum of one agent at once, so that \( B_{r,n} \) is either a unitary or an empty set at each time step. Also, \( B_{r,n} \) is the same for all \( r \), since all robots can hear the broadcaster.
The mathematical modeling of the state dynamics and the sensor measurements is described in the sequel. The sampling time is fixed to a value $T_s$.

1) ROBOT DYNAMICS

We use the dynamic model given in [27], [35], [36]. The system input is the vector $u_{r,n}$ given by

$$u_{r,n} = \left[ \Delta x'_{r,n} \Delta y'_{r,n} \Delta \psi'_{r,n} \right]^T,$$

where each entry is known by agent $r$ and corresponds to the calculated displacement of the robot given by the omnidirectional walking engine of the humanoid. They are given in the local Cartesian ($x'$, $y'$) frame of the robot, which is rotated (counter-clockwise) by an angle $\psi_{r,n}$ w.r.t. the global ($x$, $y$) frame, attached to the field. Therefore, the $x'$ and $y'$ axes are pointed according to the robot’s heading and the robot’s left, respectively.

The robot’s dynamic model is given by

$$X_{r,n+1} = f(X_{r,n}, u_{r,n}) + W_{r,n}, \quad (3)$$

where

$$f(x_{r,n}, u_{r,n}) = \begin{bmatrix} x_{r,n} + \Delta x_{r,n} \\ y_{r,n} + \Delta y_{r,n} \\ \psi_{r,n} + \Delta \psi_{r,n} \end{bmatrix},$$

and the input is converted to the field reference frame according to

$$\begin{bmatrix} \Delta x_{r,n} \\ \Delta y_{r,n} \\ \Delta \psi_{r,n} \end{bmatrix} = \begin{bmatrix} \cos(\psi_{r,n}) & -\sin(\psi_{r,n}) & 0 \\ \sin(\psi_{r,n}) & \cos(\psi_{r,n}) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \Delta x'_{r,n} \\ \Delta y'_{r,n} \\ \Delta \psi'_{r,n} \end{bmatrix}. \quad (5)$$

The state noise $W_{r,n}$ is modeled as a multivariate Gaussian vector with distribution $N(0, Q)$. Conditioning the current state vector on the previous one, we have

$$x_{r,n+1} \sim N(f(x_{r,n}, u_{r,n}), Q), \quad (6)$$

with associated p.d.f. $p_f$ given by

$$p_f(x_{r,n+1} | x_{r,n}) = f_N(x_{r,n+1} ; f(x_{r,n}, u_{r,n}), Q). \quad (7)$$

Lastly, it is worth noting that, under RoboCup rules [16], the robot’s dynamic is updated at every discrete time step.

2) DIRECTION OF ARRIVAL FROM BROADCASTING ALLY

The observation model for the direction of arrival from the broadcasting ally $b$ to robot $r$ is given by

$$\Phi_{r \leftarrow b,n} = h_\phi(X_{r,n}, X_{b,n}) + V^\phi_{r,n}, \quad (8)$$

where $h_\phi(x_{r,n}, x_{b,n}) = \text{atan2}(y_{b,n} - y_{r,n}, x_{b,n} - x_{r,n}) - \psi_{r,n}$

and $V^\phi_{r,n} \sim N(0, \sigma^2_\phi)$. The function atan2 takes into account the signs of the numerator and denominator to compute the arc tangent in the range $(-\pi, +\pi]$. The subtraction by the orientation of the receiving robot is due to the fact that all measurements are taken in local coordinates w.r.t. the robot $r$.

Conditioning the measured direction of arrival on the state of the broadcasting and receiving robots, it follows that

$$\phi_{r \leftarrow b,n} | x_{r,n}, x_{b,n} \sim N(h_\phi(x_{r,n}, x_{b,n}), \sigma^2_\phi), \quad (10)$$

with associated p.d.f. $p_\phi$ given by

$$p_\phi(\phi_{r \leftarrow b,n} | x_{r,n}, x_{b,n}) = f_N(\phi_{r \leftarrow b,n} ; h_\phi(x_{r,n}, x_{b,n}), \sigma^2_\phi). \quad (11)$$

Under RoboCup rules, the broadcast window is open at every two time instants, to a maximum of one broadcasting robot. Then each listener is informed of the broadcaster self belief and takes the opportunity to perform a DOA measurement. The message that an agent can transmit is also limited to 20 bytes in size.

3) CAMERA OBSERVATION

The observation model for the measurement of range and bearing obtained from observing an object $o$, which can be either a landmark or an ally, using the camera is given by

$$O_{r \leftarrow o,n} = \begin{bmatrix} P_{r \leftarrow o,n} \\ V^\psi_{r,n} \\ V_{r,n} \end{bmatrix} = \begin{bmatrix} d(x_{r,n}, x_{o,n}) \\ h_\psi(x_{r,n}, x_{o,n}) \\ V_{r,n} \end{bmatrix}, \quad (12)$$

where

$$d(x_{r,n}, x_{o,n}) = \sqrt{(x_{o,n} - x_{o,n})^2 + (y_{o,n} - y_{o,n})^2} \quad (13)$$

and $h_\psi$ is already defined in (9). The noise in each component is white and Gaussian, with

$$V^\psi_{r,n} | x_{r,n}, x_{o,n} \sim N(0, \sigma^2_\psi) \quad (14)$$

$$V_{r,n} \sim N(0, \sigma^2_{V_{r,n}}). \quad (15)$$

Conditioning the camera observation on the states of the robot $r$ and object $o$, we have

$$o_{r \leftarrow o,n} | x_{r,n}, x_{o,n} \sim N(\mu_{r,o}, \Sigma_{r,o}). \quad (16)$$

where

$$\mu_{r,o} = \begin{bmatrix} d(x_{r,n}, x_{o,n}) \\ h_\psi(x_{r,n}, x_{o,n}) \end{bmatrix}, \quad (17)$$

$$\Sigma_{r,o} = \begin{bmatrix} \sigma^2_{V_{r,n}} & 0 \\ 0 & \sigma^2_\psi \end{bmatrix}. \quad (18)$$

with the associated conditional p.d.f. $p_O$ written as

$$p_O(o_{r \leftarrow o,n} | x_{r,n}, x_{o,n}) = f_N(o_{r \leftarrow o,n} ; \mu_{r,o}, \Sigma_{r,o}). \quad (19)$$

Note that the dependence which the variance of the distance noise $V_{r,n}$ in (14) has on the true distance poses additional complexity, since $x_{r,n}$ is always unknown, and $x_{o,n}$ is known only if $o$ corresponds to a landmark. In the filter described later on, robot $r$ estimates the variance in (14) using local estimates $x_{r,n}$ and $x_{o,n}$ if $o$ is a teammate, or the local estimate $x_{o,n}$ and the known vector $x_{o,n}$ if $o$ is a landmark.

Under RoboCup rules, each agent captures an image from its camera every three time instants, taking the corresponding
measurements to the objects (landmarks and allies) which were in the angular section defined by the camera’s FoV, independently of the range.

We note that the ranging between agents is done using a camera, which does not rely on TOA measurements, as in [12], [15]. Although TOA ranging can yield precise measurements, it relies heavily on clock synchronization between agents, which typically has to be jointly estimated with the robot’s pose.

In order to consolidate all measurements that are taken at time instant \( n \) by robot \( r \), we denote the vector \( y_{r,n} \) as the collection of all measurements \( o_r \leftarrow l, a, n \) and \( \phi_r \leftarrow b, n \), for \( l \in L_r, a \in A_r, n \) and \( b \in B_r \).

Considering a state vector \( X_n \) and an input vector \( u_n \) comprising the individual state vectors \( \{X_{r,n}\}_{r=1}^R \) and input vectors \( \{u_{r,n}\}_{r=1}^R \) of all robots, and an observation vector \( Y_n \), collecting the measurements taken by all robots at time \( n \), we end up with the complete state space model

\[
X_{n+1} = f(X_n, u_n) + W_n \quad (20) \\
Y_n = h^n_r(X_n) + V_n, \quad (21)
\]

where \( f \) is the direct extension of function \( f \) in (4) and \( h^n_r \) is a vector function composed by instances of functions \( h_l \) and \( d \) defined respectively in (9) and (13), whose dimension depends on the number of measurements at time \( n \). The vector \( W_n \) collects the state noises in (3), whereas \( V_n \) collects the observation noises in (8) and (12).

**B. FACTORIZATION AND REPRESENTATION WITH FACTOR GRAPH**

Factor graphs are a graphic way of representing factorized multivariable functions and systematically obtaining the marginalization w.r.t. any of the variables applying the sum-product algorithm, as originally proposed in [28]. References [4], [14] are examples of the application of FG in the context of distributed statistical signal processing.

