Distributed Consensus on Robot Networks for Dynamically Merging Feature-Based Maps

Rosario Aragues, *Student Member, IEEE,* Jorge Cortes, *Senior Member, IEEE,*
Carlos Sagues, *Senior Member, IEEE*

***Abstract*—We study the feature-based map merging problem in robot networks. Along its operation, each robot observes the environment and builds and maintains a local map. Simultaneously, each robot communicates and computes the global map of the environment. The communication between the robots is range-limited. We propose a dynamic strategy based on consensus algorithms that is fully distributed and does not rely on any particular communication topology. Under mild connectivity conditions on the communication graph, our merging algorithm asymptotically converges to the global map. We present a formal analysis of its convergence rate and provide accurate characterizations of the errors as a function of the timestep. The proposed approach has been experimentally validated using real visual information.**

I. INTRODUCTION

THERE is an increasing interest in multi-robot systems. The availability of a local map allows each robot to make local decisions such as local navigation or collision avoidance. However, it is also of interest for each robot to have a representation of the environment beyond its local map. The fusion of the local observations of all the team members leads to a merged map that contains more precise information and more features. In a static map merging scenario, the information fusion is carried out after the exploration. Dynamic solutions, where the information is fused while the robots operate, are more interesting. They enable other multi-robot tasks such as cooperative exploration, or task assignment. In this paper, we study the problem of dynamic map merging, where each robot's communication radius is limited, and hence the communication topology is not a complete graph.

While multi-robot localization under communication constraints has received some attention [1], [2], most of the existing multi-robot map merging solutions are extensions of the single robot case under centralized schemes, all-to-all communication among the robots, or broadcasting methods. Particle filters [3] have been generalized to multi-robot systems assuming that the robots broadcast their controls and their observations. In multi-robot submap filters [4] and graph maps of laser scans [5] approaches, each robot builds a local submap and sends it by broadcast to all the other agents or to a central node. The same solution could be applied for many existing submapping methods [6]. However, in robot network scenarios, distributed approaches are often necessary because of limited communication, switching topologies, link failures, and limited bandwidth.

Distributed estimation methods [7]–[14] maintain a joint estimate of a system that evolves with time by combining noisy observations taken by a sensor network. Early approaches sum the measurements from the different agents in IF (Information Filter) form. If the network is complete [7], then the resulting estimator is equivalent to the centralized one. In general networks the problems of cyclic updates or double counting information appear when nodes sum the same piece of data more than once. The use of the channel filter [8], [9] avoids these problems in networks with a tree structure. The Covariance Intersection method [10] produces consistent but highly conservative estimates in general networks. More recent approaches [11]–[14] use distributed consensus filters to average the measurements taken by the nodes. The interest of distributed averaging is that the problems of double counting the information and cyclic updates are avoided. They, however, suffer from the delayed data problem that takes place when the nodes execute the state prediction without having incorporated all the measurements taken at the current step [15]. For general communication schemes [11], the delayed data problem leads to an approximate KF (Kalman Filter) estimator. An interesting solution is given in [12] but its convergence is proved in the absence of observation and system noises. In the algorithm proposed in [14], authors prove that the nodes' estimates are consistent, although these estimates have disagreement. Other algorithms have been proposed that require the previous offline computation of the gains and weights of the algorithm [13]. The main limitation of all the previous works is that they consider linear systems without inputs, and where the evolution of the system is known by all the robots. Here instead we are interested in more general scenarios, without the previous restrictions. We allow each robot to build its map by using system models not necessarily linear or known by the other robots, or where the robot odometry is modeled as an input, among others. A recent work that does not suffer from the previous limitations is given in [16]. Here each robot records its own measurements and odometry, as well as the observations and odometry from any other robot it encounters. Despite being very interesting and going beyond the state-of-art, this work has the drawback that robots must maintain an unbounded amount of memory, which depends on the time between meetings. Moreover, if a single robot fails or leaves

Manuscript created November 24, 2010. This work was supported by project Ministerio de Ciencia e Innovacion DPI2009-08126, grant MEC BES-2007-14772 and NSF award CCF-0917166.

R. Aragues and C. Sagues are with Instituto de Investigación en Ingeniería de Aragón, Universidad de Zaragoza, 50018 Zaragoza Spain (R. Aragues is the corresponding author; e-mail: raragues@unizar.es; C. Sagues, e-mail: csagues@unizar.es).

J. Cortes is with the Department of Mechanical and Aerospace Engineering, University of California San Diego, CA 92093-0411 USA (e-mail: cortes@ucsd.edu).

the network, the whole system fails, and the data association is not discussed. In our case, the information fusion is carried out on the local maps of the robots for which efficient distributed data association methods [17] already exist in the literature.

We let each robot build a local map of the environment using its own measurements. At specific time instants, robots fuse their local maps and build a global map. Robots do not introduce information from the global map into their local maps. Thus, local maps between different robots remain independent during the whole exploration. Due to the independence between local maps, our approach does not suffer from the previously mentioned problems of delayed data, which prevents robots from converging to exactly the same global map, or cyclic updates and double counting information, which lead to overconfident maps. Note that our global map is different in general than the one computed by a centralized SLAM filter, since robots do not use information from the other team members to update the local maps. An advantage of our approach is the natural robustness that results from its distributed implementation. Additionally, our method ensures that the global map structure remains sparse, and thus provides a natural submapping. Our method averages local maps, expressed in IF form, instead of their measurements, using the consensus filter. The merging of maps composed of static features does not consider inter-robot observations (although this information could also be included, see Remark II.1 later). We build on ideas from consensus algorithms that allow the introduction of new information, while taking advantage of the latest global map [18], [19]. We use a discrete-time version of the PI algorithm which is more appropriate for the robot systems we consider. As weight matrices we use the Metropolis weights [20] which have been shown to perform quite good in multi-agent systems [14], [21], [22] and that have the benefit that they can be locally computed by the agents.

The contributions of this manuscript, and novelty with respect to our previous works on distributed map merging [23], [24] are the following: (i) the proposal of the dynamic consensus strategy where, at each step, a discrete-time version of the PI algorithm is executed; (ii) the careful study of the convergence rate of the dynamic consensus strategy; (iii) the applications of this study to characterize the errors in the map merging and understand the trade-offs between the number of iterations and the performance of the algorithm; (iv) the theoretical and experimental study of its time and communication complexity; and (v) the implementation for feature-based maps taking into account the possibly different features discovered by each robot during the exploration. In [23], our robots performed the exploration of the environment, and only at the end of it, ran a static consensus algorithm to merge their maps. In this paper, instead, robots dynamically merge the information online, i.e., at the same time that they are performing the exploration. This on-the-fly fusion is technically challenging and computationally demanding. With respect to the conference version of this paper [24], here we make a formal and experimental in-depth study of the properties of the algorithm (particularly items (ii)-(v)).

This paper is organized as follows. Section II formally states the problem. Section III solves a simplified problem with scalar and constant inputs. Section IV presents the dynamic consensus strategy where the robots track the average of the scalar inputs. Section V solves the dynamic map merging problem by using multidimensional inputs containing the information matrices and vectors of the local maps; we briefly discuss the initial correspondence, map alignment and data association, and the algorithm complexity. Finally Section VI evaluates the performance of the algorithm under real visual data, and against a centralized solution. Additional information on consensus algorithms is given in the Appendix.

II. PRELIMINARIES

Throughout the paper we let n be the number of robots. Indices i, j refer to robots, r, s to elements within the maps, and G to the global map. We use $k \in \mathbb{N}$ for exploration steps and $t \in \mathbb{N}$ for iteration numbers. We let $\mathbf{I}$ be the $n \times n$ identity matrix, and $\mathbf{0}$ be a $n \times n$ matrix with all its elements equal to zero. When they are followed by a subindex $n_1 \times n_2$, this specifies their dimensions. We let $\mathbf{1} \in \mathbb{R}^n$ be a column vector with all entries equal to 1. Given a matrix W, $[W]_{ij}$ denotes its (i, j) entry, $\lambda_i(W)$, $\mathbf{v}_i(W)$ are its $i-$th eigenvalue and eigenvector, and $\lambda_{\text{eff}}(W)$ is the modulus of its eigenvalue with the second largest absolute value.

We consider a team of $n \in \mathbb{N}$ robots exploring an unknown environment. At the exploration step k, each robot i has observed $m_i^k \in \mathbb{N}$ features and it has estimated its own pose together with the positions of the features. Let the constants $\mathtt{szr}$ and $\mathtt{szf}$ represent the size of respectively a robot pose and a feature position[1]. The estimates at each robot $i \in \{1, \dots, n\}$ and each step k are stored into a stochastic map with mean $\hat{\mathbf{x}}_i^k \in \mathbb{R}^{\mathcal{M}_i^k}$ and covariance matrix $\Sigma_i^k \in \mathbb{R}^{\mathcal{M}_i^k \times \mathcal{M}_i^k}$, being $\mathcal{M}_i^k = \mathtt{szr} + m_i^k\ \mathtt{szf}$. Let $\mathbf{x}_i^k \in \mathbb{R}^{\mathcal{M}_i^k}$ contain the true robot pose and the true positions of the m_i^k features, then

$$\hat{\mathbf{x}}_i^k = \mathbf{x}_i^k + \mathbf{v}_i^k, \tag{1}$$

where $\mathbf{v}_i^k$ is a zero mean noise with covariance matrix Σ_i^k. In this paper, we do not discuss the exploration strategies or the Simultaneous Localization and Map Building (SLAM) algorithms for obtaining the local maps.

If at step k the information from the n robots was available, e.g., at a central agent, then the global map combining the information of the local maps at the n robots at step k could be computed. Let $m \in \mathbb{N}$ be the number of different features in the environment and $\mathbf{x} \in \mathbb{R}^{\mathcal{M}_G}$ be the vector with the true poses of the n robots and the true positions of the m features, being $\mathcal{M}_G = n\ \mathtt{szr} + m\ \mathtt{szf}$. Each robot $i \in \{1, \dots, n\}$ at step k has observed $m_i^k \leq m$ features and we let $H_i^k \in \{0, 1\}^{\mathcal{M}_i^k \times \mathcal{M}_G}$ be the observation matrix that relates the elements in $\mathbf{x}$ and $\mathbf{x}_i^k$ so that $\mathbf{x}_i^k = H_i^k \mathbf{x}$. The local map of each robot i (1) is a partial observation of $\mathbf{x}$,

$$\hat{\mathbf{x}}_i^k = H_i^k \mathbf{x} + \mathbf{v}_i^k, \tag{2}$$

where we assume that the noises $\mathbf{v}_i^k, \mathbf{v}_j^{k'}$ are independent for different robots $i \neq j$ and all $k, k' \in \mathbb{N}$, since every robot has

[1]e.g., $\mathtt{szr} = 3$ when the robot pose is composed of its planar position (x, y) and orientation θ; $\mathtt{szf} = 2$ or $\mathtt{szf} = 3$ for respectively 2D or 3D environments.

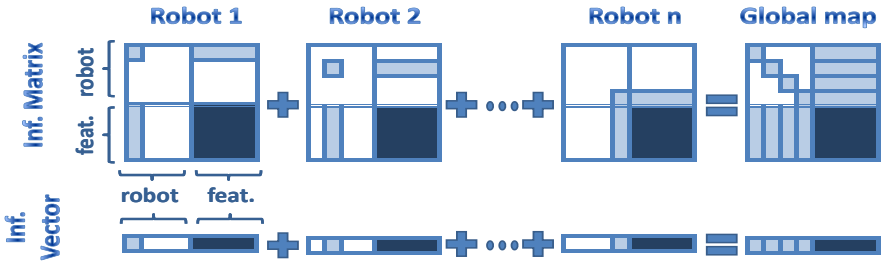


Fig. 1. Centralized merging of the local maps of n robots in IF form. The global information matrix I_G^k and vector $\mathbf{i}_G^k$ are the addition of the local ones I_i^k, $\mathbf{i}_i^k$, for $i \in \{1, \dots, n\}$. The first rows and columns contain the robot poses, and the last ones, the feature positions. The elements with zero value are displayed in white. Each robot i has information of its own pose (light blue) and of the feature positions (dark blue), but it has no information of any other robot poses $j \neq i$ (white).

constructed the map based on its own observations. Note that since the local map of a robot i at step k is an evolution of its map at any previous step $k' < k$, then the noises $\mathbf{v}_i^k, \mathbf{v}_i^{k'}$ are not independent. Let $I_i^k \in \mathbb{R}^{\mathcal{M}_G \times \mathcal{M}_G}$ and $\mathbf{i}_i^k \in \mathbb{R}^{\mathcal{M}_G}$ be the information matrix and vector of the local map at robot i and step k in IF form,

$$I_i^k = (H_i^k)^T (\Sigma_i^k)^{-1} H_i^k, \qquad \mathbf{i}_i^k = (H_i^k)^T (\Sigma_i^k)^{-1} \hat{\mathbf{x}}_i^k, \tag{3}$$

for $i \in \{1, \dots, n\}$. The information matrix I_G^k and vector $\mathbf{i}_G^k$ of the global map at step k in IF form are

$$I_G^k = \sum_{i=1}^{n} I_i^k, \qquad \mathbf{i}_G^k = \sum_{i=1}^{n} \mathbf{i}_i^k. \tag{4}$$

The previous operation is additive, commutative, and associative. For this reason, merging the maps in IF form is a common practice [25]. Equivalently, the global map at step k can be expressed by its mean and covariance matrix,

$$\hat{\mathbf{x}}_G^k = (I_G^k)^{-1}\, \mathbf{i}_G^k, \qquad \Sigma_G^k = (I_G^k)^{-1}. \tag{5}$$

Note that the global map in eqs. (3)-(5) is different from the one that would be obtained by a centralized multi-robot SLAM, since the local maps in eq. (3) do not include measurements from the other robots. Our local maps remain independent and can be fused by the addition of the information matrices and vectors as in eq. (4), and the information matrix of our global map remains sparse (Fig. 1).