At time instant \( N \), the conditional p.d.f. of interest for the filtering problem is \( p(x_N | y_{1:N}) \), which can be obtained from the marginalization of the posterior joint p.d.f. denoted by

\[
p(x_{0:N} | y_{1:N}). \quad (22)
\]

where \( x_{0:N} \) denotes the state of all agents \( r \) from time instants 0 to \( N \) and \( y_{1:N} \) denotes all measurements taken by all agents \( r \) from time instant 1 up to instant \( N \). Eq. (22) can be rewritten using Bayes’ Law as

\[
p(x_{0:N} | y_{1:N}) \propto p(y_{1:N} | x_{0:N})p(x_{0:N}), \quad (23)
\]

where the proportionality symbol \( \propto \) indicates that the right-hand side of the expression must be multiplied by a normalization factor such that the function in the left is a valid p.d.f.

Assuming that each agent moves independently and using the Markovian assumption for each individual agent, as presented in the dynamics model (7), \( p(x_{0:N}) \) is factored as

\[
p(x_{0:N} | y_{1:N}) = \prod_{r=1}^R p(x_{r,N} | x_{r,0:N-1})p(x_{r,0:N-1}) = \prod_{r=1}^R \prod_{n=1}^N p(x_{r,n} | x_{r,n-1})p(x_{r,0:N-1}) = \prod_{r=1}^R \prod_{n=1}^N p(x_{r,n} | x_{r,n-1}). \quad (24)
\]

Using the assumptions of conditional independence of each measurement and the models in (10) and (12), the likelihood \( p(y_{1:N} | x_{0:N}) \) is written as

\[
p(y_{1:N} | x_{0:N}) = \prod_{r=1}^R \prod_{n=1}^N p(y_{r,n})p(y_{r,0:N-1}) = \prod_{r=1}^R \prod_{n=1}^N p(y_{r,n} | x_{r,n})p(y_{r,0:N-1}). \quad (25)
\]

We introduce the following short notation for the p.d.f.’s in (24) and (25)

\[
\lambda^P_{r,n} \triangleq \lambda^P_{r,n}(x_{r,n}, x_{r,n-1}), \\
\lambda^L_{r-1,n} \triangleq \lambda^L_{r-1,n}(x_{r,n}), \\
\lambda^L_{r-1,n} \triangleq \lambda^L_{r-1,n}(x_{r,n}), \\
\lambda^H_{r-1,n} \triangleq \lambda^H_{r-1,n}(x_{r,n}, x_{b,n}), \\
\lambda^H_{r-1,n} \triangleq \lambda^H_{r-1,n}(x_{r,n}, x_{b,n}), \\
\lambda^A_{r-1,n} \triangleq \lambda^A_{r-1,n}(x_{r,n}, x_{b,n}), \\
\lambda^A_{r-1,n} \triangleq \lambda^A_{r-1,n}(x_{r,n}, x_{b,n}), \\
\lambda^A_{r-1,n} \triangleq \lambda^A_{r-1,n}(x_{r,n}, x_{b,n}). \quad (29)
\]

Fig. 1 depicts the factor graph corresponding to (23), using the results from (24) and (25), detailing robots \( r \) and \( b \) and their measurements at time instant \( n \). Note that, in this particular example, robots \( r \) and \( b \) are seeing each other in their cameras and robot \( b \) is broadcasting its self-belief. The symbol \( \leftarrow \) in the observations is not commutative, which means, e.g. that robot \( r \) sees \( a \) with its camera, but \( a \) does not see \( r \). Note that the symbol \( \Rightarrow \) in Fig. 1 indicates any restriction on the existence of the preceding factor nodes, i.e. the factor nodes exist whenever the logical proposition following the symbol \( \Rightarrow \) is satisfied.

The decentralized approach, where each agent’s state estimation is performed locally, results in a cyclic factor graph. By contrast, a centralized approach would generate a cycle-free factorization which can compute exact results, but is not scalable with the number of agents. The joint state vector would grow linearly with \( R \), whereas the joint observation vector could grow with \( R^2 \), considering the fully connected scenario. Furthermore, Soccer 3D rules do not allow a direct

\[\text{VOLUME 8, 2020}\]
communication link at all times between players or to a central data fusion processor, which makes any centralized approach unfeasible.

III. MESSAGE PASSING SCHEDULE

The message passing schedule comes directly from the implementation of the SPA following the rules for factor and variable nodes, described in [28], [31]. Due to the mutual measurements, our factor graph contains cycles, which implies that, to the best of our knowledge, there is no systematic procedure to exactly calculate the marginals. In addition, our focus is on the filtering problem, where the self-estimates must be available at each agent for real-time applications, which makes a smoothing approach with backward message passing prohibitive. The algorithm is detailed as follows.

▶ Step 1: predict. Let $m^B_{r,n} (x_{r,n-1})$ and $m^B_{s,n-1} (x_{s,n-1})$ denote the marginal posterior p.d.f.’s (or beliefs) of the poses of robots $r$ and $s$, respectively, stored by robot $r$ at instant $n-1$. Robot $r$ computes first the prediction ($P$) message

$$m^P_{r,n} (x_{r,n}) \propto \int \lambda^P_{r,n} (x_{r,n}, x_{r,n-1}) m^B_{r,n-1} (x_{r,n-1}) dx_{r,n-1}, \quad (30)$$

which is proportional to the predicted belief of its own pose obtained via the Chapman-Kolmogorov equation using the factor (26) defined from the transition p.d.f. in (7).
Step 2: assimilate measurements of landmarks within the camera’s FoV. Next, robot $r$ assimilates the camera range-bearing measurements $\phi_{r \rightarrow l}$ of the observed landmarks $l \in L_{r,n}$. Specifically, it computes the prediction ($P$) plus landmark assimilation ($L$) message

$$m_{r,n}^{PL}(x_{r,n}) \propto m_{r,n}^{P}(x_{r,n}) \prod_{l \in L_{r,n}} m_{r \rightarrow l}^{L}(x_{r,n}),$$

(31)

in which each message $m_{r \rightarrow l}^{L}(x_{r,n})$ represents the likelihood of the individual measurement $\phi_{r \rightarrow l}$ of a landmark $l$ at a known location $x_l$, i.e.

$$m_{r \rightarrow l}^{L}(x_{r,n}) = \lambda_{r \rightarrow l}^{L}(x_{r,n}).$$

(32)

Steps 3 and 4: broadcast self-belief (broadcasting robot) and assimilate DOA measurement (non-broadcaster robots). In the sequel, robot $r$ follows some mutually agreed convention among all network nodes to select the current broadcasting robot. If it is robot $r$’s turn, it broadcasts its locally updated belief $m_{r,n}^{PL}(x_{r,n})$ to its allies across the soccer field and makes the message $m_{r,n}^{PLH}(x_{r,n}) = m_{r,n}^{PL}(x_{r,n})$. Otherwise, it listens for a similar message from the selected broadcasting robot $b \neq r$. Upon receiving the message $m_{b,n}^{PL}(x_{b,n})$, node $r$ measures the DOA $\phi_{r \rightarrow b,n}$ of the incoming message from robot $b$ and assimilates it by computing

$$m_{r,n}^{PLH}(x_{r,n}) \propto m_{r,n}^{PL}(x_{r,n}) \lambda_{r \rightarrow b,n}^{H}(x_{r,n})H \lambda_{r \rightarrow b,n}^{H}(x_{r,n}),$$

(33)

which is given by the product between the local updated belief $m_{r,n}^{PL}(x_{r,n})$ and

$$m_{r \rightarrow b,n}(x_{r,n}) \propto \int \lambda_{r \rightarrow b,n}(x_{r,n}, x_{b,n}) m_{r \rightarrow b,n}^{PL}(x_{b,n}) d x_{b,n},$$

(34)

where $\lambda_{r \rightarrow b,n}^{H}(x_{r,n}, x_{b,n})$ is defined in (28) and $m_{r \rightarrow b,n}^{PL}(x_{b,n})$ is the latest belief that robot $r$ has about robot $b$’s pose at instant $n$. In this case, $m_{r,n}^{PL}(x_{r,n})$ is directly assigned to $m_{r,n}^{PL}(x_{r,n})$, since $b$ has just broadcast the most updated belief it has for itself.

Step 5: extrapolate beliefs. When assimilating the camera range-bearing measurements $\phi_{r \rightarrow a,n}$ of the observed allies $a \in A_{r,n}$, robot $r$ will not have access to the most updated belief of a teammate that had not broadcast at time instant $n$. Therefore, each node locally keeps an extrapolated belief of the other robots, for all $s \in \{1, \ldots, R\} \setminus \{r\}$, computed as

$$m_{r,s,n}^{B}(x_{s,n}) \propto \int p_w(x_{s,n} | x_{s,n-1}) \times m_{r \rightarrow s,n-1}(x_{s,n-1}) d x_{s,n-1},$$

(35)

where $p_w(x_{s,n} | x_{s,n-1})$ denotes the transition p.d.f. of the extrapolation model. In the robot soccer context, a sophisticated model could take into consideration the strategy followed by the team. In our case, however, we adopt a simple random walk model with Gaussian increments in the components of the state vector, in such a way that $p_w(x_{s,n} | x_{s,n-1}) \Delta \lambda_{s,n}^{W}(x_{s,n} | x_{s,n-1}) = \lambda_{s,n}^{W}(x_{s,n} | x_{s,n-1})$.

$$\Sigma_w = \frac{1}{9} \begin{bmatrix} \Delta x_{MAX}^2 & 0 & 0 \\ 0 & \Delta y_{MAX}^2 & 0 \\ 0 & 0 & \Delta \psi_{MAX}^2 \end{bmatrix}. \tag{36}$$

The covariance matrix in (36) is diagonal and the entries $\Delta x_{MAX}^2$, $\Delta y_{MAX}^2$, and $\Delta \psi_{MAX}^2$ satisfy $3 \Delta x_{MAX}^2 = \Delta y_{MAX}^2$, $3 \Delta \psi_{MAX}^2 = \Delta y_{MAX}^2$, and $3 \Delta \psi_{MAX}^2 = \Delta y_{MAX}^2$, resulting in the $1/9$ factor. For a Gaussian random walk model, this guarantees a probability of 99.7% that the displacement on each direction is smaller than the maximum possible displacement.

As in (26), we define the following short notation for $p_w$.

$$p_w(x_{s,n} | x_{s,n-1}) = p_w(x_{s,n} | x_{s,n-1}).$$

(37)

Step 6: assimilate measurement of allies within the camera’s FoV. Then, robot $r$ incorporates the observations of allied (A) robots by computing the message

$$m_{r,n}^{PLHA}(x_{r,n}) \propto m_{r,n}^{PLH}(x_{r,n}) \prod_{a \in A_{r,n}} m_{r \rightarrow a,n}^{A}(x_{r,n}),$$

(38)

where each message $m_{r \rightarrow a,n}^{A}(x_{r,n})$, $\forall a \in A_{r,n}$, is computed as

$$m_{r \rightarrow a,n}^{A}(x_{r,n}) \propto \int \lambda_{r \rightarrow a,n}(x_{r,n}, x_{a,n}) m_{r \rightarrow a,n}(x_{a,n}) d x_{a,n}$$

(39)

with $\lambda_{r \rightarrow a,n}(x_{r,n}, x_{a,n})$ defined in (29) and $m_{r \rightarrow a,n}(x_{a,n})$ representing robot $r$’s belief of robot $a$’s pose at instant $n$.

Step 7: assign new updated belief. Finally, we make the updated belief $m_{r,n}^{U}(x_{r,n}) = m_{r,n}^{PLHA}(x_{r,n})$ and store it along with the remaining robot pose beliefs $m_{r,n}(x_{r,n})$, $s \in \{1, \ldots, R\} \setminus \{r\}$, to be used in the next time step iteration $n+1$. Algorithm 1 summarizes the proposed message passing schedule indicating the SPA procedure running at each network location $r$ at a particular time step $n$. The algorithm is initialized at each network node $r$ assuming that $m_{r,0}(x_{r,0}) = f_{N}(x_{r,0}, \mu_{r,0}, \Sigma_{r,0})$ and $m_{r,0}^{B}(x_{r,0}) = f_{N}(x_{r,0}, \mu_{r,0}, \Sigma_{r,0}), \forall s \in \{1, \ldots, R\} \setminus \{r\}$.

Fig. 2 shows the derived factor graph and message passing scheme, highlighting the communication between two different nodes $r$ and $b$ at instant $n$. The blue and red arrows represent the messages computed inside nodes $r$ and $b$, respectively. The filled arrows show the intranode messages, while the dashed arrow shows one internode message, i.e. a message sent over the communication network. The variables and factors filled in gray and the messages over them represent the extrapolation that each node performs of its teammates.

IV. SMC IMPLEMENTATION OF THE SPA

Due to the nonlinearities in the model and the lack of closed-form analytical expressions for the integrals in the problem, we implement the forward SPA in Algorithm 1 using Monte Carlo representations of the functions that correspond to the messages according to the following steps:

Initialization step: obtain a particle representation of the previous belief. When initializing the algorithm, we directly sample $N_p$ particles of the known distribution $m_{r,0}^{B}(x_{r,0})$ in...
order to obtain its corresponding weighted particle representation \( \{ (w_{r,0}^B, x_{r,0}^p) \}, p \in \{1, \ldots, N_p\} \), where

\[
\begin{align*}
x_{r,0}^p &\sim m_{r,0}^B(x_{r,0}) \\
w_{r,0}^p &= \frac{1}{N_p}, \quad \forall p
\end{align*}
\]

and the notation \( x_{r,0}^p \sim f(x) \) indicates that the sample \( x_{r,0}^p \) is drawn according to the continuous probability distribution specified by the p.d.f. \( f \).

After the algorithm has started running, its last step already returns \( m_{r,n-1}^B(x_{r,n-1}) \) as a weighted particle representation \( \{ (w_{r,n-1}^B, x_{r,n-1}^p) \}, p \in \{1, \ldots, N_p\} \), thus this step is not needed in the subsequent time instants.

**Step 1: predict.** The prediction step consists of evaluating the Chapman-Kolmogorov equation, which can be approximated using the particle representation for \( m_{r,n-1}^B \) according to

\[
m_{r,n}(x_{r,n}) \propto \int \lambda_{r,n}(x_{r,n}, x_{r,n-1})m_{r,n-1}^B(x_{r,n-1})dx_{r,n-1}
\]

\[
\approx \sum_{p=1}^{N_p} w_{r,n-1}^B(x_{r,n-1}^p) \mathcal{N}\left(x_{r,n}; f(x_{r,n-1}^p; u_{r,n}), Q\right).
\]

The approximate representation for message \( m_{r,n}(x_{r,n}) \) in (43) can be interpreted as a multimodal Gaussian, with \( N_p \) modes, each with weight \( w_{r,n-1}^B \) and centered at \( f(x_{r,n-1}^p; u_{r,n}) \), calculated from the previous particle state and the desired displacement. Note, however, that the approximation in (43) differs from the method proposed for the prediction step in [14] where the equivalent to both \( m_{r,n-1}^B(x_{r,n-1}) \) and \( \lambda_{r,n}(x_{r,n}, x_{r,n-1}) \) are approximated by Gaussian or sums.