Remark II.1. The framework described so far does not take into account inter-robot observations. We can include this information in our framework by consider the pose of a robot j at a certain step as a static feature in the environment, observed simultaneously by robot j itself, and by the robot that took the inter-robot measurement. This option, however, increases the map size and requires additional coordination mechanisms to notify a robot that someone is taking an observation of it. •

Maps in information form have the property that entries (r, s) and r in the information matrix I_i^k and vector $\mathbf{i}_i^k$ associated to the elements not observed by robot i are zero (Fig. 1, white elements). Consider a feature observed by several robots $\mathcal{R} \subseteq \{1, \dots, n\}$ (Fig. 1, dark blue area). The associated entries (r, s) and r in the global map $[I_G^k]_{rs}$, $[\mathbf{i}_G^k]_r$ (4) are the addition of the different values $[I_i^k]_{rs}$, $[\mathbf{i}_i^k]_r$, for $i \in \mathcal{R}$,

$$[I_G^k]_{rs} = \sum_{i \in \mathcal{R}} [I_i^k]_{rs}, \qquad [\mathbf{i}_G^k]_r = \sum_{i \in \mathcal{R}} [\mathbf{i}_i^k]_r.$$

Here, each robot i reaches a consensus between its own and the others' values $[I_j^k]_{rs}$, $[\mathbf{i}_j^k]_r$, for $j \in \mathcal{R}$. Consider now the estimated pose of a robot i. It was exclusively observed by i, and thus for any other robot $j \neq i$ the associated entries (r, s) and r are zero, $[I_j^k]_{rs} = 0$, $[\mathbf{i}_j^k]_r = 0$. Only robot i is providing information of these entries for the global map,

$$[I_G^k]_{rs} = [I_i^k]_{rs}, \qquad [\mathbf{i}_G^k]_r = [\mathbf{i}_i^k]_r,$$

and thus it is clear that here there is no need for consensus. The dynamic map merging problem can be separated into two parts. The first part consists of propagating the rows and columns of I_i^k, $\mathbf{i}_i^k$ associated with the pose of a robot j. Any other robot $i \neq j$ just incorporates this data into its global map. The second part, which consists of reaching a consensus on the entries associated exclusively to features, is discussed along the following sections.

Problem 1. *We consider $n \in \mathbb{N}$ robots exploring and acquiring local maps at some exploration steps $k = 1, 2, \dots$ as in eqs.* (1)-(3)*. The communication is range-limited and two robots can exchange data only if they are close enough. We let $\mathcal{G}_k = (\mathcal{V}, \mathcal{E}_k)$ be the undirected communication graph at step k. The nodes are the robots, $\mathcal{V} = \{1, \dots, n\}$. If robots i, j can communicate then there is an edge between them, $(i, j) \in \mathcal{E}_k$. The set of neighbors $\mathcal{N}_i^k$ of robot i at step k is*

$$\mathcal{N}_i^k = \{j \mid (i, j) \in \mathcal{E}_k, j \neq i\}.$$

The goal is the design of distributed algorithms so that each robot $i \in \mathcal{V}$ computes and tracks the global map in eqs. (4)-(5) *based on local interactions with its neighbors $\mathcal{N}_i^k$.*

III. CONSENSUS ON CONSTANT SCALAR INPUTS

We start by considering a simplified version of Problem 1, where there is a single exploration step k. Instead of an information matrix and a vector, each robot $i \in \mathcal{V}$ has a single scalar input $u_i \in \mathbb{R}$. The global data $x_G \in \mathbb{R}$ is the sum of the inputs u_i and we let $x_{avg} \in \mathbb{R}$ be their average,

$$x_G = \sum_{i=1}^{n} u_i, \qquad x_{avg} = \frac{1}{n} \sum_{i=1}^{n} u_i = \frac{1}{n} x_G. \tag{6}$$

The goal is that each robot $i \in \mathcal{V}$ computes an estimate $x_i(t) \in \mathbb{R}$ of x_{avg} by local interactions with its neighbors $\mathcal{N}_i$.

The previous simplified problem can be solved by distributed consensus algorithms [26] for systems with constant inputs. In particular, we analyze in depth a discrete version of the Proportional Integral (PI) estimator [18] in the context of dynamic consensus. As we will show, the capabilities of the PI for re-using past information are crucial for the considered map merging scenario. The PI algorithm is:

$$\begin{bmatrix} \dot{\mathbf{x}}(t) \\ \dot{\mathbf{w}}(t) \end{bmatrix} = \begin{bmatrix} -\gamma\mathbf{I} - L_P & L_I^T \\ -L_I & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{x}(t) \\ \mathbf{w}(t) \end{bmatrix} + \begin{bmatrix} \gamma\mathbf{I} \\ \mathbf{0} \end{bmatrix} \mathbf{u}, \quad (7)$$

where $\mathbf{u} \in \mathbb{R}^n = (u_1, \ldots, u_n)^T$, $\mathbf{x}(t) \in \mathbb{R}^n = (x_1(t), \ldots, x_n(t))^T$ and $\mathbf{w}(t) \in \mathbb{R}^n = (w_1(t), \ldots, w_n(t))^T$ are the inputs and variables at the n nodes, L_P and L_I are respectively the proportional and the integral Laplacian weight matrices, and the parameter $\gamma > 0$ establishes the rate at which new information replaces old information. Note that in addition to the variable $x_i(t)$, each robot $i \in \mathcal{V}$ also maintains a second variable $w_i(t) \in \mathbb{R}$. More information of this PI algorithm can be found at the Appendix.

Discrete-time algorithms are more appropriate for the robot systems we consider. In this section we analyze a discrete-time version of the PI algorithm (7) with equal, symmetric, positive semidefinite Laplacian matrices $\mathcal{L} \in \mathbb{R}^n$ so that $L_P = L_I = \mathcal{L}$, $\mathcal{L}^T = \mathcal{L}$ and we let W be its associated weight matrix, $\mathcal{L} = \text{diag}(W\mathbf{1}) - W$. We analyze its convergence properties and its convergence speed depending on the step size h and the parameter γ. The theoretical results we give are general for any weighting matrix. We later extend them to the particular choice of the Metropolis weight matrix $W = W_M$ and its Laplacian matrix $\mathcal{L} = L_M$ given by eqs. (55)-(56) in the Appendix. From now on, we let $\mathbf{r} \in \mathbb{R}^n$ be the eigenvector of $\mathcal{L}$ associated to the eigenvalue $\lambda_1(\mathcal{L}) = 0$,

$$\mathbf{r} = \mathbf{1}/\sqrt{n}. \quad (8)$$

We let $S_2, \ldots, S_n$ be the remaining $n-1$ eigenvectors of $\mathcal{L}$ so that $[\mathbf{r}\ S_2 \ldots S_n] = [\mathbf{r}\ S]$ is a basis of eigenvectors of $\mathcal{L}$,

$$[\mathbf{r}\ S]^T \mathcal{L} [\mathbf{r}\ S] = \text{diag}\left(\lambda_1(\mathcal{L}), \ldots, \lambda_n(\mathcal{L})\right), \quad (9)$$

with the eigenvalues sorted as $\lambda_1(\mathcal{L}) \leq, \ldots, \leq \lambda_n(\mathcal{L})$. This orthonormal basis exists since $\mathcal{L}$ is symmetric with real entries. For connected communication graphs, all the other eigenvalues $\lambda_2(\mathcal{L}), \ldots, \lambda_n(\mathcal{L})$ are strictly greater than zero and we let $\mathcal{L}^{(-1)}$ be

$$\mathcal{L}^{(-1)} = (\mathbf{I} - \mathbf{r}\mathbf{r}^T)\left(\mathcal{L} + \mathbf{r}\mathbf{r}^T\right)^{-1}(\mathbf{I} - \mathbf{r}\mathbf{r}^T). \quad (10)$$

For all $i \in \mathcal{V}$ we let $b_i = b(\lambda_i(\mathcal{L}))$ be

$$b_i = b(\lambda_i(\mathcal{L})) = \sqrt{(\gamma + \lambda_i(\mathcal{L}))^2 - (2\ \lambda_i(\mathcal{L}))^2}. \quad (11)$$

The discrete-time consensus algorithm with constant scalar inputs, with equal and symmetric Laplacian matrices $\mathcal{L}$, and step size $h > 0$ is given by

$$\begin{bmatrix} \mathbf{x}(t+1) \\ \mathbf{w}(t+1) \end{bmatrix} = A \begin{bmatrix} \mathbf{x}(t) \\ \mathbf{w}(t) \end{bmatrix} + \begin{bmatrix} h\gamma\mathbf{I} \\ \mathbf{0} \end{bmatrix} \mathbf{u}, \quad \text{with} \quad (12)$$

$$A = \mathbf{I}_{2n\times 2n} + h \begin{bmatrix} -\gamma\mathbf{I} - \mathcal{L} & \mathcal{L} \\ -\mathcal{L} & \mathbf{0} \end{bmatrix}. \quad (13)$$

A more general form of the previous algorithm would consist of having the term $-\gamma\mathbf{I} - \mu\mathcal{L}$ instead of $-\gamma\mathbf{I} - \mathcal{L}$ in eq. (13), with the parameter $\mu > 0$ weighting the relative effects of the Proportional and Integral components. In the following we focus on the study of the system for the case $\mu = 1$ as in eq. (13) and we give conditions on parameters h, γ that ensure the convergence in real scenarios. The optimal combination of the proportional and integral weighting matrices depends on the graph, and on h and γ. The analysis of the properties for each case can be done as a replication of the theoretical analysis presented here. Note that this algorithm is fully distributed as each robot updates its states using information from its immediate neighbors. Along this section we will show that under certain conditions, the states at the nodes asymptotically converge to the average of the inputs, $x_i(t) \to x_{avg}$ as $t \to \infty$; equivalently in vector form, that $\mathbf{x}(t) \to \mathbf{1}x_{avg}$. We let the vectors $\mathbf{x}_*$ and $\mathbf{w}_* \in \mathbb{R}^n$ be

$$\begin{aligned} \mathbf{x}_* &= \mathbf{r}\mathbf{r}^T\mathbf{u} = \mathbf{1}x_{avg}, \\ \mathbf{w}_* &= \mathbf{r}\mathbf{r}^T\mathbf{w}(0) - \gamma\ \mathcal{L}^{(-1)}\mathbf{u}, \end{aligned} \quad (14)$$

where $\mathbf{r}$ and $\mathcal{L}^{(-1)}$ are given by eqs (8) and (10).

In order to analyze the convergence conditions and convergence speed of algorithm (12), we first analyze the eigenvalues of the system matrix A in eq. (13). The following result establishes a relationship between them and the eigenvalues of the Laplacian $\mathcal{L}$ associated to the weight matrix.

Proposition 1. *For each eigenvalue $\lambda_i(\mathcal{L})$ of the Laplacian $\mathcal{L}$ associated to the weight matrix, there exist eigenvalues $\lambda_i(A)$ and $\lambda_{n+i}(A)$ of the system matrix A in eq.* (13)*,*

$$\begin{aligned} \lambda_i(A) &= 1 - h\left(\gamma + \lambda_i(\mathcal{L}) + b_i\right)/2, \\ \lambda_{n+i}(A) &= 1 - h\left(\gamma + \lambda_i(\mathcal{L}) - b_i\right)/2, \end{aligned} \quad (15)$$

for $i \in \mathcal{V}$, being b_i given by (11)*. Note that for $\lambda_1(\mathcal{L}) = 0$, eq.* (15) *gives $\lambda_1(A) = 1 - h\gamma$ and $\lambda_{n+1}(A) = 1$.*

Proof: Denote $Z = \begin{bmatrix} -\gamma\mathbf{I} - \mathcal{L} & \mathcal{L} \\ -\mathcal{L} & \mathbf{0} \end{bmatrix}$, such that $A = \mathbf{I}_{2n\times 2n} + hZ$. The relationship between the eigenvalues and eigenvectors of A and Z for all $i \in \{1, \ldots, 2n\}$ is

$$\lambda_i(A) = 1 + h\lambda_i(Z), \qquad \mathbf{v}_i(A) = \mathbf{v}_i(Z). \quad (16)$$

We define the change of basis $Y = P^T Z P$, with

$$P = \begin{bmatrix} \mathbf{r}\ S_2 \ldots S_n & \mathbf{0} \\ \mathbf{0} & \mathbf{r}\ S_2 \ldots S_n \end{bmatrix}, \quad (17)$$

where $[\mathbf{r}\ S_2 \ldots S_n]$ is an orthonormal basis of eigenvectors of $\mathcal{L}$ as in eq. (9), so that the eigenvalues and eigenvectors of Z and Y are related by

$$\lambda_i(Z) = \lambda_i(Y), \qquad \mathbf{v}_i(Z) = P\ \mathbf{v}_i(Y). \quad (18)$$

We focus on the matrix Y,

$$Y = \begin{bmatrix} -\gamma\mathbf{I} - [\mathbf{r}\ S]^T\mathcal{L}[\mathbf{r}\ S] & [\mathbf{r}\ S]^T\mathcal{L}[\mathbf{r}\ S] \\ -[\mathbf{r}\ S]^T\mathcal{L}[\mathbf{r}\ S] & \mathbf{0} \end{bmatrix}, \quad (19)$$

which, because of eq. (9), has a sparse structure,

$$[\mathbf{r}\ S]^T\mathcal{L}[\mathbf{r}\ S] = \text{diag}(\lambda_1(\mathcal{L}),\dots,\lambda_n(\mathcal{L})),$$
$$-\gamma\mathbf{I} - [\mathbf{r}\ S]^T\mathcal{L}[\mathbf{r}\ S] = \text{diag}(-\gamma-\lambda_1(\mathcal{L}),\dots,-\gamma-\lambda_n(\mathcal{L})).$$

By solving for $Y\mathbf{v}_i(Y) = \lambda_i(Y)\mathbf{v}_i(Y)$, we get the following expression for the eigenvalues of Y, for all $i \in \mathcal{V}$:

$$\lambda_i(Y) = -(\gamma + \lambda_i(\mathcal{L}) + b_i)/2,$$
$$\lambda_{n+i}(Y) = -(\gamma + \lambda_i(\mathcal{L}) - b_i)/2. \tag{20}$$

Its eigenvectors $\mathbf{v}_i(Y)$, $\mathbf{v}_{n+i}(Y)$ have all its elements equal to zero but the $i-$th and the $(n+i)-$th components,

$$[\mathbf{v}_i(Y)]_i = 1, \qquad [\mathbf{v}_i(Y)]_{n+i} = -\lambda_i(\mathcal{L})/\lambda_i(Y),$$
$$[\mathbf{v}_{n+i}(Y)]_i = 1, \qquad [\mathbf{v}_{n+i}(Y)]_{n+i} = -\lambda_i(\mathcal{L})/\lambda_{n+i}(Y). \tag{21}$$

for $i \in \mathcal{V}$, except for $\mathbf{v}_{n+1}(Y)$, which has $[\mathbf{v}_{n+1}(Y)]_1 = 0$ and $[\mathbf{v}_{n+1}(Y)]_{n+1} = 1$. Combining eqs. (16),(18),(20), the expression for the eigenvalues of A in (15) is obtained. ■

The following result upper bounds the modulus of the eigenvalues of the system matrix A in eq. (13) by selecting appropriate values for the step size h and the parameter γ. This result is used later to prove the convergence of the system.

Proposition 2. *If the step size h and the parameter γ satisfy*

$$\gamma \geq (3/2)\lambda_n(\mathcal{L}), \tag{22a}$$
$$h\gamma < 3/2, \tag{22b}$$

where $\lambda_n(\mathcal{L})$ is the maximum eigenvalue of $\mathcal{L}$, then all the eigenvalues of A in (15) *are real. For connected communication graphs, all of them but $\lambda_{n+1}(A) = 1$ have modulus strictly less than one.*

Proof: The eigenvalues of A are related to the ones of $\mathcal{L}$ by (15) as stated by Proposition 1. All the eigenvalues $\lambda_i(\mathcal{L})$ are real and positive because $\mathcal{L}$ is positive semidefinite. Since both γ and h are real, the imaginary part of $\lambda_i(A)$ is $\pm\text{Im}\,[b_i]$, which is 0 because of (22a). As a result, all the eigenvalues of A are real. Note that imposing $\gamma \geq \lambda_n(\mathcal{L})$ instead of (22a) would make $\text{Im}\,[\lambda_i(A)] = 0$ as well. However, imposing $\gamma \geq (3/2)\lambda_n(\mathcal{L})$ forces $b_i \geq \gamma$ for all $i \in \{1,\dots,n\}$, and this greatly simplifies the characterization of the convergence speed in the remaining of this section.

Regarding the modulus, first note that for connected graphs $\lambda_i(\mathcal{L}) > 0$ for all $i \in \{2,\dots,n\}$. Due to (22a) b_i given by eq. (11) satisfies $\gamma \leq b_i \leq (2/\sqrt{3})\gamma$, and both $\lambda_i(Y)$ and $\lambda_{n+i}(Y)$ in eq. (20) decrease as i increases, satisfying $-(4/3)\gamma \leq \lambda_i(Y) < -\gamma$, $-(1/3)\gamma \leq \lambda_{n+i}(Y) < 0$, for all $i \in \{2,\dots,n\}$.

We first consider the eigenvalue $\lambda_1(A) = 1 - h\gamma$. It is strictly less than 1 since $h > 0$, $\gamma > 0$, and it is greater than $-1/2$ because of (22b). Then, its modulus is strictly less than 1. For all $i \in \{2,\dots n\}$, both $\lambda_i(A) = 1 + h\lambda_i(Y) < 1$ and $\lambda_{n+i}(A) = 1 + h\lambda_{n+i}(Y) < 1$, since $h > 0$ and $\lambda_i(Y) < 0, \lambda_{n+i}(Y) < 0$. Besides, $\lambda_i(A) \geq 1 - (4/3)h\gamma > -1$, and $\lambda_{n+i}(A) \geq 1 - (1/3)h\gamma > 1/2$. Then the modulus of both $\lambda_i(A)$ and $\lambda_{n+i}(A)$ are strictly less than 1. Finally, $\lambda_{n+1}(A) = 1$ as stated in Proposition 1. ■

In particular, the selection of $\gamma \geq 3$ and $h < 3/(2\gamma)$ when the Metropolis Laplacian matrix L_M (eqs. (55)-(56) in the Appendix) is used, satisfies Proposition 2 for any connected communication graph, since its eigenvalues satisfy $\lambda_1(L_M) = 0$ and $0 < \lambda_i(L_M) < 2$ for all $i \in \{2,\dots,n\}$.

We discuss now which one is the second eigenvalue $\lambda_{\text{eff}}(A)$ of A with maximum absolute value. Observe that $\lambda_{n+i}(A) \geq 1/2$ decreases as i increases, thus the greatest absolute value of $\lambda_{n+i}(A)$ for $i \in \{2,\dots,n\}$ is associated to $\lambda_{n+2}(A)$. Also $\lambda_i(A)$ decreases as i increases, and it takes both positive and negative values. For all i such that $\lambda_i(A) \geq 0$, the associated $\lambda_{n+i}(A)$ has greater modulus. For all i such that $\lambda_i(A) < 0$, the maximum absolute value is associated to $\lambda_n(A)$. We conclude that $\lambda_{\text{eff}}(A) = \max\{\lambda_{n+2}(A), -\lambda_n(A)\}$.

At this point we are ready to prove the convergence of algorithm (12) and to characterize its convergence speed.

Theorem 3. *Let $\mathcal{L}$ be the positive semidefinite Laplacian matrix associated to the connected undirected communication graph $\mathcal{G}$. Let us consider that the robots execute algorithm* (12) *with a step size $h > 0$ and a parameter $\gamma > 0$ as in Proposition 2. Then, for any input $\mathbf{u} \in \mathbb{R}^n$ and any initial states $\mathbf{x}(0) \in \mathbb{R}^n$, $\mathbf{w}(0) \in \mathbb{R}^n$, the states $\mathbf{x}(t) \in \mathbb{R}^n$, $\mathbf{w}(t) \in \mathbb{R}^n$ of the consensus algorithm* (12) *converge exponentially to*

$$\lim_{t\to\infty}\mathbf{x}(t) = \mathbf{x}_*, \qquad \lim_{t\to\infty}\mathbf{w}(t) = \mathbf{w}_*, \tag{23}$$

as $t \to \infty$, where $\mathbf{x}_*$ and $\mathbf{w}_*$ are given by (14)*. Moreover, if we let $\beta = 2\sqrt{10}/3$, then the error vector $e_{\mathbf{xw}}(t) = \left(\mathbf{x}(t)^T, \mathbf{w}(t)^T\right)^T - \left(\mathbf{x}_*^T, \mathbf{w}_*^T\right)^T$ after t iterations[2] satisfies*

$$\|e_{\mathbf{xw}}(t)\|_2 \leq \beta\ (\lambda_{\text{eff}}(A))^t\, \|e_{\mathbf{xw}}(0)\|_2\,. \tag{24}$$

Proof: First we prove the convergence. Let us assume that the relation in (24) is true. Since h and γ satisfy conditions (22a)-(22b) then, as stated by Proposition 2, $|\lambda_{n+1}(A)| = 1$ and the other eigenvalues have modulus strictly less than one $|\lambda_i(A)| < 1$. In particular, this is true for $\lambda_{\text{eff}}(A)$, and thus $(\lambda_{\text{eff}}(A))^t$ tends to 0 as $t \to \infty$ and the norm of the error $\|e_{\mathbf{xw}}(t)\|_2$ converges to zero.

Next, we prove that the error vector satisfies (24). Note that $\mathbf{x}_*$ and $\mathbf{w}_*$ satisfy

$$\begin{bmatrix} \mathbf{x}_* \\ \mathbf{w}_* \end{bmatrix} = A \begin{bmatrix} \mathbf{x}_* \\ \mathbf{w}_* \end{bmatrix} + \begin{bmatrix} h\gamma\mathbf{I} \\ \mathbf{0} \end{bmatrix} \mathbf{u}.$$

Therefore, the discrete time consensus algorithm (12) expressed in terms of the error $e_{\mathbf{xw}}(t)$,

$$e_{\mathbf{xw}}(t+1) = A\ e_{\mathbf{xw}}(t) + A \begin{bmatrix} \mathbf{x}_* \\ \mathbf{w}_* \end{bmatrix} + \begin{bmatrix} h\gamma\mathbf{I} \\ \mathbf{0} \end{bmatrix} \mathbf{u} - \begin{bmatrix} \mathbf{x}_* \\ \mathbf{w}_* \end{bmatrix},$$

$$\text{gives} \quad e_{\mathbf{xw}}(t+1) = A\ e_{\mathbf{xw}}(t), \qquad e_{\mathbf{xw}}(t) = A^t\ e_{\mathbf{xw}}(0),$$

where A^t is the $t-$th power of the system matrix A. We define the following change of basis,

$$C = P^T A P = \mathbf{I} + hY,$$

[2]All along this paper, we characterize the convergence speed for an even t in order to give more accurate bounds.

where P and Y are given by (17) and (19), and we let $e_{\mathbf{zy}}$ be the error in the new coordinates,

$$e_{\mathbf{zy}}(t) = P^T\ e_{\mathbf{xw}}(t), \qquad e_{\mathbf{xw}}(t) = P\ e_{\mathbf{zy}}(t), \tag{25}$$

which has the same Euclidean norm, $\|e_{\mathbf{xw}}(t)\|_2 = \|e_{\mathbf{zy}}(t)\|_2$. We focus on the system in the new coordinates,

$$e_{\mathbf{zy}}(t+1) = C\ e_{\mathbf{zy}}(t), \qquad e_{\mathbf{zy}}(t) = C^t\ e_{\mathbf{zy}}(0),$$

where the initial error is $e_{\mathbf{zy}}(0) = (\mathbf{z}(0)^T, \mathbf{y}(0)^T)^T - (\mathbf{z}_*^T, \mathbf{y}_*^T)^T$. By applying the change of basis to (23), the limit values in the new coordinates are

$$\mathbf{z}_* = \begin{bmatrix} \mathbf{r}^T\mathbf{u} \\ \mathbf{0}_{n-1} \end{bmatrix}, \quad \mathbf{y}_* = \begin{bmatrix} \mathbf{r}^T\mathbf{w}(0) \\ -\gamma(S^T\mathcal{L}S)^{-1}S^T\mathbf{u} \end{bmatrix}, \tag{26}$$

where $\mathbf{r}^T\mathbf{w}(0) = [\mathbf{y}(0)]_1$. Note that as a result the component $[e_{\mathbf{zy}}(0)]_{n+1}$ of the error in the new coordinates is zero. Let us decompose the initial error into a linear combination of the eigenvectors of C,

$$e_{\mathbf{zy}}(0) = a_1\ \mathbf{v}_1(C) + \cdots + a_{2n}\ \mathbf{v}_{2n}(C), \tag{27}$$

where it can be seen that for all $i \in \{1, \ldots, 2n\}$

$$\lambda_i(C) = \lambda_i(A), \text{ and} \qquad \mathbf{v}_i(C) = \mathbf{v}_i(Y), \tag{28}$$

being $\lambda_i(A)$ and $\mathbf{v}_i(C)$ given by respectively eqs. (15) and (21). Now we compute the coefficients a_i, a_{n+i} in eq. (27) as follows. Each pair $[e_{\mathbf{zy}}(0)]_i$, $[e_{\mathbf{zy}}(0)]_{n+i}$ of elements in $e_{\mathbf{zy}}(0)$ give two equations on a_i, a_{n+i}, for $i \in \mathcal{V}$,

$$\begin{aligned} [e_{\mathbf{zy}}(0)]_i &= a_i + a_{n+i}, \\ [e_{\mathbf{zy}}(0)]_{n+i} &= -a_i\lambda_i(\mathcal{L})/\lambda_i(Y) - a_{n+i}\lambda_i(\mathcal{L})/\lambda_{n+i}(Y), \end{aligned}$$