Algorithm 1 Distributed SPA at Robot $r$

1: procedure SPA($\{o_{r \rightarrow l,n}\}_{l \in L_n}^r$, $\{o_{r \rightarrow a}\}_{a \in A_r}^r$), $m_{r,n,1}^B(x_{r,n-1})$, $m_{r,n-1}^B(x_{r,n-1})$) $\forall r \in \{1, \ldots, R\}$ \n
2: \hspace{1em} \textbf{Step 1: predict} \n
3: \hspace{2em} Compute message $m_{r,n}^p(x_{r,n})$ from the previous belief $m_{r,n}^B(x_{r,n})$ as in (30). \n
4: \hspace{2em} \textbf{Step 2: assimilate landmarks} \n
5: \hspace{2em} for $l \in L_n$ do \n
6: \hspace{3em} Compute the likelihood $m_{r,n,l}^p(x_{r,n})$ as in (32) using the measurement $o_{r \rightarrow l,n}$ of the $l$-th observed landmark at a known location $x_l$. \n
7: \hspace{2em} end for \n
8: \hspace{2em} Compute message $m_{r,n}^{PL}(x_{r,n})$ as in (31) from the predicted message $m_{r,n}^p(x_{r,n})$ and the individual likelihoods $\{m_{l,n}^p(x_{r,n})\}_{l \in L_n}$. \n
9: \hspace{2em} \textbf{Steps 3 and 4: broadcast or assimilate DOA} \n
10: \hspace{2em} Use the team policy to determine the broadcasting node $b$ at instant $n$. \n
11: \hspace{2em} if Broadcasting node $b$ equals $r$ then \n
12: \hspace{3em} Broadcast message $m_{r,n}^{PL}(x_{r,n})$. \n
13: \hspace{3em} Make $m_{r,n}^{PL}(x_{r,n}) = m_{r,n}^p(x_{r,n})$. \n
14: \hspace{2em} else \n
15: \hspace{3em} Receive message $m_{b,n}^{PL}(x_{b,n})$ from remote node $b$ and obtain the DOA measurement $\phi_{r \rightarrow b,n}$. \n
16: \hspace{3em} Make $m_{r,n}^B(x_{r,n}) = m_{b,n}^{PL}(x_{b,n})$. \n
17: \hspace{2em} Compute message $m_{r,n}^{PL}(x_{r,n})$ as in (33) from the messages $m_{r,n}^B(x_{r,n})$ and $m_{r,n}^{PL}(x_{r,n})$. \n
18: \hspace{2em} end if \n
19: \hspace{2em} \textbf{Step 5: extrapolate beliefs} \n
20: \hspace{2em} for $s \in \{1, \ldots, R\} \setminus \{r\}$ do \n
21: \hspace{3em} Compute the extrapolated belief $m_{r|s,n}^B(x_{s,n})$ of node $s$ at node $r$ as in (35) from $m_{r|s,n-1}^B(x_{s,n-1})$. \n
22: \hspace{2em} end for \n
23: \hspace{2em} \textbf{Step 6: assimilate observed allies} \n
24: \hspace{2em} for $a \in A_r$ do \n
25: \hspace{3em} Compute message $m_{r|a,n}^{PLHA}(x_{a,n})$ as in (39) using the extrapolated belief $m_{r|a,n}^B(x_{a,n})$ of the $a$-th ally and the observation $o_{r \rightarrow a,n}$. \n
26: \hspace{2em} end for \n
27: \hspace{2em} \textbf{Step 7: assign new belief} \n
28: \hspace{3em} Compute message $m_{r,n}^{PLHA}(x_{r,n})$ as in (38) from the messages $m_{r,n}^{PL}(x_{r,n})$ and $m_{r|a,n}^{PLHA}(x_{a,n})$. \n
29: \hspace{3em} Make $m_{r,n}^B(x_{r,n}) = m_{r,n}^{PLHA}(x_{r,n})$. \n
30: \hspace{3em} return $m_{r,n}^B(x_{r,n})$, $m_{r,n}^B(x_{s,n})$ \n
31: end procedure



\begin{equation}
\begin{aligned}
\hat{\mu}_{r,n} = \sum_{p=1}^{N_p} w_{r,n}^{L(p)} x_{r,n}^{(p)}, \\
\hat{\Sigma}_{r,n} = \sum_{p=1}^{N_p} w_{r,n}^{L(p)} (x_{r,n}^{(p)} - \hat{\mu}_{r,n})(x_{r,n}^{(p)} - \hat{\mu}_{r,n})^T, \\

\end{aligned}
\end{equation}

which requires the transmission of 9 real numbers (3 for the mean vector and 6 for the covariance matrix, considering its symmetry). For our particular signal model, the heading of an agent is not needed for assimilating the mutual measurements, thus further reducing the bandwidth requirement to 5 real numbers (2 for the mean vector, 3 for the covariance matrix). Considering the limitation of 20 bytes, we have to represent each real number with 4 bytes.

When robot $r$ receives the belief from a broadcaster $b$ at time $n$, $r$ assigns a Gaussian $N(\hat{\mu}_{r,n}, \hat{\Sigma}_{r,n})$ to the belief it has about $b$, $m_{r,n}^B(x_{b,n})$, with $\mu_{r,b,n} = \hat{\mu}_{b,n}$ and $\Sigma_{r|b,n} = \hat{\Sigma}_{b,n}$, as calculated by $b$.

\textbf{Step 4: assimilate DOA measurement (non-broadcaster robots).} Similarly to the assimilation of landmarks, at this point, each particle preserves its state, but its weight $w_{r,n}^{L(p)}$ is changed according to the likelihood of the DOA measurement $\phi_{r \rightarrow b,n}$ given the state of the particle and of the broadcaster ally. However, we only have the parameterized marginal probability distribution of that ally’s state, i.e. $m_{r|b,n}^B(x_{b,n})$. In the particle filter approach, in order to represent the ally’s state, we start by directly sampling $N_p$ particles from $m_{r|b,n}^B(x_{b,n})$ with uniform weights, i.e. we build the weighted

\begin{equation}
w_{r,n}^{L(p)} = \frac{1}{N_p}, \forall p.
\end{equation}

The computational complexity of this step is $O(N_p)$, corresponding to the sampling of new particles.
particle set \( \{(w_{b,n}^{B(q)}, x_{b,n}^{(q)})\}, q \in \{1, \ldots, N_q\} \), where

\[
x_{b,n}^{(q)} \sim m_B w_{b,n}^B(x_{b,n}^{(q)})
\]

(49)

\[
w_{b,n}^B = \frac{1}{N_q}, \quad \forall q.
\]

(50)

A particle representation \( \{(w_{r,n}^{H(p)}, x_{r,n}^{(p)})\}, p \in \{1, \ldots, N_p\} \) of \( m_{PLH}^{PLH}(x_{r,n}) \) is built next by keeping the state \( x_{r,n}^{(p)} \) of the landmark assimilation and updating the weights according to the likelihood of the measurement, i.e.

\[
w_{r,n}^{H(p)} \propto w_{r,n}^{L(p)} \sum_{q=1}^{N_q} w_{b,n}^{B(q)} m_{r,a,n}^B(x_{r,n}^{(p)}, x_{b,n}^{(q)})
\]

\[
= w_{L(p)}^{r,n} \frac{1}{N_q} \sum_{q=1}^{N_q} m_{r,a,n}^B(x_{r,n}^{(p)}, x_{b,n}^{(q)})
\]

(51)

with \( \sum_{p=1}^{N_p} w_{r,n}^{H(p)} = 1 \). Note that the rightmost sum on the right-hand side of (51) is the Monte Carlo approximation of the integral on the right-hand side of (34) using the weighted particle set \( \{(w_{b,n}^{B(q)}, x_{b,n}^{(q)})\} \).

The computational complexity of this step is \( O(N_pN_q) \), corresponding to the update of the weights of the \( N_p \) particles using the \( N_q \) particles of the broadcaster robot.

- **Step 5:** extrapolate beliefs. At time \( n \), each robot \( r \) updates the Gaussian representation \( N(\hat{\mu}_{r,s,n}, \hat{\Sigma}_{r,s,n}) \) of the belief it has about the other robots, \( m_{r,s,a,n}^B(x_{r,n}) \), except for a robot that may have just broadcast. For the zero mean random walk model that we adopt, extrapolation consists of maintaining the mean vector and inflating the covariance matrix, i.e.

\[
\hat{\mu}_{r,s,n} = \hat{\mu}_{r,s,n-1},
\]

(52)

\[
\hat{\Sigma}_{r,s,n} = \hat{\Sigma}_{r,s,n-1} + \Sigma_w.
\]

(53)

- **Step 6:** assimilate measurement of allies within the camera’s FoV. Similarly to the assimilation of DOA, for each observation \( o_{r,a,n} \), we must have a local representation with \( N_q \) particles for the belief of ally \( a \), drawn directly from \( m_{r,a,n}^B \), i.e.

\[
x_{a,n}^{(q)} \sim m_{r,a,n}^B(x_{a,n}^{(q)}),
\]

(54)

\[
w_{a,n}^B = \frac{1}{N_q}, \quad \forall q.
\]

(55)

As in (51), we construct a particle representation of \( m_{PLHA}^{PLHA}(x_{r,n}) \) as the weighted particle set \( \{(w_{r,n}^{H(p)}, x_{r,n}^{(p)})\}, p \in \{1, \ldots, N_p\} \) of the DOA assimilation and updating the weights according to the likelihood of each observation, i.e.

\[
w_{r,n}^{H(p)} \propto w_{r,n}^{L(p)} \prod_{a \in A_{r,n}} \sum_{q=1}^{N_q} w_{a,n}^{B(q)} m_{r,a,n}^A(x_{r,n}^{(p)}, x_{a,n}^{(q)})
\]

\[
= w_{r,n}^{H(p)} \prod_{a \in A_{r,n}} \frac{1}{N_q} \sum_{q=1}^{N_q} m_{r,a,n}^A(x_{r,n}^{(p)}, x_{a,n}^{(q)})
\]

(56)

with \( \sum_{p=1}^{N_p} w_{r,n}^{H(p)} = 1 \). Again, the rightmost sum on the right-hand side of (56) is the Monte Carlo approximation of the integral on the right-hand side of (39) using the weighted particle set \( \{(w_{a,n}^{B(q)}, x_{a,n}^{(q)})\} \).