For $i = 1$, the previous equations give $[e_{\mathbf{zy}}(0)]_1 = a_1 + a_{n+1}$ and $[e_{\mathbf{zy}}(0)]_{n+1} = 0$. Thus, we can chose the first coefficient a_1 to be the first element in the error vector, $a_1 = [e_{\mathbf{zy}}(0)]_1$, and its associated $a_{n+1} = 0$. Proceeding in a similar fashion with the remaining coefficients a_i, a_{n+i}, for $i \in \{2, \ldots, n\}$, we get

$$\begin{aligned} a_i &= -\frac{\lambda_i(\mathcal{L})}{b_i}\left([e_{\mathbf{zy}}(0)]_{n+i} + \frac{\lambda_i(Y)}{\lambda_i(\mathcal{L})}[e_{\mathbf{zy}}(0)]_i\right), \\ a_{n+i} &= \frac{\lambda_i(\mathcal{L})}{b_i}\left([e_{\mathbf{zy}}(0)]_{n+i} + \frac{\lambda_{n+i}(Y)}{\lambda_i(\mathcal{L})}[e_{\mathbf{zy}}(0)]_i\right), \end{aligned} \tag{29}$$

where b_i is given by (11). With the initial error decomposed as in (27), the error after t iterations can be expressed as follows

$$e_{\mathbf{zy}}(t) = C^t\ e_{\mathbf{zy}}(0) = \sum_{i=1}^{2n} a_i\ (\lambda_i(C))^t\ \mathbf{v}_i(C),$$

which combined with (28) and (29) gives

$$[e_{\mathbf{zy}}(t)]_1 = (\lambda_1(A))^t\ [e_{\mathbf{zy}}(0)]_1, \qquad [e_{\mathbf{zy}}(t)]_{n+1} = 0,$$

and for all $i \in \{2, \ldots, n\}$,

$$\begin{aligned} [e_{\mathbf{zy}}(t)]_i &= c_{i,n+i}\ [e_{\mathbf{zy}}(0)]_{n+i} + c_{i,i}\ [e_{\mathbf{zy}}(0)]_i, \\ [e_{\mathbf{zy}}(t)]_{n+i} &= -c_{i,n+i}\ [e_{\mathbf{zy}}(0)]_i + c_{n+i,n+i}\ [e_{\mathbf{zy}}(0)]_{n+i}, \end{aligned} \tag{30}$$

$$\begin{aligned} \text{with } c_{i,i} &= \left[\lambda_{n+i}(Y)(\lambda_{n+i}(A))^t - \lambda_i(Y)(\lambda_i(A))^t\right]/b_i, \\ c_{i,n+i} &= \lambda_i(\mathcal{L})\left[(\lambda_{n+i}(A))^t - (\lambda_i(A))^t\right]/b_i, \\ c_{n+i,n+i} &= \left[-\lambda_i(Y)(\lambda_{n+i}(A))^t + \lambda_{n+i}(Y)(\lambda_i(A))^t\right]/b_i. \end{aligned} \tag{31}$$

The squared Euclidean norm $\|e_{\mathbf{zy}}(t)\|_2^2$ of the error vector at iteration t, is given by

$$\begin{aligned} &\|e_{\mathbf{zy}}(t)\|_2^2 = \sum_{i=1}^{n}([e_{\mathbf{zy}}(t)]_i)^2 + ([e_{\mathbf{zy}}(t)]_{n+i})^2 \\ &= (\lambda_1(A))^{2t}([e_{\mathbf{zy}}(0)]_1)^2 + \sum_{i=2}^{n}(c_{i,n+i}^2 + c_{i,i}^2)([e_{\mathbf{zy}}(0)]_i)^2 \\ &+ \sum_{i=2}^{n}(c_{i,n+i}^2 + c_{n+i,n+i}^2)([e_{\mathbf{zy}}(0)]_{n+i})^2 \\ &+ \sum_{i=2}^{n} 2c_{i,n+i}(c_{i,i} - c_{n+i,n+i})[e_{\mathbf{zy}}(0)]_i[e_{\mathbf{zy}}(0)]_{n+i}, \end{aligned} \tag{32}$$

where $2c_{i,n+i}(c_{i,i} - c_{n+i,n+i})$ is

$$= -2\lambda_i(\mathcal{L})(\gamma + \lambda_i(\mathcal{L}))\left((\lambda_{n+i}(A))^t - (\lambda_i(A))^t\right)^2/b_i^2.$$

Note that when k_1a and k_2b have the same sign, then $|k_1a - k_2b| \leq \max\{|k_1|, |k_2|\}\max\{|a|, |b|\}$. By taking into account that both $(\lambda_{n+i}(A))^t$ and $(\lambda_i(A))^t \geq 0$ for t even, and that $1/b_i \leq 1/\gamma$, $\lambda_i(\mathcal{L}) \leq (2/3)\gamma$, $\max\{-\lambda_i(Y), -\lambda_{n+i}(Y)\} \leq (4/3)\gamma$, then it can be seen that

$$\begin{aligned} &c_{i,n+i}^2 \leq \frac{2^2}{3^2}(\lambda_{\text{eff}}(A))^{2t}, \quad \max\{c_{n+i,n+i}^2, c_{i,i}^2\} \leq \frac{4^2}{3^2}(\lambda_{\text{eff}}(A))^{2t}, \\ &\text{and} \quad |2c_{i,n+i}(c_{i,i} - c_{n+i,n+i})| \leq (20/3^2)(\lambda_{\text{eff}}(A))^{2t}. \end{aligned}$$

$$\text{In addition, } \sum_{i=2}^{n}|[e_{\mathbf{zy}}(0)]_i[e_{\mathbf{zy}}(0)]_{n+i}|$$

$$\leq \sum_{i=2}^{n}(\max\{|[e_{\mathbf{zy}}(0)]_i|, |[e_{\mathbf{zy}}(0)]_{n+i}|\})^2 \leq \|e_{\mathbf{zy}}(0)\|_2^2.$$

Combining the previous results, we get

$$\|e_{\mathbf{zy}}(t)\|_2^2 \leq (40/9)\ (\lambda_{\text{eff}}(A))^{2t}\ \|e_{\mathbf{zy}}(0)\|_2^2. \tag{33}$$

Then, $\|e_{\mathbf{xw}}(t)\|_2 = \|e_{\mathbf{zy}}(t)\|_2$ satisfies

$$\|e_{\mathbf{xw}}(t)\|_2 \leq (2\sqrt{10}/3)\ (\lambda_{\text{eff}}(A))^t\ \|e_{\mathbf{xw}}(0)\|_2, \tag{34}$$

as in eq. (24) and the proof is completed. ■

Note that the convergence speed in Theorem 3 depends on $\lambda_{\text{eff}}(A) = \max\{\lambda_{n+2}(A), -\lambda_n(A)\}$, which is related to the eigenvalues $\lambda_2(\mathcal{L}), \lambda_n(\mathcal{L})$ of the Laplacian $\mathcal{L}$ of the communication graph. These eigenvalues depend on the graph topology and require global information of the network. Several distributed algorithms have been proposed [27], [28] that allow each node to compute these eigenvalues in a distributed fashion. Then, the agents can compute $\lambda_n(A)$, $\lambda_{n+2}(A)$ and find the one with the largest absolute value. In this case, they can also compute the optimal step size h^* such that $-\lambda_n(A) = \lambda_{n+2}(A)$,

$$h^* = 4/(2\gamma + \lambda_n(\mathcal{L}) + \lambda_2(\mathcal{L}) + b_n - b_2). \tag{35}$$

IV. DYNAMIC CONSENSUS WITH SCALAR INPUTS

Next we consider the dynamic scenario, where each robot $i \in \mathcal{V}$ has a scalar input $u_i^k \in \mathbb{R}$, whose value varies along the steps $k = 1, \ldots, K$. The global data $x_G^k \in \mathbb{R}$ is the sum of the inputs u_i^k at step k, and we let $x_{avg}^k \in \mathbb{R}$ be their average,

$$x_G^k = \sum_{i=1}^{n} u_i^k, \qquad x_{avg}^k = \frac{1}{n}\sum_{i=1}^{n} u_i^k = \frac{1}{n}x_G^k. \tag{36}$$

The goal is that, at every step k, each robot $i \in \mathcal{V}$ computes an estimate $x_i^k(t)$ that correctly tracks x_{avg}^k by local interactions with its neighbors $\mathcal{N}_i^k$. We adopt a strategy where, at each step $k = 1, \dots, K$, the robots run the consensus algorithm in Section III to compute the average of the inputs up to step k. The robots use the obtained states for initializing the consensus algorithm at the next step $k+1$. The robots execute a total number of L consensus iterations, divided into l iterations per input update step, and the remaining $L - l(K-1)$ after the last step. We assume that l is an even number so that the convergence rate in Theorem 3 holds.

Remark IV.1. Throughout the paper, we consider that the maximum number of consensus iterations L is limited by the problem requirements and it is a priori given to the robots. This value L may depend, e.g., on the amount of energy each robot has for carrying out its operation, the power consumption of each data exchange operation, and the energy assigned to other robot tasks. We consider that the number of iterations per step l is also established a priori. It may be selected so that the timespan of input update steps is the desired one, taking into account the time consumed by the computation and communication operations executed by the robots.

In case the robot team does not have any of the previous limitations, then the robots can select the desired l_k^* for each step so that their estimates reach a certain precision. For instance, if their goal is maintaining a relative estimation error of ϵ at each step k, $\|e_k^{\mathbf{xw}}(l_k^*)\|_2 \ / \ \|e_k^{\mathbf{xw}}(0)\|_2 \leq \epsilon$, then, from eq. (24), the desired value of l_k^* would be

$$l_k^* \geq \left(\log(\epsilon) - \log(\beta)\right) / \log\left(\lambda_{\text{eff}}(A)\right).$$

We do not specify the number of local observation-estimation iterations carried out by each robot between consecutive steps k and $k+1$. Using this strategy, if a map update step starts, and a robot is not ready for transmitting its updated local map, it can act as if it was disconnected from the communication network. •

From now on, we let $\mathcal{L}_k$ be the Laplacian which is associated to the communication graph $\mathcal{G}_k = (\mathcal{V}, \mathcal{E}_k)$ at step k and we let $[\mathbf{r} \ S_2^k \dots S_n^k] = [\mathbf{r} \ S_k]$ be a basis of eigenvectors of $\mathcal{L}_k$ as in eq. (9). Note that the eigenvector $\mathbf{r}$ is common to all the Laplacians $\mathcal{L}_k$. We let A_k be the system matrix associated to $\mathcal{L}_k$ given by (13). We also add the index k to the inputs $\mathbf{u}_k = (u_1^k, \dots, u_n^k)$ and the states $\mathbf{x}_k(t) = (x_1^k(t), \dots, x_n^k(t))$, $\mathbf{w}_k(t) = (w_1^k(t), \dots, w_n^k(t))$ of the consensus algorithm with constant inputs (12) to identify the associated step k. We define $\mathbf{x}_k^*$ and $\mathbf{w}_k^* \in \mathbb{R}^n$ as we did in the previous section but using the variables at step k,

$$\begin{aligned} \mathbf{x}_k^* &= \mathbf{r}\mathbf{r}^T\mathbf{u}_k = \mathbf{1}x_{avg}^k, \\ \mathbf{w}_k^* &= \mathbf{r}\mathbf{r}^T\mathbf{w}_k(0) - \gamma \ \mathcal{L}_k^{(-1)}\mathbf{u}_k, \end{aligned} \tag{37}$$

being $\mathbf{r}$, $\mathcal{L}_k^{(-1)}$ as in (8), (10), and let $\lambda_\star$ be

$$\lambda_\star = \max_{k \in \{1,\dots,K\}} \lambda_{\text{eff}}(A_k). \tag{38}$$

The proposed dynamic consensus algorithm is detailed in Algorithm 1, where the step size $h > 0$ and parameter $\gamma > 0$ of the consensus algorithm with constant inputs (12) (Algorithm 1, lines 6 and 13) are as in Proposition 2 for all k. In the same way that the consensus algorithm with constant inputs was fully distributed, the dynamic consensus algorithm is distributed as each robot updates its data by local interactions with its neighbors.

Algorithm 1 *Dynamic consensus algorithm* - Robot i

1: – *Initialization* at $k = 1$
2: $x_i^k(0) = 0$,$w_i^k(0) = 0$, $u_i^k \leftarrow$ current local map
3: –*Algorithm*
4: **for** each step $k = 1, \dots, K-1$ **do**
5: execute algorithm (12) for $t = l$ iterations:
6: $[x_i^k(t), w_i^k(t)] = \text{consensus\_alg}\left(u_i^k, x_i^k(0), w_i^k(0)\right)$
7: initialize the states with the previous estimates:
8: $x_i^k(0) = x_i^k(t)$, $w_i^k(0) = w_i^k(t)$,
9: $u_i^k \leftarrow$ current local map
10: **end for**
11: – *Final step* at $k = K$
12: execute (12) for the remaining $t = L-(K-1)l$ iterations:
13: $[x_i^k(t), w_i^k(t)] = \text{consensus\_alg}\left(u_i^k, x_i^k(0), w_i^k(0)\right)$

The rate of convergence for the dynamic consensus algorithm (Algorithm 1) depends on (i) the initial input and graph at $k = 1$, and (ii) the changes on both the input and the graph topology during consecutive steps. We let α and σ represent this information,

$$\alpha = \alpha_k \text{ for } k = 1, \quad \sigma = \max_{k \in \{1,\dots,K-1\}} \sigma_k, \tag{39}$$

$$\begin{aligned} \text{with } \alpha_k &= \left(\|\mathbf{r}^T\mathbf{u}_k\|_2^2 + \gamma^2\|\mathcal{L}_k^{(-1)}\mathbf{u}_k\|_2^2\right)^{1/2}, \\ \text{and } \sigma_k &= (\|\mathbf{r}^T(\mathbf{u}_k - \mathbf{u}_{k+1})\|_2^2 \\ &\quad + \gamma^2\|\mathcal{L}_k^{(-1)}\mathbf{u}_k - \mathcal{L}_{k+1}^{(-1)}\mathbf{u}_{k+1}\|_2^2)^{1/2}. \end{aligned} \tag{40}$$

As the following result states, under mild connectivity conditions on the communication graphs $\mathcal{G}_k$, the states $x_i^k(t)$ at each node i correctly track the average of the inputs x_{avg}^k for each k.

Theorem 4. *Assume all robots in $\mathcal{V}$ execute the dynamic consensus strategy detailed in Algorithm 1 and that their undirected communication graphs $\mathcal{G}_k$ are connected for any step $k \in \{1, \dots, K\}$. Then, the states $\mathbf{x}_k(t) \in \mathbb{R}^n$, $\mathbf{w}_k(t) \in \mathbb{R}^n$ of Algorithm 1 converge exponentially to*

$$\lim_{t\to\infty} \mathbf{x}_k(t) = \mathbf{x}_k^*, \qquad \lim_{t\to\infty} \mathbf{w}_k(t) = \mathbf{w}_k^*, \tag{41}$$

as $t \to \infty$, where $\mathbf{x}_k^*$ and $\mathbf{w}_k^*$ are given by (37)*. Moreover, the error vector $e_k^{\mathbf{xw}}(t) = [(\mathbf{x}_k(t))^T, (\mathbf{w}_k(t))^T]^T - [(\mathbf{x}_k^*)^T, (\mathbf{w}_k^*)^T]^T$ for each step $k \in \{1, \dots, K\}$ after t iterations, with t even, satisfies*

$$\|e_k^{\mathbf{xw}}(t)\|_\infty \leq \|e_k^{\mathbf{xw}}(t)\|_2 \leq \alpha f_k(t) + \sigma g_k(t), \tag{42}$$

$$f_k(t) = (\beta)^k(\lambda_\star)^{t+(k-1)l}, \quad g_k(t) = \beta(\lambda_\star)^t \sum_{p=0}^{k-2}(\beta(\lambda_\star)^l)^p,$$

where l is the number of iterations of the consensus algorithm with constant inputs executed per input update step, $\beta = 2\sqrt{10}/3$, and $\lambda_\star$, α and σ are given by eqs. (38) *and* (39)*.*

Proof: The convergence of the states $\mathbf{x}_k(t) \in \mathbb{R}^n$ and $\mathbf{w}_k(t) \in \mathbb{R}^n$ to $\mathbf{x}_k^*$ and $\mathbf{w}_k^*$ in (37) is a consequence of Theorem 3. Regarding the convergence rate, as stated by Theorem 3 the error vector after l iterations satisfies

$$\|e_k^{\mathbf{xw}}(l)\|_2 \leq \beta(\lambda_\star)^l \, \|e_k^{\mathbf{xw}}(0)\|_2 \,. \tag{43}$$

The final error vector $e_k^{\mathbf{xw}}(l)$ at step k and the initial error vector $e_{k+1}^{\mathbf{xw}}(0)$ at the next step $k+1$ are related as follows:

$$e_{k+1}^{\mathbf{xw}}(0) = e_k^{\mathbf{xw}}(l) + \begin{bmatrix} \mathbf{x}_k^* \\ \mathbf{w}_k^* \end{bmatrix} - \begin{bmatrix} \mathbf{x}_{k+1}^* \\ \mathbf{w}_{k+1}^* \end{bmatrix}. \tag{44}$$

Combining the previous results we get

$$\begin{aligned} \|e_k^{\mathbf{xw}}(t)\|_2 &\leq (\beta)^k (\lambda_\star)^{t+(k-1)l} \, \|e_1^{\mathbf{xw}}(0)\|_2 \\ &+ \beta(\lambda_\star)^t \sum_{p=0}^{k-2} (\beta(\lambda_\star)^l)^p \left\| \begin{bmatrix} \mathbf{x}_{k-p-1}^* - \mathbf{x}_{k-p}^* \\ \mathbf{w}_{k-p-1}^* - \mathbf{w}_{k-p}^* \end{bmatrix} \right\|_2, \end{aligned} \tag{45}$$

where at step $k = 1$ the states are initialized with zeros (Algorithm 1, line 2), and thus the initial error at step $k = 1$ and iteration $t = 0$ is $e_1^{\mathbf{xw}}(0) = [(-\mathbf{x}_1^*)^T, (-\mathbf{w}_1^*)^T]^T$. We compute the norm of the initial error $\|e_1^{\mathbf{xw}}(0)\|_2$ and obtain α in eq. (39), where we have used the fact that $\left\|(a^T, b^T)^T\right\|_2^2 = \|a\|_2^2 + \|b\|_2^2$. Proceeding in a similar fashion, for $k = 1, \ldots, K-1$, the norms $\left\|((\mathbf{x}_k^* - \mathbf{x}_{k+1}^*)^T, (\mathbf{w}_k^* - \mathbf{w}_{k+1}^*)^T)^T\right\|_2$ are equal to σ_k in eq. (40), and thus they are smaller than σ in eq. (39). We finally obtain the expression in eq. (42). ∎

The interest of the proposed method is that the estimates at the previous step $k-1$ are used for initializing their estimates at step k. Looking at the rate of convergence,

$$(\beta)^k (\lambda_\star)^{t+(k-1)l} \alpha_1 + \beta(\lambda_\star)^t \sum_{p=0}^{k-2} (\beta(\lambda_\star)^l)^p \sigma_{k-p-1},$$

errors associated to previous steps $(\beta)^k (\lambda_\star)^{t+(k-1)l} \alpha_1$, and $\beta(\lambda_\star)^t (\beta(\lambda_\star)^l)^{k-p-1} \sigma_p$, for $p = 1, \ldots, k-2$, are small since they have already been reduced by the execution of the algorithm. The last step error $\beta(\lambda_\star)^t \sigma_{k-1}$ depends on the variation of the input and graph topology between steps $k-1$ and k. Consider instead a zero-initialization strategy, where at each step k the robots discard their old estimates (initializing their estimates with zeros). The rate of convergence of this zero-initialization strategy would be given by $\beta(\lambda_\star)^t \alpha_k$, where the term α_k depends on the input and the graph itself (eq. (39)). Therefore, if the variation of inputs and graph topologies σ_k are small compared to the input itself α_k, then the dynamic consensus algorithm is preferable to the zero-initialization strategy.

Equivalently, we briefly discuss the behavior of the algorithm under changes in the communication graph. Consider that after $t < l$ iterations we let the graph change. This is equivalent to having a new step $k+1$ with a smaller l, and with the new graph $\mathcal{G}_{k+1}$ and with the same input, $\mathbf{u}_k = \mathbf{u}_{k+1}$. In this case, the additional error introduced due to the graph change σ_{k+1} in eq. (40) is $\gamma \|\mathcal{L}_{k+1}^{(-1)} \mathbf{u}_k\|_2$. Therefore, as long as the changes in the topology σ_{k+1} are small and slow enough compared to the number of iterations t and l, the algorithm will correctly track the average of the inputs.

V. CONSENSUS ON FEATURE-BASED MAPS

We extend the dynamic consensus strategy (Algorithm 1) presented in Section IV to operate on matrices and vectors instead of on scalar inputs. This generalization is key for merging feature-based stochastic IF maps. The local maps to be merged given by (3) are represented by a $\mathcal{M}_G \times \mathcal{M}_G$ information matrix and an information vector of size $\mathcal{M}_G$. The robots execute in parallel many instances of Algorithm 1 on each entry (r, s) within its information matrix and on each r entry within its information vector, for $r, s \in \{1, \ldots, \mathcal{M}_G\}$[3]. Let us add the subscripts $\{I, r, s\}$ or $\{\mathbf{i}, r\}$ to the variables $u_i^k, x_i^k(t), w_i^k(t)$ to identify the instance we are referring to. At step k, each entry within the information matrix $[I_i^k]_{rs}$ and vector $[\mathbf{i}_i^k]_r$ of the local map of robot $i \in \mathcal{V}$ (3) is used as an input for an instance of Algorithm 1,

$$u_{i\ \{I,r,s\}}^k = [I_i^k]_{rs}, \qquad u_{i\ \{\mathbf{i},r\}}^k = [\mathbf{i}_i^k]_r,$$

for $r, s \in \{1, \ldots, \mathcal{M}_G\}$. Each robot i executing Algorithm 1 computes an estimate $x_i^k(t)$ of the average x_{avg}^k of the inputs u_j^k (Section IV, eqs. (36), (37)). We arrange the states $x_i^k(t)$ of all the instances of Algorithm 1 into the following temporal averaged information matrix $I_{avg,i}^k(t) \in \mathbb{R}^{\mathcal{M}_G \times \mathcal{M}_G}$ and vector $\mathbf{i}_{avg,i}^k(t) \in \mathbb{R}^{\mathcal{M}_G}$ of robot i at iteration t,

$$[I_{avg,i}^k(t)]_{rs} = x_{i\ \{I,r,s\}}^k(t), \quad [\mathbf{i}_{avg,i}^k(t)]_r = x_{i\ \{\mathbf{i},r\}}^k(t),$$

for $r, s \in \{1, \ldots, \mathcal{M}_G\}$. We define the information matrix $I_{G,i}^k(t) \in \mathbb{R}^{\mathcal{M}_G \times \mathcal{M}_G}$ and vector $\mathbf{i}_{G,i}^k(t) \in \mathbb{R}^{\mathcal{M}_G}$, mean $\hat{\mathbf{x}}_{G,i}^k(t) \in \mathbb{R}^{\mathcal{M}_G}$, and covariance $\Sigma_{G,i}^k(t) \in \mathbb{R}^{\mathcal{M}_G \times \mathcal{M}_G}$, of the global map of robot i at iteration t and step k, as

$$\begin{aligned} I_{G,i}^k(t) &= n I_{avg,i}^k(t), \quad \hat{\mathbf{x}}_{G,i}^k(t) = \left(I_{avg,i}^k(t)\right)^{-1} \mathbf{i}_{avg,i}^k(t), \\ \mathbf{i}_{G,i}^k(t) &= n \mathbf{i}_{avg,i}^k(t), \quad \Sigma_{G,i}^k(t) = \left(I_{avg,i}^k(t)\right)^{-1} / n. \end{aligned} \tag{46}$$

Recall our discussion in Section II about the two parts of the dynamic map merging: (i) propagating the rows and columns of I_i^k, $\mathbf{i}_i^k$ associated the pose of a robot j; and (ii) reaching consensus on the entries associated exclusively to features through the instances of Algorithm 1. In eq. (46) we are assuming that the information concerning the poses of the robots has already been received by the robots and incorporated into their information matrices and vectors.

For simplicity, we are presenting the structures of the information matrices and vectors $\mathbf{i}_{avg,i}^k(t)$, $I_{avg,i}^k(t)$, as fixed and known by all the robots. Actually, the robots discover the features observed by the others in the messages exchanged at each iteration, and introduce new columns and rows in $\mathbf{i}_{avg,i}^k(t)$, $I_{avg,i}^k(t)$ accordingly. A brief discussion of this issue appears in Section V-A. The interest is that $I_{avg,i}^k(t)$ in eq. (46) can be inverted at each iteration of the algorithm and thus the global map can always be estimated.

The following two results regarding the convergence of the map merging algorithm are a consequence of Theorem 4.

[3]Initially, this would suppose a total of $\mathcal{M}_G^2 + \mathcal{M}_G$ instances of the consensus algorithm. However, since the information matrix is symmetric, it only has $\frac{1}{2}\mathcal{M}_G(\mathcal{M}_G + 1)$ different entries. Therefore, the robots actually execute $\mathcal{M}_G + \frac{1}{2}\mathcal{M}_G(\mathcal{M}_G + 1)$ instances of the consensus algorithm instead of $\mathcal{M}_G + \mathcal{M}_G^2$.

Corollary 5. *Assume all the robots $i \in \mathcal{V}$ execute the dynamic consensus (Algorithm 1) on each entry of its information matrix and vector as detailed above, and assume that their undirected communication graphs $\mathcal{G}_k$ are connected for all $k \in \{1,\dots,K\}$. Then, for the last step K, the estimated information matrix $I^K_{G,i}(t)$, information vector $\mathbf{i}^K_{G,i}(t)$, mean $\hat{\mathbf{x}}^K_{G,i}(t)$, and covariance $\Sigma^K_{G,i}(t)$, at each robot $i \in \mathcal{V}$ asymptotically converge to the information matrix I^K_G, information vector $\mathbf{i}^K_G$, mean $\hat{\mathbf{x}}^K_G$, and covariance Σ^K_G, of the global map given by eqs.* (4) *and* (5)*,*

$$\lim_{t\to\infty} I^K_{G,i}(t) = I^K_G, \qquad \lim_{t\to\infty} \hat{\mathbf{x}}^K_{G,i}(t) = \hat{\mathbf{x}}^K_G,$$
$$\lim_{t\to\infty} \mathbf{i}^K_{G,i}(t) = \mathbf{i}^K_G, \qquad \lim_{t\to\infty} \Sigma^K_{G,i}(t) = \Sigma^K_G. \tag{47}$$

Corollary 6. *Assume all the robots $i \in \mathcal{V}$ execute the dynamic consensus (Algorithm 1) on each entry of its information matrix and vector as detailed above, and assume that their undirected communication graphs $\mathcal{G}_k$ are connected for all $k \in \{1,\dots,K\}$. Let I^k_{avg}, $\mathbf{i}^k_{avg}$ be the average of the local maps in IF form at step k,*

$$I^k_{avg} = \frac{1}{n}\sum_{j=1}^{n} I^k_j, \qquad \mathbf{i}^k_{avg} = \frac{1}{n}\sum_{j=1}^{n} \mathbf{i}^k_j. \tag{48}$$

Let $\alpha_{\{I,r,s\}}$ and $\sigma_{\{I,r,s\}}$, be defined as α, σ in (39) *for the inputs $u^k_{i\ \{I,r,s\}} = [I^k_i]_{rs}$; equivalently $\alpha_{\{\mathbf{i},r\}}$, $\sigma_{\{\mathbf{i},r\}}$ for the inputs $u^k_{i\ \{\mathbf{i},r\}} = [\mathbf{i}^k_i]_r$. We let α_I and σ_I, respectively $\alpha_{\mathbf{i}}$ and $\sigma_{\mathbf{i}}$, be the maximum over all the entries of I, respectively of $\mathbf{i}$,*

$$\alpha_I = \max_{r,s\in\{1,\dots,\mathcal{M}_G\}} \alpha_{\{I,r,s\}}, \quad \sigma_I = \max_{r,s\in\{1,\dots,\mathcal{M}_G\}} \sigma_{\{I,r,s\}},$$
$$\alpha_{\mathbf{i}} = \max_{r\in\{1,\dots,\mathcal{M}_G\}} \alpha_{\{\mathbf{i},r\}}, \qquad \sigma_{\mathbf{i}} = \max_{r\in\{1,\dots,\mathcal{M}_G\}} \sigma_{\{\mathbf{i},r\}}. \tag{49}$$

Then, for all $i \in \mathcal{V}$, $k \in \{1,\dots,K\}$, $r,s \in \{1,\dots,\mathcal{M}_G\}$, $t \geq 0$, the entry $[I^k_{avg,i}(t)]_{rs}$ within the averaged information matrix and the entry $[\mathbf{i}^k_{avg,i}(t)]_r$ within the averaged information vector estimated by robot i after t iterations satisfy

$$|[I^k_{avg,i}(t)]_{rs} - [I^k_{avg}]_{rs}| \leq \alpha_I f_k(t) + \sigma_I g_k(t),$$
$$|[\mathbf{i}^k_{avg,i}(t)]_r - [\mathbf{i}^k_{avg}]_r| \leq \alpha_{\mathbf{i}} f_k(t) + \sigma_{\mathbf{i}} g_k(t), \tag{50}$$

where the convergence speed expressions $f_k(t)$, $g_k(t)$, are defined in Theorem 4, eq. (42)*.*

Next we present an interesting property of the map merging algorithm. As the following result shows, the temporal global maps $\hat{\mathbf{x}}^k_i(t)$ estimated at each robot i, are unbiased estimates of the true feature positions $\mathbf{x}$. As a result, the robots do not need to wait for any specific number of iterations of the map merging algorithm. Instead, they can make decisions on their temporal global map estimates whenever they need.