The computational complexity of this step is \( O(N_pN_qR) \), corresponding to the update of the weights of the \( N_p \) particles using the \( N_q \) particles of a maximum of \( R - 1 \) observed teammates. In practice, however, due to limitations in the camera's FoV, the cardinality of the set \( A_{r,n} \), which collects the measurements \( o_{r,a,n} \) at node \( r \) at any instant \( n \) is typically lower than \( R - 1 \), and especially much lower when \( R \) is large, resulting in a tenable computational cost even when the number of agents grows. In soccer applications in particular, \( R \) is usually limited.

- **Step 7:** assign new updated belief. Message \( m_{r,n}^{PLHA}(x_{r,n}) \) is then adopted as the most recent belief of robot \( r \), i.e. the particle representation of \( m_{r,a,n}^B(x_{r,n}) \) is obtained by keeping the state \( x_{r,n}^{(p)} \) and making the weights \( w_{r,n}^{H(p)} \) equal to \( w_{r,n}^{A(p)} \).

Considering all the assimilation steps, the overall computational complexity of the fully cooperative algorithm is \( O(N_pN_qR + N_pL) \).

V. RESULTS AND DISCUSSION

A. SCENARIO GENERATION

We simulated a team of \( R = 5 \) robots in a single computer, using a simulator developed in Matlab®, applying the timing constraints for moving, communicating and collecting camera measurements specified by the RoboCup 3D Soccer Simulator [16]. There are no opponents in the field. The robot trajectories and all sensor measurements were generated according to the physical models described in Sec. II-A. Both the dynamic and observation noise parameters were empirically estimated by comparing the measurements provided by the Soccer 3D simulator and the corresponding ground truth. More specifically, the noise parameters of the camera include not only the intrinsic noise of this sensor, but also the effect of the coordinate conversion from the image frame to the local frame – whose origin is the vertical projection of the center of the robot’s torso and is oriented according to its heading.

We simulated a total of \( N = 5000 \) discrete time instants, with sampling time \( T_s = 20 \) ms. Each robot \( r \) receives a fixed input sequence \( \{u_{r,n}\}_{n=1:N} \) indicating the desired displacements, given in the local reference frame, for moving around the field. The actual trajectory \( \{x_{r,n}\}_{n=1:N} \) each robot \( r \) performs (ground truth) also depends, however, on the particular realization of the state noise sequence \( \{w_{r,n}\}_{n=1:N} \) as described in (3), with covariance matrix \( \mathbf{Q} \) given by

\[
\mathbf{Q} = 10^{-6} \times \begin{bmatrix}
9.1 & -0.7 & 8.9 \\
-0.7 & 8.7 & 9.9 \\
8.9 & 9.9 & 357.5
\end{bmatrix}.
\]

Fig. 3 shows the evaluated scenario with the initial robot’s positions and orientation. The colored circles indicate the landmarks across the soccer field, where the green and red circles represent respectively the field corners and the goal posts. The sector in “V” emanating from robot 5 illustrates
the coverage area of its camera FoV. It is worth noting that we keep each camera fixed w.r.t. the robot’s torso such that its FoV is always pointing to the robot’s heading. For the sake of simplicity, all teammates and landmarks within the angular section defined by the camera FoV are detected, independently of their range. Also, those objects are treated as points, thus this model does not consider that one object may be occluded by another.

As each agent moves, it collects external measurements as modeled in (8) and (12), whose noise variances are $\sigma_\phi^2 = 2.3 \times 10^{-5}$ rad$^2$ for the DOA measurements and $\sigma_\phi,\text{cam}^2 = 2.3 \times 10^{-5}$ rad$^2$, $\sigma^2 = 1.21 \times 10^{-6}$ for the camera measurements.

Note that the particular sequence of landmarks and teammates detected in the camera depends on the ground truth of each robot, which in turn is random due to the presence of state noise. Nevertheless, we defined the noise-free trajectory of each robot over a circle so that their ground truth poses is periodically roughly repeated.

### B. FILTER SETUP

The filter performance was assessed performing a total of $M = 130$ Monte Carlo simulations, where the sequence of displacements for each agent $\{u_{r,n}\}_{n=1:N}$ is the same, but both the ground truth trajectories $\{x_{r,n}\}_{n=1:N}$ and the measurements taken by a particular robot are different for each of the $M$ runs. The calculations in the filter use a total of $N_p = N_q = 200$ particles for representing each required message.

The initial self belief of each agent is represented by a set of particles drawn from a distribution $\mathcal{N}(\tilde{x}_{r,0}, \Sigma_{r,0})$, where $\tilde{x}_{r,0}$ is one sample of the distribution $\mathcal{N}(x_{r,0}, \Sigma_{r,0})$, centered in the ground truth of agent $r$, $x_{r,0}$. The latter belief is also used by the other teammates to initialize their belief about $r$.

For all agents $r$, we choose the initial covariance matrix as

$$
\Sigma_{r,0} = \begin{bmatrix}
1 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 25
\end{bmatrix}.
$$

(58)

In the prediction step, each robot $r$ uses the input signal $u_{r,n}$ and the covariance matrix $Q$ of the state noise. On the other hand, since each landmark, DOA or ally position measurement can be independently assimilated, we can use different assimilation schemes within the observation step leading to different versions of the proposed filter. Specifically, in order to compare to which degree the cooperation improves the state estimation, we propose four variants of the cooperative filter:
1) **PL.** Non-cooperative filter in which the observation step consists solely of landmark assimilation, similar to [27]. Steps 1 and 2 of Algorithm 1 are performed.

2) **PLH.** Steps 1 to 4 are performed. This version extends the PL filter by assimilating DOA measurements.

3) **PLA.** Steps 1 to 7, excluding 4, are performed. This version extends the PL filter by assimilating detections of allies within the camera’s FoV.

4) **PLHA.** Steps 1 to 7 are performed. All cooperative measurements are assimilated.

When needed, the \( r \)-th robot extrapolates the beliefs of the other agents with a random walk model assuming maximum speeds of 1 m/s for \( x \) and \( y \) components and 3.0 rad/s for \( \psi \). For the sampling interval \( T_s = 20 \) ms, \( \Delta x_{\text{MAX}} = \Delta y_{\text{MAX}} = 0.02 \) m and \( \Delta \psi_{\text{MAX}} = 0.06 \) rad in (36).

At every two time instants, one robot broadcasts its current belief, generating a DOA measurement at the listeners. The communication schedule follows a Round-robin scheme, starting with robot \( r = 1 \) at time instant 2 \( T_s \), changing to robot \( r + 1 \) in the next communication slot and restarting from robot 1 after robot \( R \)'s turn.

Concerning implementation details, given that the sensors have very small covariance parameters, and it is possible that a relatively large number of measurements are available at each agent for jointly assimilation, we artificially increased those covariance values seen by the filter, a technique called covariance inflation. Higher covariance parameters correspond to probability densities with wider peaks and whose value do not fall rapidly to zero. This technique was necessary to avoid that a great number of particles have their weights collapsed to zero due to the very small variance of the sensors. The multiplicative factors were chosen empirically as 100 and 50 to the range and bearing variance of the camera, respectively, and 20 to the DOA variance.

The overconfidence that a robot may have about its own belief is another problem that may affect the assimilation of mutual measurements. It is characterized by the error being much larger than the magnitude of the elements in the diagonal of the covariance matrix of the self-belief. In order to minimize that effect, whenever a robot broadcasts its self-belief, we enlarged the covariance matrix by adding a term \( \delta = 0.01 \) to the elements in its diagonal.

Finally, in order to avoid the phenomenon of particle impoverishment of the resulting belief \( m_{r,n}^B(x_r,n) \), we performed resampling and regularization [21], [22] whenever the effective number of particles, defined as the inverse of the sum of the square of the weights, falls below 0.6\( N_p \).

### C. PERFORMANCE METRICS

Let

\[
P_{x,y} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}
\]

and \( P_\psi = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \) define two projection matrices such that \( P_{x,y} x_r,n \) and \( P_\psi x_r,n \) represent the 2D robot position and orientation, respectively.

The base metric for evaluating the filter performance is the root mean square error (RMSE) between the ground truth state \( x_r,n \) and its estimate \( \hat{x}_r,n \).

Detailing each metric, the first one is the estimated position root mean square error (RMSE) versus time \( e_{n,x}^r \), taking the average over all Monte Carlo runs and all robots, as

\[
e_n^{x,y} = \sqrt{\frac{1}{R} \sum_{r=1}^{R} \frac{1}{M} \sum_{m=1}^{M} \left| P_{x,y} x_r,n - P_{x,y} \hat{x}_r,n \right|^2},
\]

(59)

where \( ||x|| \) denotes the Euclidean norm of the real vector \( x \).

Similarly, we define the position RMSE versus time per robot \( e_{r,n}^{x,y} \) as

\[
e_{r,n}^{x,y} = \sqrt{\frac{1}{M} \sum_{m=1}^{M} \left| P_{x,y} x_r,n - P_{x,y} \hat{x}_r,n \right|^2}.
\]

(60)

The third metric of interest is the position RMSE per robot \( e_{r}^{x,y} \), obtained as the average over all Monte Carlo runs and time instants, as

\[
e_{r}^{x,y} = \frac{1}{N} \sum_{n=1}^{N} \frac{1}{M} \sum_{m=1}^{M} \left| P_{x,y} x_r,n - P_{x,y} \hat{x}_r,n \right|^2.
\]

(61)

Analogously, the RMSE metrics \( e_{\psi}^r \), \( e_{\psi}^{\psi} \), and \( e_{\psi}^y \) are computed for the estimated orientation by replacing \( P_{x,y} \) with the projection matrix \( P_\psi \) into equations (59) through (61), respectively.

In the sequel, let the set \( C \triangleq \{0, 3, 6, \ldots \} \) collect the discrete time indices at which robots are able to perform camera observations – which occur every 3 \( T_s \) starting at instant 0. Moreover, let \( \mathcal{L}_{r,n,m} \) denote the set of landmarks detected by robot \( r \) at instant \( n' \in C \) at the \( m \)-th Monte Carlo run. We define the average number of landmarks observed by robot \( r \) at instant \( n' \) as

\[
\bar{\tilde{\mathcal{L}}}_{r,n'} = \frac{1}{M} \sum_{m=1}^{M} |\mathcal{L}_{r,n',m}|
\]

(62)

where \( |\mathcal{L}_{r,n',m}| \) denotes the cardinality of the set \( \mathcal{L}_{r,n',m} \).

Additionally, for each robot \( r \) and Monte Carlo run \( m \), we define \( \beta_{r,n',m} \) as the blackout duration, i.e. the number of consecutive camera observation instants ending at time index \( n' \) such that the set of observations \( \mathcal{L}_{r,n,m} \) is empty for all \( n \) within the discrete interval \( \{n' - 3(\beta_{r,n',m} - 1), \cdots , n' \} \subset C \). In precise mathematical terms, we use the following recursion to compute the blackout duration

\[
\beta_{r,n',m} = \begin{cases} 0, & \mathcal{L}_{r,n',m} \neq \emptyset \\
1, & \mathcal{L}_{r,n',m} = \emptyset \text{ and } n' = 0 \\
1 + \beta_{r,n'-3,m}, & \mathcal{L}_{r,n',m} = \emptyset \text{ and } n' \neq 0. 
\end{cases}
\]

(63)

Finally, we define then the mean blackout duration metric \( \bar{\beta}_{r,n'} \) as the average of \( \beta_{r,n',m} \) over \( m \), i.e.

\[
\bar{\beta}_{r,n'} = \frac{1}{M} \sum_{m=1}^{M} \beta_{r,n',m}.
\]

(64)
D. SIMULATION RESULTS

Figs. 4 and 5 show respectively $e_{n}^{x,y}$ and $e_{n}^{\psi}$, i.e. the averaged position and orientation RMSE evolution over time – considering all robots and Monte Carlo runs – for the four variants of the proposed filter. As expected, the higher degree of cooperation improved the self-localization ability, whereas the non-cooperative PL variant presented larger errors. Among the cooperative filters, it was verified that the PLHA variant tends to present the lowest error, which is expected since it assimilates all measurements available at the network. Moreover, the cooperative PLHA filter shows a faster convergence time, i.e. it shortens the required time to eliminate the initial uncertainty from the prior distribution. During specific time intervals, the PLH and PLA filters, which assimilate only part of the mutual measurements, achieve similar or even slight lower error levels compared to that of the PLHA filter and present similar convergence times – whereas PLA is slightly slower, PLH has almost the same convergence rate. Finally, we recall that no stationary state is expected in this problem since the number of camera observations depends on the particular realization of the robots’ trajectories, and therefore can vary significantly over time.

Switching the perspective to each robot individually, Figs. 6 and 7 display respectively $e_{r}^{x,y}$ and $e_{r}^{\psi}$, i.e. the position and orientation RMSE – averaged over all time instants and Monte Carlo runs – for the four variants of the filter. It offers a different view of how cooperation impacts the error for each agent. In particular, note that agent 2, which has the worst RMSE for the non-cooperative filter, benefits the most from the cooperation. This is reflected in a smaller variability of the errors among the individual agents, each performing a different trajectory, when they estimate their poses cooperatively. Note that this variability is stronger for the position error. Finally, it is worth noting that, compared
We further investigate the RMSE peaks in Figs. 4 and 5 by detailing the RMSE curves of robots 2 and 5. In particular, Figs. 8 and 9 show – for robots 2 and 5, respectively – the position and orientation RMSE versus time of the PL and PLHA filters along with the average number of observed landmarks $\bar{\ell}_{r,n'}$ and the mean blackout duration $\bar{\beta}_{r,n'}$ for each time instant.

Robot 5 exhibits very similar error levels for both the position and orientation RMSE when comparing the non-cooperative PL filter and the fully cooperative PLHA filter. On the other hand, robot 2 presents much worse errors when comparing the PL and PLHA filters, especially presenting peaks at time instants 35 and 80 seconds. The average number of observed landmarks, which is the only measurement assimilated in the non-cooperative filter, does not explain alone the higher error by the robot 2, since the error peaks occur at time instants where the same number of landmarks are seen by the robot. Additionally, when considering all time instants, both robots 2 and 5 observe a similar average number of landmarks – refer to Figs. 8-c and 9-c.

A characteristic of the signal model of this problem is that a single landmark observation is sufficient to maintain a non-ambiguous estimate of a robot’s pose once it has been determined. The continuous absence of measurements, or a
blackout, makes the robot blind about its environment and, if that blackout is long enough, the robot is prone to lose its track due to the dead reckoning. Figs. 8-d and 9-d indicate the mean blackout duration \( \beta \) experienced respectively by robots \( r = 2 \) and \( r = 5 \) at each time instant \( n' \) considering all Monte Carlo runs. In the sequel, we investigate the correlation between the mean blackout evolution and the peaks along the Monte Carlo runs. In the sequel, we investigate the correlation between the mean blackout evolution and the peaks along the Monte Carlo runs.

First, we define the set \( C_p \triangleq \{ n' \in C \mid n_c \leq n' \leq N \} \), with cardinality \( |C_p| = \lfloor (N - n_c)/3 \rfloor + 1 \), containing all camera observation instants starting at \( n_c = 99 \) to ignore the initial convergence time. Then we define the vectors \( e^{x,y}_{t:R,n',n'' \in C_p} \), \( e^{\psi}_{t:R,n',n'' \in C_p} \), \( \beta_{t:R,n',n'' \in C_p} \), and \( \ell_{t:R,n',n'' \in C_p} \), which collect the position and orientation RMSE, as well as the mean blackout duration and average number of landmarks, from all robots and all camera observation instants in \( C_p \). We suppress their subscripts from now on. We proceed to calculate their Pearson correlation coefficient, using the formula

\[
\rho_{ab} = \frac{\sum_{i=1}^{N} (a_i - \mu_a)(b_i - \mu_b)}{\sqrt{\sum_{i=1}^{N} (a_i - \mu_a)^2} \sqrt{\sum_{i=1}^{N} (b_i - \mu_b)^2}},
\]

where \( N, \mu_a \) and \( \mu_b \) are, respectively, the number of samples in the vectors \( a \) and \( b \) and their sample means. In our case, \( a = e^{x,y} \) or \( e^{\psi} \), \( b = \beta \) or \( \ell \), and \( N = R \times |C_p| \).

The results are shown in Table 1. The correlation values between 0.674 and 0.900 indicate that, indeed, there exists a correlation between those quantities. Specifically, the strong positive correlation between the mean blackout and the position / orientation RMSE curves suggests that the absence of camera observations indicated by the former significantly contributes to the error increase of the latter. On the other hand, the weak negative correlation between the RMSE curves and the average number of observed landmarks indicates that our hypothesis is true, i.e. the peaks along the RMSE curves are not related to fluctuations on the average number of assimilated landmarks.