Proposition 7. *The estimates of the global map mean $\hat{\mathbf{x}}^k_{G,i}(t)$, for each robot $i \in \mathcal{V}$, at a step $k \in \{1,\dots,K\}$, after t iterations of the dynamic consensus algorithm, are unbiased estimates of the true feature positions $\mathbf{x}$,*

$$\mathrm{E}\left[\hat{\mathbf{x}}^k_{G,i}(t)\right] = \mathrm{E}\left[\left(I^k_{avg,i}(t)\right)^{-1} \mathbf{i}^k_{avg,i}(t)\right] = \mathbf{x}. \tag{51}$$

Proof: The temporal values of $I^k_{avg,i}(t)$, $\mathbf{i}^k_{avg,i}(t)$, that evolve according to Algorithm 1, can be alternatively expressed as a function of the inputs $I^1_j,\dots,I^K_j$, $\mathbf{i}^1_j,\dots,\mathbf{i}^K_j$, (3), and the initial states. Since the states at $k=1$ and $t=0$ are zero (Algorithm 1, line 2), then $I^k_{avg,i}(t)$ and $\mathbf{i}^k_{avg,i}(t)$ are

$$I^k_{avg,i}(t) = \sum_{j=1}^{n} [\Phi(k,t)]_{ij}\ I^k_j + \sum_{p=1}^{k-1}\sum_{j=1}^{n} [\Omega(k,t,p)]_{ij}\ I^p_j,$$
$$\mathbf{i}^k_{avg,i}(t) = \sum_{j=1}^{n} [\Phi(k,t)]_{ij}\ \mathbf{i}^k_j + \sum_{p=1}^{k-1}\sum_{j=1}^{n} [\Omega(k,t,p)]_{ij}\ \mathbf{i}^p_j, \tag{52}$$

where the matrices $\Phi(k,t), \Omega(k,t,p), \Psi(t_1,t_2) \in \mathbb{R}^{2n\times 2n}$ are

$$\Phi(k,t) = \sum_{\tau=1}^{t} \Psi\left(\tau + (k-1)l, t-1+(k-1)l\right) \begin{bmatrix} h\gamma\mathbf{I} \\ \mathbf{0} \end{bmatrix},$$
$$\Omega(k,t,p) = \sum_{\tau=1}^{l} \Psi\left(\tau + (p-1)l, t-1+(k-1)l\right) \begin{bmatrix} h\gamma\mathbf{I} \\ \mathbf{0} \end{bmatrix},$$
$$\Psi(t_1,t_2) = A(t_2)\dots A(t_1+1)A(t_1), \quad \text{for } t_1 < t_2,$$
$$\Psi(t_1,t_1) = A(t_1), \quad \text{and } \Psi(t_1,t_2) = \mathbf{I}, \quad \text{for } t_1 > t_2,$$

and $A(t+kl) = A_k(t)$ is the system matrix associated to the iteration t and step k given by (13). Since the local maps $\hat{\mathbf{x}}^k_j$ at each robot j are an estimate of the true $\mathbf{x}$ (2),

$$\hat{\mathbf{x}}^k_j = H^k_i \mathbf{x} + \mathbf{v}^k_j, \text{ with} \qquad \mathrm{E}\left[\mathbf{v}^k_j\right] = 0,$$

then the inputs $\mathbf{i}^k_j = (H^k_j)^T(\Sigma^k_j)^{-1}\hat{\mathbf{x}}^k_j$ are

$$\mathbf{i}^k_j = (H^k_j)^T(\Sigma^k_j)^{-1}\mathbf{v}^k_j + I^k_j\mathbf{x}. \tag{53}$$

Combining eqs. (52) and (53), variables $\mathbf{i}^k_{avg,i}(t)$ are given by

$$\mathbf{i}^k_{avg,i}(t) = \sum_{j=1}^{n} [\Phi(k,t)]_{ij}(H^k_j)^T(\Sigma^k_j)^{-1}\mathbf{v}^k_j +$$
$$\sum_{p=1}^{k-1}\sum_{j=1}^{n} [\Omega(k,t,p)]_{ij}(H^{k-p}_j)^T(\Sigma^{k-p}_j)^{-1}\mathbf{v}^{k-p}_j +$$
$$\left(\sum_{j=1}^{n} [\Phi(k,t)]_{ij}\ I^k_j + \sum_{p=1}^{k-1}\sum_{j=1}^{n} [\Omega(k,t,p)]_{ij}\ I^p_j\right)\mathbf{x},$$

where the last term is exactly $I^k_{avg,i}(t)\mathbf{x}$, with $I^k_{avg,i}(t)$ as in eq. (52). Then $\hat{\mathbf{x}}^k_{G,i}(t) = (I^k_{avg,i}(t))^{-1}\mathbf{i}^k_{avg,i}(t)$ is

$$\hat{\mathbf{x}}^k_{G,i}(t) = \mathbf{x} + (I^k_{avg,i}(t))^{-1}\left(\sum_{j=1}^{n} [\Phi(k,t)]_{ij}(H^k_j)^T(\Sigma^k_j)^{-1}\mathbf{v}^k_j\right) +$$
$$(I^k_{avg,i}(t))^{-1}\left(\sum_{p=1}^{k-1}\sum_{j=1}^{n} [\Omega(k,t,p)]_{ij}(H^{k-p}_j)^T(\Sigma^{k-p}_j)^{-1}\mathbf{v}^{k-p}_j\right).$$

Since the noises $\mathbf{v}^k_j$ have zero mean for all $k \in \{1,\dots,K\}$ and all $j \in \mathcal{V}$, the expected value of $\hat{\mathbf{x}}^k_{G,i}(t)$ is $\mathbf{x}$. ■

Note that this property holds also for time-varying graphs, where $A(t+kl)$ is different for each iteration t and each step k. For fixed graphs, $\Psi\left(t_1+kl, t_2+kl\right)$ is simply $(A_k)^{t_2-t_1+1}$.

We end this section by discussing two additional issues that arise in map merging scenarios: the initial correspondence problem, and the data association and feature labeling.

A. Initial correspondence and data association

The expression in eq. (3) implicitly assumes that the local maps are expressed in a common reference frame. This issue is related to initial correspondence or map alignment problems. The robots usually start their operation at unknown poses and, before merging their maps, they must agree on a common reference frame. This common frame needs to be computed at least once, and usually only requires the robots to know the relative pose of its nearby teammates, see e.g., [29]–[31] where different methods for computing robot-to-robot measurements are presented. There exist several distributed algorithms that combine these measurements to produce the common frame, e.g., [32]–[34], or our recent work [35] and references therein.

The data association consists of establishing a relationship between the features observed by the different robots. In this paper, for simplicity, we have presented the formulation as if the data association had been previously given to the robots, encoded in the observation matrices H_i^k in eq. (3). One of the benefits of performing the merging of information in the form of stochastic maps is that the distributed data association method in [17], [23] which is suitable for feature-based maps, can be executed by the robots at the beginning of each step k. This method can be integrated with a wide variety of local matchers (Nearest Neighbor, Maximum Likelihood, Joint Compatibility Branch and Bound, etc.). Each feature is assigned a label in such a way that during the merging process, any two features with the same label are merged together. Robots exchange these labels together with the maps. When they discover new features in the information received from their neighbors, they introduce additional rows and columns in the information matrices and vectors for them. If a robot has never received information of a feature, e.g., cause it has been observed by a distant robot, it simply does not have any space for it in its information matrix and vector. As a result, the information matrices and vectors do not contain non-informative zero rows and columns. Information matrices can be inverted at each iteration of the algorithm and thus the global map can always be estimated. Due to the limited space, we do not discuss this data association algorithm in detail here. The reader is referred to [17], [23] for further information.

B. Complexity analysis

We analyze the algorithm complexity regarding execution time and amount of communication required. Let $\mathcal{M}_{\max}$ be the highest size of the local map of any robot, and $d_{\max}$ be the highest number of neighbors of any robot,

$$\mathcal{M}_{\max} = \max_{\substack{i\in\{1,\dots,n\},\\ k\in\{1,\dots,K\}}} \mathcal{M}_i^k, \qquad d_{\max} = \max_{\substack{i\in\{1,\dots,n\},\\ k\in\{1,\dots,K\}}} |\mathcal{N}_i^k|.$$

The *computational* complexity per iteration and robot is $O(d_{\max}\mathcal{M}_G^2)$, employed in the addition of at most $d_{\max}$ information matrices of size $\mathcal{M}_G \times \mathcal{M}_G$. The *communication* complexity per iteration and robot is $O(\mathcal{M}_G^2)$, employed by the robot in sending its information matrix to the network. Since we keep the local maps independent, each information matrix $I_{G,i}^k(t)$ remains sparse, with the significant coefficients grouped around the main diagonal. Therefore, if a compression algorithm is used, the cost of sending matrix $I_{G,i}^k(t)$ to the network and the computational costs for processing the neighbors' data can be expressed respectively as $O(n\mathcal{M}_{\max}^2)$ and $O(d_{\max}n\mathcal{M}_{\max}^2)$. Alternatively, the communication and computational costs can be expressed respectively as $O(n+m)$ and $O(d_{\max}(n+m))$ if we consider the local map sizes as constants.

VI. EXPERIMENTS

The behavior of our dynamic map merging method is analyzed with real data. We use a data set [36] with bearing information obtained with vision (Sony EVI-371DG) in an environment of 60×45 meters. There total length of the robot path is 505 meters and it is divided into 3297 steps. It is an indoor scenario, where the robot moves along corridors and 29 rooms. The data set contains real odometry data and images captured at every step. The images are processed and measurements to natural landmarks are provided. The natural landmarks are vertical lines (Fig. 2) extracted from the images and processed in the form of bearing-only data. The observations in the dataset are labeled so that we have the

Fig. 2. An example of the images used by the 8 robots during the navigation to test the proposed method [36]. We test the algorithm using the lines extracted from natural landmarks (in yellow).

ground-truth data association. There are 1406 different vertical lines labeled in the scene. We select 8 subsections of the whole path for the operation of 8 different robots (Fig. 8 (a)). We run a separate SLAM in each robot. We use a recursive filtering SLAM algorithm (not discussed here), for planar bearing-only data, with features parameterized in inverse-depth [37] followed by a transform to Cartesian coordinates before each merging process. The robots execute the proposed algorithm for merging their local maps communicating through range-limited graphs as in Fig. 3, with the Metropolis weights given by eqs. (55)-(56) in the Appendix, and with the parameters $\gamma = 1.8$ and $h = 0.8$. In this experiment, we get $\lambda_\star = 0.97$. They execute a total of $L = 500$ consensus iterations. The robots run a total of $K = 5$ map update steps. Between consecutive map update steps k, $k+1$, each robot performs 10 steps of the bearing-only SLAM algorithm (Fig. 4).

The algorithm is executed for 3 different configurations. In the first one, the robots execute a small number of consensus iterations $l = 25$ after each map update step $k = 1, \dots, 4$, and the remaining $L - (K-1)l = 400$ iterations after the last one. In the second case, they use $l = 50$ and execute the remaining 300 at $k = K$. And in the last one, they use an equal number of iterations per step $l = (L/K) = 100$.

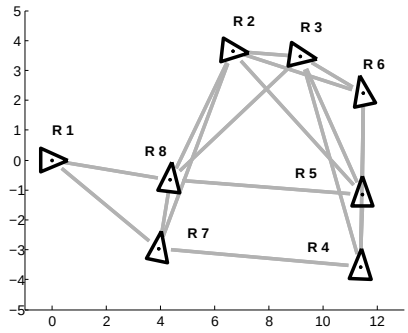


Fig. 3. Sample communication graph. There is an edge (gray lines) between two robots (triangles) if their distance is smaller than $7.5\ m$.