Finally, we assessed the computational cost of the algorithm by measuring the average CPU time. The second column in Table 2 shows the CPU time for each of the four variants of the algorithm, namely PL, PLH, PLA, and PLHA, averaged over all robots, Monte Carlo runs and time instants. The values are presented in normalized units such that the average CPU time for the non-cooperative PL variant, where players only assimilate landmark measurements, is normalized to one. The normalization is performed since the absolute values obtained using the Matlab\textregistered implementation do not properly represent the values that would be obtained by an efficient implementation (using C++, for example) designed to assimilate measurements online under a real world scenario. Moreover, the discussion in Klaas et al. [37] suggests that an implementation of marginal particle filter-type algorithms similar to ours, with \( N_p = N_q \), might be feasible with complexity \( O(N_p \log N_p) \) rather than \( O(N_p^2) \).

On the other hand, previous results from our robotics team reported in [27] show that a non-cooperative Monte Carlo localization implementation running with \( N_p = 200 \) particles in a similar scenario would have a CPU time of 0.025 ms if run on an Intel\textregistered Core\textsuperscript{TM} i7-4720HQ CPU @ 2.60 GHz, a typical setup for the robot soccer simulation. The third column in Table 2 shows the projected CPU time for the proposed algorithms using that value as a time estimate for an alternative, efficient implementation. The fourth column in turn shows the duty cycle, i.e. the average CPU usage per sampling time, of the localization algorithms considering the sampling time \( T_s = 20 \) ms.

As expected, the average CPU time increases as more available data in the network is assimilated by the individual agents. In particular, as shown in the second column of Table 2, the PLHA variant, where players assimilate directions of arrival of broadcast messages in addition to range and bearing measurements for both landmarks and other allied teammates, required an average CPU time in our simulations roughly 9 times greater than the average CPU time for the PL variant, indicating a trade-off between computational cost and localization performance. The simulations were conducted using Matlab\textregistered R2015a, on a 64-bit machine with an Intel\textregistered Core\textsuperscript{TM} i9-7900X CPU @ 3.30 GHz and 64 GB RAM. Finally, the estimated duty cycles indicate that efficient implementations of the proposed algorithms are probably suitable for online processing.

### VI. CONCLUSION

In this paper, we approached the self-localization problem for multiple robotic agents in a cooperative and distributed way assuming a scenario where communication is limited by external restrictions such that robots cannot exchange information whenever they need to perform cooperation. The factor graph formulation allows a decentralized modeling, which is the only feasible approach for this problem.

### TABLE 1. Pearson correlation coefficients between the position and orientation RMSE, \( e^{x,y} \) and \( e^{\psi} \), and the mean blackout duration, \( \beta \), or the average number of landmarks, \( \ell \). The four vectors collect robots 1 to \( R \) and time instants \( n' \in C_p \).

| Variant | Mean blackout | Average landmarks |
|---------|--------------|-------------------|
|         | Position     | Orientation       |                  |
| PL      | 0.750        | 0.900             | -0.150           | -0.464               |
| PLH     | 0.674        | 0.778             | -0.136           | -0.499               |
| PLA     | 0.678        | 0.859             | -0.125           | -0.422               |
| PLHA    | 0.700        | 0.774             | -0.215           | -0.497               |

### TABLE 2. CPU time, averaged per robot, Monte Carlo run and all time instants, for the four variants of the algorithm. Normalized time, projected time using previous data from real-world simulation of a similar algorithm, and duty cycle considering \( T_s = 20 \) ms.

| Variant | Average time, normalized (ms) | Average time, projected (ms) | Duty cycle |
|---------|-------------------------------|-------------------------------|------------|
| PL      | 1                             | 0.025                         | 0.12%      |
| PLH     | 4.7                           | 0.117                         | 0.58%      |
| PLA     | 5.5                           | 0.137                         | 0.68%      |
| PLHA    | 9.2                           | 0.230                         | 1.15%      |
The implementation with sequential Monte Carlo allows the numerical evaluation of intractable expressions that do not have a closed-form analytical solution.

Concerning the final algorithm’s RMSE, the cooperative approach provides marginal improvements for agents that present high errors due to a prolonged lack of landmark detections. The main drawback of the proposed implementation of cooperative filters is its high computational cost, which, although feasible in our scenario, can be a restriction in others.

In particular, the computational cost of the implementation, especially when assimilating mutual measurements using the marginal particle filter, is the main limitation of the proposed method. Although our results indicate that our hardware can run the algorithm, that could not be the case in other scenarios where there are either tight restrictions on processing time or hardware specifications, or the number of particles must be increased for higher resolution of the posterior p.d.f.

Future work can be done on proposing distributed approaches that reduce the need for sampling particles of a teammate’s distribution. Moreover, the proposed cooperative formulation allows the incorporation of joint localization and tracking of objects, such as the ball or opponents. Other topics are the evaluation of performance bounds for the proposed algorithm, considering the communication restrictions and the topology exhibited by the network being not regular over time. Finally, alternative methods from the literature can be the evaluation of performance bounds for the proposed approach provides marginal improvements for agents that have a closed-form analytical solution.

As a supplementary material, we provide the source code of the simulations and a video with the scenario generation and an animated representation of the results.

REFERENCES

[1] J. Boyd, “A discourse on winning and losing,” in Curtis E. LeMay Center for Doctrine Development and Education. Montgomery, AL, USA: Air Univ. Press, 2018.

[2] C. Haskins, “Systems engineering handbook, version 3. A guide for system life cycle processes and activities,” in Proc. Int. Council Syst. Eng. (INCOSE), Seattle, WA, USA, Jun. 2006.

[3] N. Patwari, J. N. Ash, S. Kyperountas, A. O. Hero, R. L. Moses, and N. S. Correal, “Locating the nodes: Cooperative localization in wireless sensor networks,” IEEE Signal Process. Mag., vol. 22, no. 4, pp. 54–69, Jul. 2005.

[4] H. Wymeersch, U. Ferner, and M. Win, “Cooperative Bayesian self-tracking for wireless networks,” IEEE Commun. Lett., vol. 12, no. 7, pp. 505–507, Jul. 2008.

[5] H. Wymeersch, J. Lien, and M. Z. Win, “Cooperative localization in wireless networks,” Proc. IEEE, vol. 97, no. 2, pp. 427–450, Feb. 2009.

[6] M. Z. Win, F. Meyer, Z. Liu, W. Dai, S. Bartoletti, and A. Conti, “Efficient multisensor localization for the Internet of Things: Exploring a new class of scalable localization algorithms,” IEEE Signal Process. Mag., vol. 35, no. 5, pp. 153–167, Sep. 2018.

[7] F. Meyer, Z. Liu, and M. Z. Win, “Network localization and navigation using measurements with uncertain origin,” in Proc. 21st Int. Conf. Inf. Fusion (FUSION) IEEE, Jul. 2018, pp. 1–7.

[8] I. Shames, B. Fidan, B. D. O. Anderson, and H. Hnam, “Cooperative self-localization of mobile agents,” IEEE Trans. Aerosp. Electron. Syst., vol. 47, no. 3, pp. 1926–1947, Jul. 2011.

[9] D. Fox, W. Burgard, H. Kruppa, and S. Thrun, “A probabilistic approach to collaborative multi-robot localization,” Auto. Robots, vol. 8, no. 3, pp. 325–344, Jun. 2000.

[10] S. S. Kia, S. Rounds, and S. Martinez, “Cooperative localization for mobile agents: A recursive decentralized algorithm based on Kalman-filter decoupling,” IEEE Control Syst. Mag., vol. 36, no. 2, pp. 86–101, Apr. 2016.

[11] J. Zhu and S. S. Kia, “Cooperative localization under limited connectiv- ity,” IEEE Trans. Robot., vol. 35, no. 6, pp. 1523–1530, Dec. 2019.

[12] W. Yuan, N. Wu, B. Etzlinger, Y. Li, C. Yan, and L. Hanzo, “Expectation–maximization-based passive localization relying on asynchronous receivers: Centralized versus distributed implementations,” IEEE Trans. Commun., vol. 67, no. 1, pp. 668–681, Aug. 2018.

[13] M. A. Caceres, F. Penna, H. Wymeersch, and R. Garello, “Hybrid cooperative positioning based on distributed belief propagation,” IEEE J. Sel. Areas Commun., vol. 29, no. 10, pp. 1948–1958, Nov. 2011.

[14] H. Oliveira, S. S. Dias, and M. G. da Silva Bruno, “Cooperative terrain navigation using hybrid GMM/SMC message passing on factor graphs,” IEEE Trans. Aerosp. Electron. Syst., vol. 56, no. 5, pp. 3958–3970, Oct. 2020.

[15] W. Yuan, N. Wu, Q. Guo, X. Huang, Y. Li, and L. Hanzo, “TOA-based passive localization constructed over factor graphs: A unified framework,” IEEE Trans. Commun., vol. 67, no. 10, pp. 6952–6965, Oct. 2019.