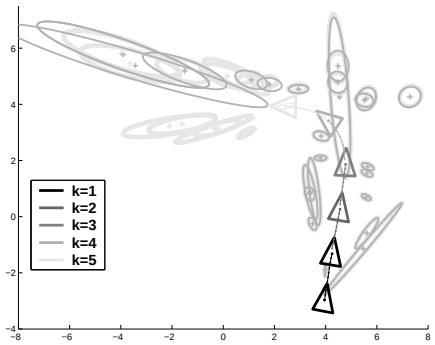
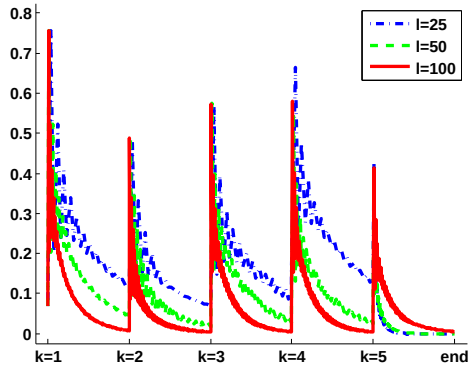
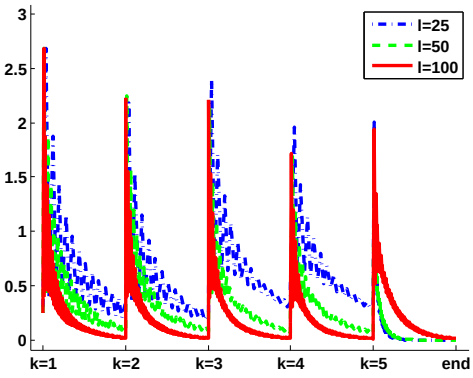


Fig. 4. Local map of robot R7. For clarity, feature estimates are only displayed for steps $k = 4$ (gray) and $k = 5$ (light gray). R7 initiates a new map update step k after executing 10 motions (triangles). During these motions, new features are introduced in the map, and previous features estimates are improved.



(a) $\max_{i,r,s} |[I^k_{avg,i}(t)]_{rs} - [I^k_{avg}]_{rs}|/\sigma_I$.



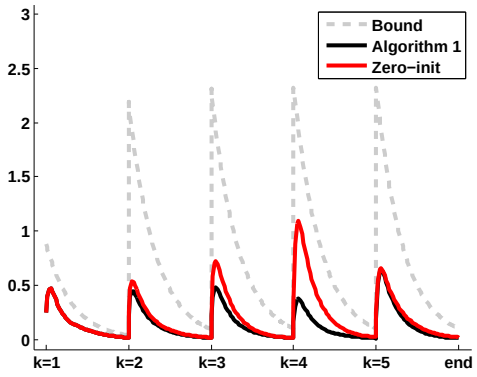
(b) $\max_{i,r} |[\mathbf{i}^k_{avg,i}(t)]_r - [\mathbf{i}^k_{avg}]_r|/\sigma_{\mathbf{i}}$.

Fig. 5. Estimation errors along the L consensus iterations. (a) $\max_{i,r,s} |[I^k_{avg,i}(t)]_{rs} - [I^k_{avg}]_{rs}|/\sigma_I$, (b) $\max_{i,r} |[\mathbf{i}^k_{avg,i}(t)]_r - [\mathbf{i}^k_{avg}]_r|/\sigma_{\mathbf{i}}$. The configuration $l = 25$ (blue dashed-dotted) maintains a high error along the intermediate steps $k = 1, \ldots, K-1$, but at the last step, it gets a high precision. For $l = 100$ (red solid) the precision at the end of each intermediate step $k = 1, \ldots, K-1$ is very high, but finishes with the worst final error. The configuration $l = 50$ (green dashed) produces accurate results during both the last and the intermediate steps.

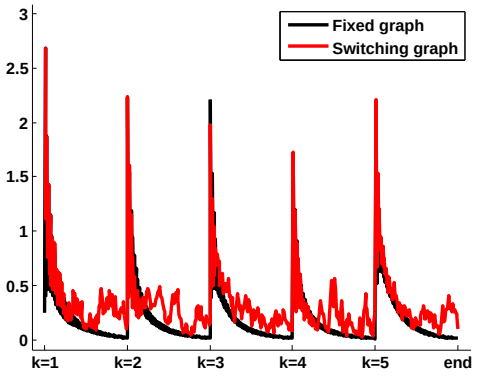
The obtained scaled estimation errors for the information matrices $|[I^k_{avg,i}(t)]_{rs} - [I^k_{avg}]_{rs}|/\sigma_I$ and information vectors $|[\mathbf{i}^k_{avg,i}(t)]_r - [\mathbf{i}^k_{avg}]_r|/\sigma_{\mathbf{i}}$ are displayed in Fig. 5 (a) and (b) along the L consensus iterations. During the intermediate steps $k = 1, \ldots, K-1$, the configuration $l = 100$ (red solid) exhibits the fastest convergence, whereas $l = 50$ (green dashed) also produces good results. The configuration $l = 25$ (blue dashed-dotted) however is less precise and its estimates are further from the average value. During the last step $k = K$ both $l = 25$ and $l = 50$ configurations reach a small final error. However, the configuration $l = 100$ which was reaching the best results during the previous steps, finishes with the worst final error. The configuration $l = 50$ produces interesting results since the intermediate errors are almost as good as for $l = 100$, whereas the final error is similar to the obtained by $l = 25$.

We compare the behavior of the dynamic consensus algorithm with a zero-initialization strategy (Fig. 6 (a)). The errors associated to the information vectors for even iteration numbers t are shown for both, our dynamic consensus algorithm with $l = 100$ (black solid), and the zero-initialization strategy with $l = 100$ (red solid). For $k = 1$ both errors are equal since the dynamic consensus algorithm performs a zero-initialization. For the other steps $k = 2, \ldots, K$, the errors of our proposed algorithm are smaller than the ones obtained with the zero-initialization strategy. They are upper bounded by the theoretical rate of convergence in eq. (50) (gray dashed). We analyze the behavior of the algorithm under time-varying communication graphs (Fig. 6 (b)). Robots exchange data according to the communication graph $\mathcal{G}$ in Fig. 3. At each iteration t and step k, one of the links $\mathcal{G}$ fails and it is erased from $\mathcal{G}$. We display the estimation when the robots execute the proposed algorithm with $l = 100$ under the fixed graph in Fig. 3 without (black solid) and with (red solid) link failures. Here, although the variations in the graph topology take place very often (at each iteration), these variations are small. Therefore, as discussed in Section IV, the estimates of the proposed algorithm correctly track the average of the inputs (red solid). Obviously, this convergence is slower than for the fixed graph case (black solid).

The average execution times and messages sent per iteration and robot can be seen in Fig. 7 for $l = 100$. Immediately after each new step k the execution times are higher, since robots make additional memory space for the new variables that appear in their maps. After that, the execution times of the remaining iterations are much lower. As the size of the maps increase, times increase as well. The communication complexity increases with the size of the maps. Within a step k, the size of the messages is almost the same for all the



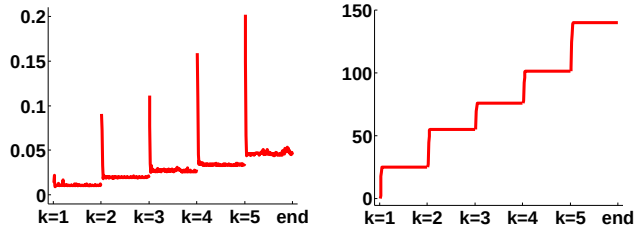
(a) $\max_{i,r} |[\mathbf{i}^k_{avg,i}(t)]_r - [\mathbf{i}^k_{avg}]_r|/\sigma_{\mathbf{i}}$ **versus** $f_k(t)\alpha_{\mathbf{i}}/\sigma_{\mathbf{i}} + g_k(t)$



(b) $\max_{i,r} |[\mathbf{i}^k_{avg,i}(t)]_r - [\mathbf{i}^k_{avg}]_r|/\sigma_{\mathbf{i}}$

Fig. 6. Estimation errors along the L consensus iterations for $l = 100$. (a) Comparison with a zero-initialization strategy and with the bounds for even iteration numbers t. The errors obtained (black solid) are always within the theoretical bounds (gray dashed) and they are smaller than the errors associated to the zero-initialization strategy (red solid). (b) Comparison with a switching graph. The robots execute Algorithm 1 using the communication graph $\mathcal{G}$ in Fig. 3 where, at each iteration t and step k, one of the links is selected randomly and erased from $\mathcal{G}$.

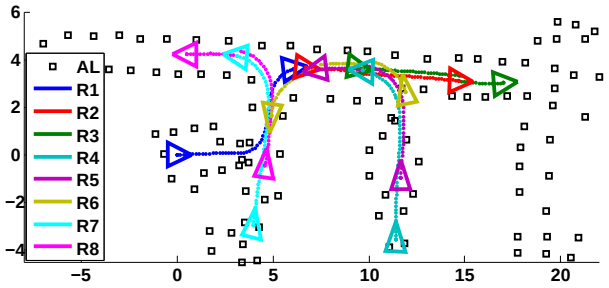
iterations of the algorithm. For the different configurations $l = 25, 50, 100$, the messages exchanged and execution times per robot and iteration, are almost the same.



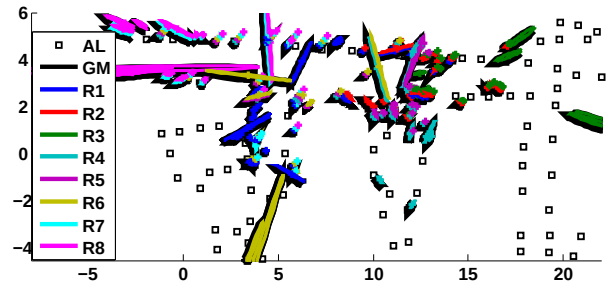
(a) Execution times (seconds) **(b) Messages (KBytes)**

Fig. 7. Execution times and messages sent per robot and iteration for $l = 100$. (a) Times with a laptop with i5-M460 (2.53GHz x 4 cores) processor, 4GB RAM, with MatLab 7.10 and Windows 7 as a background framework. (b) Messages with numbers encoded with single precision (4 bytes).

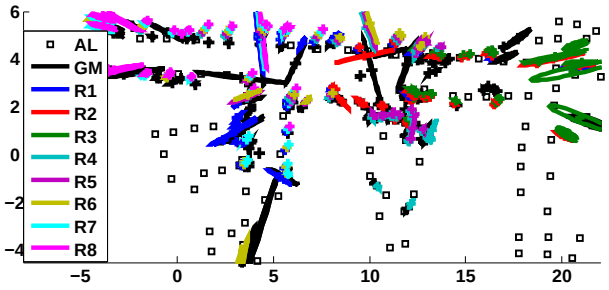
After the L iterations, the final global maps $\hat{\mathbf{x}}^k_{G,i}(t)$, $\Sigma^k_{G,i}(t)$, computed by the dynamic map merging algorithm are very close to the global map $\hat{\mathbf{x}}^k_G$, Σ^k_G in eq. (5) at step $k = K$. We show the global map at robot 1, for the $l = 100$ configuration (Fig. 8 (b)) after L iterations, which is very similar to the maps computed by the other robots (they are equal in the limit). Similar results have been obtained using the other configurations.



(a) Approximate trajectories followed by the 8 robots (triangles).



(b) Global map $\hat{\mathbf{x}}^k_{G,i}(t)$, $\Sigma^k_{G,i}(t)$, estimated by robot $i = 1$ at the last consensus iteration L, for the configuration $l = 100$.



(c) Centralized SLAM map vs. global map eq. (5).

Fig. 8. Robots follow the trajectories in **(a)**. Since there is no ground truth information available, a set of artificial landmarks (AL, black squares) are displayed to give an idea of the scene. Features belonging to the maps that were originally observed by different robots are displayed in different colors ($R1, \ldots, R8$) in **(b)** and **(c)** ; the global map $\hat{\mathbf{x}}^k_G$, Σ^k_G in eq. (5) is displayed with black crosses and lines (GM). **(b)** The global map estimated with our distributed method ($R1, \ldots, R8$) is very similar to the global map $\hat{\mathbf{x}}^k_G$, Σ^k_G in eq. (5) at step $k = K$ (GM). **(c)** The centralized SLAM map ($R1, \ldots, R8$) is similar to the global map $\hat{\mathbf{x}}^k_G$, Σ^k_G in eq. (5) at step $k = K$ (GM) as well.

The global map computed with our method (eq. (5)) has been compared to a centralized SLAM. We have executed a centralized multi-robot version of the SLAM algorithm used for building the individual maps, i.e., with features parameterized in inverse-depth, followed by a transform into cartesian before drawing. Although both are similar, (Fig. 8 (c)), the centralized map is more precise: the trace of the centralized covariance matrix is lower, the average covariance per feature is lower as well, and a high percent of the features (64.79 %) has been estimated with a higher precision (Table I). On the other hand, the centralized version maintains a single representation per feature and, as a result, for some features (35.21 %) it has not been able to use all the available robot observations, producing estimates less precise than for the distributed case.

Our distributed approach has a lower computational and

communication complexity than the centralized method. In Table I we display the size of the messages exchanged per robot and iteration, with number encoding and processor as in Fig. 7, for the configuration $l = 100$. The centralized SLAM makes robots propagate their observations to the central unit at each SLAM step using flooding (about 2 iterations in our experiments); then, the central node computes the centralized map and propagates it back to the robots. In our experiments, this process takes about 10+2+2 iterations for each step k. The complexities of both methods increase with the size of the scene, being the size of the messages exchanged per iteration and robot higher for the centralized SLAM, for all the steps $k = 1, \dots, K$ (Table I). Regarding the execution times, in the centralized case a single robot is responsible of the workload, whereas in the distributed case, the computations are shared by the robots. In Table I we show the total times for completing a step k, i.e., 10 SLAM iterations followed by $l = 100$ consensus iterations for the distributed case, versus 10 SLAM iterations of the centralized algorithm. Note that among the distributed cases ($l = 25, 50, 100$) tested, $l = 100$ is the one with the highest times. As we can see, even for this case, our times are lower than the centralized SLAM ones. Even if we sum up the execution times at all the robots, the obtained execution times are lower than for the centralized algorithm. This is due to the fact that the local map construction depends exclusively on the local features, and thus the times are much lower than for the centralized algorithm, whose complexity depends on the size of the scene.