[16] RoboCup 3D. Simspark Documentation. Accessed: Oct. 12, 2020. [Online]. Available: http://simspark.sourceforge.net/wiki/index.php/Main_Page

[17] G. Mao, B. Fidan, and B. D. S. Anderson, “Wireless sensor network localization techniques,” Comput. Netw., vol. 51, no. 10, pp. 2529–2553, Jul. 2007.

[18] E. A. Wan and R. Van Der Merwe, “The unscented Kalman filter for non-linear estimation,” in Proc. IEEE Adapt. Syst. Signal Process., Commun., Cont. Symp., Oct. 2000, pp. 153–158.

[19] G. Welch and G. Bishop, “An introduction to the Kalman filter,” Dept. Comput. Sci., Univ. North Carolina, Tech. Rep. TR 95-041, 1995.

[20] B. D. O. Anderson and J. B. Moote, Optimal Filtering, New York, NY: Dover, 2005.

[21] A. Doucet, S. Godsill, and C. Andrieu, “On sequential Monte Carlo sampling methods for Bayesian filtering,” Statist. Comput., vol. 10, no. 3, pp. 197–208, Jul. 2000.

[22] M. G. S. Bruno, “Sequential Monte Carlo methods for nonlinear discrete-time filtering,” in Proc. 21st Int. Conf. Inf. Fusion (FUSION) IEEE, Jul. 2018, pp. 1–7.

[23] X. Li, H. Lu, D. Xiong, H. Zhang, and Z. Zheng, “A survey on visual perception for RoboCup MSL soccer robots,” Int. J. Adv. Robotic Syst., vol. 10, no. 2, p. 110, Feb. 2013.

[24] S. Y. Chen, “Kalman filter for robot vision: A survey,” IEEE Trans. Ind. Electron., vol. 59, no. 11, pp. 4409–4420, Nov. 2012.

[25] S. Thrun, D. Fox, W. Burgard, and F. Del laert, “Robust Monte Carlo localization for mobile robots,” Artif. Intell., vol. 128, nos. 1–2, pp. 99–141, May 2001.

[26] F. Dellaert, D. Fox, W. Burgard, and S. Thrun, “Monte Carlo localization for mobile robots,” in Proc. IEEE Int. Conf. Robot. Autom., May 1999, pp. 1322–1328.

[27] A. Muzio, L. Aguier, M. R. O. A. Máximo, and S. C. Pinto, “Monte Carlo localization with field lines observations for simulated humanoid robotic soccer,” in Proc. 13th Latin American Robotics Symp. IV Brazilian Robot. Symp. (LARS/SBR), Oct. 2016, pp. 334–339.

[28] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, “Factor graphs and the sum-product algorithm,” IEEE Trans. Inf. Theory, vol. 47, no. 2, pp. 498–519, Feb. 2001.

[29] B. J. Frey, F. R. Kschischang, H.-A. Loeliger, and N. Wiberg, “Factor graphs and algorithms,” in Proc. Annu. Allerton Conf. Commun. Contr. Comput., Sep. 1997, pp. 666–680.

[30] H. Loeliger, “An introduction to factor graphs,” IEEE Signal Process. Mag., vol. 21, no. 1, pp. 28–41, Jan. 2004.

[31] G. C. G. Fernandes and M. G. S. Bruno, “Complexity-reduced suboptimal equalization with Monte Carlo based MIMO detector,” in Proc. 27th Eur. Signal Process. Conf. (EUSIPCO), Sep. 2019, pp. 1–5.

[32] A. Ahmad, G. D. Tipaldi, P. Lima, and W. Burgard, “Cooperative robot localization and target tracking based on least squares minimization,” in Proc. IEEE Int. Conf. Robot. Autom., May 2013, pp. 5696–5701.
P. Jensfelt and H. I. Christensen, “Pose tracking using laser scanning and...
M. Klaas, N. Freitas, and A. Doucet, “Toward practical N²...
P. R. A. Silva and M. G. S. Bruno, “Mobile robot localization using...
P. Pinheiro and P. Lima, “Bayesian sensor fusion for cooperative object localization and world modeling,” in Proc. 8th Conf. Intell. Auton. Syst., May 2004.
A. Ahmad, G. Lawless, and P. Lima, “An online scalable approach to unified multirobot cooperative localization and object tracking,” IEEE Trans. Robot., vol. 33, no. 5, pp. 1184–1199, Oct. 2017.
P. Jensfelt and H. I. Christensen, “Pose tracking using laser scanning and minimalistic environmental models,” IEEE Trans. Robot. Autom., vol. 17, no. 2, pp. 138–147, Apr. 2001.
P. R. A. Silva and M. G. S. Bruno, “Mobile robot localization using improved SIR filters and parametric models of the environment,” in Proc. Int. Conf. Acoust., Speech Signal Process. (ICASSP), May 2006, p. 3.
M. Klaas, N. Freitas, and A. Doucet, “Toward practical N² Monte Carlo: The marginal particle filter,” in Proc. 21st Conf. Uncertainty Artif. Intell., 2005, pp. 308–315.

GUILHERME C. G. FERNANDES (Member, IEEE) received the bachelor’s degree (Hons.) (summa cum laude) in electronic engineering and the master’s degree in electronic and computer engineering from the Instituto Tecnológico de Aeronáutica (ITA), São José dos Campos, Brazil, in 2017 and 2019, respectively. He is currently with the Embraer S.A. Research and Development Team, São José dos Campos, developing multi-sensor fusion algorithms for autonomous systems. His main interests are the applications of signal processing and statistical signal processing in robotics, communications, surveillance, and defense systems. His current research focuses on modeling and assessing distributed algorithms for Bayesian estimation and multi-sensor fusion.

STIVEN S. DIAS (Member, IEEE) received the bachelor’s degree in computer engineering from the Universidade Federal do Espírito Santo, Vitória, Brazil, in 2006, the master’s degree in aeronautical engineering from the Instituto Tecnológico de Aeronáutica (ITA), São José dos Campos, Brazil, in 2008, and the Ph.D. degree in electrical and computer engineering from ITA, in 2014. His main research interests include distributed estimation in sensor networks, multi-target, multi-sensor tracking, and non-linear filtering with passive sensors. He has been with Embraer S.A. as a Research Engineer since 2008, developing advanced data fusion systems. His main professional interests include distributed data fusion over limited bandwidth datalinks and radar-based coastal surveillance systems. Since 2018, he has also been involved in several initiatives related to autonomous vehicles.

MARCOS R. O. A. MAXIMO (Member, IEEE) received the B.Sc. degree (Hons.) (summa cum laude) in computer engineering, and the M.Sc. and Ph.D. degrees in electronic and computer engineering from the Aeronautics Institute of Technology (ITA), Brazil, in 2012, 2015, and 2017, respectively. He is currently a Professor at ITA, where he is a member of the Autonomous Computational Systems Laboratory (LAB-SCA) and leads the Robotics Competition Team: ITAndroids. He is especially interested in humanoid robotics. Moreover, his research interests include mobile robotics, dynamical systems control, and artificial intelligence.

MARCELO G. S. BRUNO (Senior Member, IEEE) received the bachelor’s and master’s degrees in electrical engineering from the University of São Paulo, Brazil, in 1992 and 1993, respectively, and the Ph.D. degree in electrical and computer engineering from Carnegie Mellon University, Pittsburgh, PA, USA, in 1998. He is currently affiliated with the Electrical Engineering Division, Instituto Tecnológico de Aeronáutica (ITA), São José dos Campos, Brazil, where he is currently an Associate Professor. His research interests are in statistical signal processing, particularly Markov random fields (Mrfs), hidden Markov models (HMMs), particle filters/sequential Monte Carlo methods (SMC), Markov Chain Monte Carlo (MCMC), statistics on manifolds, random finite sets (RFS), Bernoulli and multi-Bernoulli filters, non-parametric belief propagation, and their applications in target detection/tracking, image processing, machine learning, mobile robotics, and telecommunication. His current research is focused on distributed detection, estimation, and filtering in sensor networks combining Bayesian methods with iterative consensus, diffusion, and random information dissemination strategies. His particular areas of application under investigation include cooperative equalization of frequency-selective, manifold-constrained communication channels using multiple receivers, subspace tracking in manifolds, and joint localization and tracking in distributed multiagent, multisensor networks. He served as an Associate Editor of the IEEE Signal Processing Letters, IEEE Transactions on Signal Processing, and the IEEE Transactions on Signal and Information Processing Over Networks.

* * *

213184
VOLUME 8, 2020