TABLE I
COMPARISON BETWEEN OUR GLOBAL MAP AND THE CENTRALIZED ONE.

		Global map eq. (5)		**Centralized map**
Trace cov. mat.		19.42		4.80
Cov. per feat.		0.068		0.017
Feats. precise		35.21 %		64.79 %
Messages per robot and iter. (KBytes)	$k = 1$	24.70 KB		60.17 KB
	$k = 2$	54.35 KB		83.89 KB
	$k = 3$	75.55 KB		96.52 KB
	$k = 4$	100.61 KB		126.25 KB
	$k = 5$	138.81 KB		152.75 KB
		per robot	**sum robots**	
Exec. times per step k (seconds)	$k = 1$	1.55s	12.4s	13.0s
	$k = 2$	2.45s	19.6s	23.1s
	$k = 3$	3.25s	26.0s	37.7s
	$k = 4$	4.02s	32.2s	67.4s
	$k = 5$	5.65s	45.2s	125.9s

VII. CONCLUSIONS

We have presented an algorithm for dynamically merging visual maps in a robot network with limited communication. This algorithm allows the robots to have a better map of the environment containing the features observed by any other robot in the team. Thus, it helps the coordination of the team in several multi-robot tasks such as exploration or rescue. The algorithm correctly propagates the new information added by the robots to their local maps. We have shown that, with the proposed strategy, the robots correctly track the global map. At the final step, they obtain the last global map, which contains the last updated information at all the robots. In this paper we consider a fixed number of consensus iterations l per step. As future work we will analyze an adaptive algorithm where this number l dynamically changes for every step, depending on the problem requirements. The study of the robustness properties of the algorithm under link failures, changes of the topology, robots entering/leaving the network, is an interesting avenue of future research. Other extensions of this work are related to the improvement of the communication network usage. The number of consensus iterations may be optimized by a proper selection of the weights and μ in eq. (13) or by controlling the network topology to maximize its connectivity. The amount of information exchanged by the robots can be improved by applying submapping ideas or by sending only the most informative elements.

APPENDIX
CONSENSUS ALGORITHMS AND METROPOLIS WEIGHTS

The Proportional Integral (PI) consensus algorithm for systems with constant inputs presented in [18] is given by

$$\begin{bmatrix} \dot{\mathbf{x}}(t) \\ \dot{\mathbf{w}}(t) \end{bmatrix} = \begin{bmatrix} -\gamma \mathbf{I} - L_P & L_I^T \\ -L_I & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{x}(t) \\ \mathbf{w}(t) \end{bmatrix} + \begin{bmatrix} \gamma \mathbf{I} \\ \mathbf{0} \end{bmatrix} \mathbf{u}, \quad (54)$$

where $\mathbf{u} \in \mathbb{R}^n = (u_1, \dots, u_n)^T$, $\mathbf{x}(t) \in \mathbb{R}^n = (x_1(t), \dots, x_n(t))^T$ and $\mathbf{w}(t) \in \mathbb{R}^n = (w_1(t), \dots, w_n(t))^T$ are the inputs and variables at the n nodes. Note that in addition to the variable $x_i(t)$, each robot $i \in \mathcal{V}$ also maintains a second variable $w_i(t) \in \mathbb{R}$. L_P and L_I are Laplacian associated to respectively the proportional and the integral weight matrices. Their (i, j) entries associated to non neighbor robots $j \notin \mathcal{N}_i$ have zero value. And $\gamma > 0$ is a parameter that establishes the rate at which new information replaces old information. The PI algorithm is a continuous-time distributed averaging method where the state vector $\mathbf{x}(t)$ converges to the average of the inputs $\mathbf{1}\mathbf{1}^T\mathbf{u}/n$ asymptotically as $t \to \infty$ [18, Theorem 5].

A common choice for the weight matrices in distributed consensus are the Metropolis weights $W_M \in \mathbb{R}^{n \times n}$ introduced in [20] where for all $i, j \in \mathcal{V}$,

$$[W_M]_{ij} = \begin{cases} \frac{1}{1+\max\{|\mathcal{N}_i|, |\mathcal{N}_j|\}} & \text{if } j \in \mathcal{N}_i, \\ 0 & \text{if } j \notin \mathcal{N}_i \cup \{i\}, \\ 1 - \sum_{j' \in \mathcal{N}_i} [W_M]_{ij'} & \text{if } j = i, \end{cases} \quad (55)$$

being $|\mathcal{N}_i|$ and $|\mathcal{N}_j|$ the number of neighbors of nodes i, j. We let $L_M \in \mathbb{R}^{n \times n}$ be its associated Laplacian,

$$L_M = \mathrm{diag}(W_M \mathbf{1}) - W_M = \mathbf{I} - W_M. \quad (56)$$

Note that each agent can compute the weights that affect its evolution using only local information. The Metropolis weights associated to connected undirected communication graphs $\mathcal{G}$ have the following properties:

(i) W_M is symmetric, $W_M = W_M^T$, doubly stochastic, $W_M \mathbf{1} = \mathbf{1}$ and $\mathbf{1}^T W_M = \mathbf{1}^T$. It has a single eigenvalue at 1 and all its other eigenvalues $\lambda(W_M) \in (-1, 1)$;

(ii) L_M is symmetric, positive semidefinite [38, Theorem 1.37]. It has an eigenvalue at 0, and all the others $\lambda(L_M) \in (0, 2)$.

ACKNOWLEDGMENTS

The data set used in the experiments was provided by U. Frese and J. Kurlbaum. We thank the reviewers for their useful comments.

REFERENCES

[1] S. Roumeliotis and G. Bekey, "Distributed multirobot localization," *IEEE Transactions on Robotics and Automation*, vol. 18, no. 5, pp. 781–795, 2002.

[2] K. Y. K. Leung, T. D. Barfoot, and H. Liu, "Decentralized localization of sparsely-communicating robot networks: A centralized-equivalent approach," *IEEE Transactions on Robotics*, vol. 26, no. 1, pp. 62–77, 2010.

[3] A. Howard, "Multi-robot simultaneous localization and mapping using particle filters," *International Journal of Robotics Research*, vol. 25, no. 12, pp. 1243–1256, 2006.

[4] S. B. Williams and H. Durrant-Whyte, "Towards multi-vehicle simultaneous localisation and mapping," in *IEEE Int. Conf. on Robotics and Automation*, Washington, DC, USA, May 2002, pp. 2743–2748.

[5] R. Vincent, D. Fox, J. Ko, K. Konolige, B. Limketkai, B. Morisset, C. Ortiz, D. Schulz, and B. Stewart, "Distributed multirobot exploration, mapping, and task allocation," *Annals of Mathematics and Artificial Intelligence*, vol. 52, no. 1, pp. 229–255, 2008.

[6] L. M. Paz, J. D. Tardos, and J. Neira, "Divide and conquer: EKF SLAM in $o(n)$," *IEEE Transactions on Robotics*, vol. 24, no. 5, pp. 1107–1120, 2008.

[7] E. M. Nebot, M. Bozorg, and H. F. Durrant-Whyte, "Decentralized architecture for asynchronous sensors," *Autonomous Robots*, vol. 6, no. 2, pp. 147–164, 1999.

[8] S. Utete and H. F. Durrant-Whyte, "Routing for reliability in decentralised sensing networks," in *American Control Conference*, vol. 2, Jun. 1994, pp. 2268 – 2272.

[9] S. Grime and H. Durrant-Whyte, "Data fusion in decentralized sensor networks," *Control Engineering Practice*, vol. 2, no. 5, pp. 849 – 863, 1994.

[10] S. Julier and J. K. Uhlmann, "General decentralised data fusion with covariance intersection (CI)," in *Handbook of Multisensor Data Fusion*, D. L. Hall and J. Llinas, Eds. CRC Press, 2001.

[11] R. Olfati-Saber, "Distributed Kalman filter with embedded consensus filters," in *IEEE Conf. on Decision and Control*, Sevilla, Spain, 2005, pp. 8179–8184.

[12] ——, "Distributed Kalman filtering for sensor networks," in *IEEE Conf. on Decision and Control*, New Orleans, LA, Dec. 2007, pp. 5492–5498.

[13] P. Alriksson and A. Rantzer, "Distributed Kalman filtering using weighted averaging," in *Int. Symposium on Mathematical Theory of Networks and Systems*, Kyoto, Japan, Jul. 2006.

[14] D. W. Casbeer and R. Beard, "Distributed information filtering using consensus filters," in *American Control Conference*, St. Louis, USA, Jun. 2009, pp. 1882 –1887.

[15] R. Carli, A. Chiuso, L. Schenato, and S. Zampieri, "Distributed Kalman filtering based on consensus strategies," *IEEE Journal on Selected Areas in Communications*, vol. 26, no. 4, pp. 622–633, 2008.

[16] K. Y. K. Leung, T. D. Barfoot, and H. H. T. Liu, "Decentralized cooperative simultaneous localization and mapping for dynamic and sparse robot networks," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Taipei, Taiwan, Oct. 2010, pp. 3554 –3561.

[17] R. Aragues, E. Montijano, and C. Sagues, "Consistent data association in multi-robot systems with limited communications," in *Robotics: Science and Systems*, Zaragoza, Spain, Jun. 2010.

[18] R. A. Freeman, P. Yang, and K. M. Lynch, "Stability and convergence properties of dynamic average consensus estimators," in *IEEE Conf. on Decision and Control*, San Diego, CA, Dec. 2006, pp. 398–403.

[19] K. M. Lynch, I. B. Schwartz, P. Yang, and R. A. Freeman, "Decentralized environmental modeling by mobile sensor networks," *IEEE Transactions on Robotics*, vol. 24, no. 3, pp. 710–724, 2008.

[20] L. Xiao, S. Boyd, and S. Lall, "A scheme for robust distributed sensor fusion based on average consensus," in *Symposium on Information Processing of Sensor Networks (IPSN)*, Los Angeles, CA, Apr. 2005, pp. 63–70.

[21] ——, "A space-time diffusion scheme for peer-to-peer least-square estimation," in *Symposium on Information Processing of Sensor Networks (IPSN)*, Nashville, TN, Apr. 2006, pp. 168–176.

[22] G. Calafiore and F. Abrate, "Distributed linear estimation over sensor networks," *International Journal of Control*, vol. 82, no. 5, pp. 868–882, 2009.

[23] R. Aragues, J. Cortes, and C. Sagues, "Distributed consensus algorithms for merging feature-based maps with limited communication," *Robotics and Autonomous Systems*, vol. 59, no. 3-4, pp. 163–180, 2011.

[24] ——, "Dynamic consensus for merging visual maps under limited communications," in *IEEE Int. Conf. on Robotics and Automation*, Anchorage, AK, May 2010, pp. 3032–3037.

[25] S. Thrun, Y. Liu, D. Koller, A. Ng, and H. Durrant-Whyte, "Simultaneous localisation and mapping with sparse extended information filters," *International Journal of Robotics Research*, vol. 23, no. 7-8, pp. 693–716, 2004.

[26] W. Ren and R. W. Beard, *Distributed Consensus in Multi-vehicle Cooperative Control*, ser. Communications and Control Engineering. London: Springer Verlag, 2008.

[27] M. Franceschelli, A. Gasparri, A. Giua, and C. Seatzu, "Decentralized Laplacian eigenvalues estimation for networked multi-agent systems," in *IEEE Conf. on Decision and Control*, Shanghai, P. R. China, Dec. 2009, pp. 2717 –2722.

[28] M. C. D. Gennaro and A. Jadbabaie, "Decentralized control of connectivity for multi-agent systems," in *IEEE Conf. on Decision and Control*, San Diego, CA, Dec. 2006, pp. 3628 –3633.

[29] X. Zhou and S. Roumeliotis, "Robot-to-robot relative pose estimation from range measurements," *IEEE Transactions on Robotics*, vol. 24, no. 6, pp. 1379–1393, 2008.

[30] N. Trawny, X. S. Zhou, K. X. Zhou, and S. I. Roumeliotis, "Inter-robot transformations in 3-d," *IEEE Transactions on Robotics*, vol. 26, no. 2, pp. 226–243, 2010.

[31] J. J. Guerrero, A. C. Murillo, and C. Sagues, "Localization and matching using the planar trifocal tensor with bearing-only data," *IEEE Transactions on Robotics*, vol. 24, no. 2, pp. 494–501, 2008.

[32] M. Franceschelli and A. Gasparri, "On agreement problems with gossip algorithms in absence of common reference frames," in *IEEE Int. Conf. on Robotics and Automation*, Anchorage, USA, May 2010, pp. 4481–4486.

[33] G. Calafiore, L. Carlone, and M. Wei, "A distributed gradient method for localization of formations using relative range measurements," in *IEEE Multi-Conf. on Systems and Control*, Yokohama, Japan, Sep. 2010, pp. 1146 – 1151.

[34] B. D. O. Anderson, I. Shames, G. Mao, and B. Fidan, "Formal theory of noisy sensor network localization," *SIAM Journal on Discrete Mathematics*, vol. 24, no. 2, pp. 684–698, 2010.

[35] R. Aragues, L. Carlone, G. Calafiore, and C. Sagues, "Multi-agent localization from noisy relative pose measurements," in *IEEE Int. Conf. on Robotics and Automation*, Shanghai, China, May 2011, pp. 364–369.

[36] U. Frese and J. Kurlbaum, "A data set for data association," Jun. 2008, electronically available at http://www.sfbtr8.spatial-cognition.de/insidedataassociation/.

[37] J. M. M. Montiel, J. Civera, and J. Davison, "Unified inverse depth parametrization for monocular SLAM," in *Robotics: Science and Systems*, Philadelphia, USA, Aug. 2006.

[38] F. Bullo, J. Cortes, and S. Martinez, *Distributed Control of Robotic Networks*, ser. Applied Mathematics Series. Princeton University Press, 2009, electronically available at http://coordinationbook.info.