Graph-Based Distributed Cooperative Navigation for a General Multi-Robot Measurement Model

Vadim Indelman[*], Pini Gurfil[†], Ehud Rivlin[‡] and Hector Rotstein[§]

April 3, 2012

Abstract

Cooperative navigation (CN) enables a group of cooperative robots to reduce their individual navigation errors. For a general multi-robot (MR) measurement model that involves both inertial navigation data and other onboard sensor readings, taken at different time instances, the various sources of information become correlated. Thus, this correlation should be solved for in the process of information fusion to obtain consistent state estimation. The common approach for obtaining the correlation terms is to maintain an augmented covariance matrix. This method would work for relative pose measurements, but is impractical for a general MR measurement model, because the identities of the robots involved in generating the measurements, as well as the measurement time instances, are unknown a priori. In the current work, a new consistent information fusion method for a general MR measurement model is developed. The proposed approach relies on graph theory. It enables explicit on-demand calculation of the required correlation terms. The graph is locally maintained by every robot in the group, representing all the MR measurement updates. The developed method calculates the correlation terms in the most general scenarios of MR measurements while properly handling the involved process and measurement noise. A theoretical example and a statistical study are provided, demonstrating the performance of the method for vision-aided navigation based on a three-view measurement model. The method is compared, in a simulated environment, to a fixed-lag centralized smoothing approach. The method is also validated in an experiment that involved real imagery and navigation data. Computational complexity estimates show that the newly-developed method is computationally efficient.

1 Introduction

Autonomous navigation of a group of cooperative robots has attracted much attention in the recent decade. The ability of a group of cooperative robots to autonomously

[*] V. Indelman is with the College of Computing, Georgia Institute of Technology, Atlanta, GA 30332, USA (e-mail: indelman@cc.gatech.edu).

[†] P. Gurfil is with the Faculty of Aerospace Engineering, Technion - Israel Institute of Technology, Haifa 32000, Israel (e-mail: pgurfil@technion.ac.il).

[‡] E. Rivlin is with the Department of Computer Science, Technion - Israel Institute of Technology, Haifa 32000, Israel (e-mail: ehudr@cs.technion.ac.il).

[§] H. Rotstein is with RAFAEL - Advanced Defense Systems Limited, Israel (e-mail: hector@rafael.co.il).

carry out various tasks strongly depends on the navigation capabilities of each individual in the group. While each robot may be capable of calculating its own whereabouts, cooperative navigation is expected to be beneficial to all of the robots in the group. This is true particularly when operating in environments in which the global positioning system signals are unavailable or unreliable, such as indoors, underwater, or on other planets. In such cases, the robots must apply alternative techniques for correcting the evolving dead-reckoning or inertial navigation errors.

In cooperative navigation (CN), the robots provide navigation aids to each other during their respective missions, thereby allowing to correct their respective navigation solutions. Over the years, much attention has been devoted to distributed CN, in which each robot in the group has the same local strategy. Most of the proposed methods for CN rely on relative pose measurements between pairs of robots [Kurazume et al., 1994, Roumeliotis and Bekey, 2002, Fenwick et al., 2002, Smaili et al., 2008, Knuth and Barooah, 2009, Sharma and Taylor, 2009], allowing to perform a navigation update whenever one robot observes another robot. Another approach for CN is to identify a common scene observed by different robots, and to feed the resulting constraints as measurements to the navigation filter. Such an approach was recently suggested in [Merino et al., 2006, Kim et al., 2010], considering measurements that combine pairs of robots. In [Indelman et al., 2011], it was suggested to apply a vision-aided navigation technique based on three-view geometry [Indelman et al., 2012] to distributed CN. A measurement is formulated whenever the same scene is observed from three different views, which may be captured by different robots, possibly at different time instances.

Regardless of the method applied for CN, the navigation data involved in the measurement is obtained from different robots. In the general case, these sources of information may be statistically dependent. For instance, the navigation data of any two robots become correlated after the first update is carried out. Ignoring this correlation may result in inconsistent and over-confident estimations [Bahr et al., 2009].

Several approaches have been proposed for coping with the correlation terms in multi-robot (MR) systems, assuming relative pose measurements. In [Roumeliotis and Bekey, 2002], an augmented covariance matrix, composed of covariance and cross-covariance matrices relating all the robots in the group, was maintained in a distributed manner. In [Fenwick et al., 2002], this approach was applied to cooperative mapping and localization. In this case, the augmented covariance matrix also contains parameters that represent the landmarks observed by each robot in the group. Howard et al. [Howard et al., 2003] suggested a method that allows to avoid correlated updates in certain situations. Similarly, in [Bahr et al., 2009], the cross-covariance terms were not explicitly estimated. Instead, the authors proposed to maintain a bank of filters, tracking the origins of measurements and preventing a measurement to be used more than once. References [Mourikis et al., 2007] and [Lazaro and Castellanos, 2010] studied the filter inconsistency when correlated measurement sequences are used.

In this paper, we consider a *general* MR measurement model for CN. This model relates between the navigation data from any number of robots and the actual readings of the onboard sensors of these robots, which are *not* necessarily taken at the same time. In the general case, all the involved sources of information may be correlated. In addition to the a priori unknown identities of the robots that generate an MR measurement, our general MR model yields a manifold of a priori unknown parameters – the time instances that appear in the measurement equation. These additional unknown parameters render any method that is based on maintaining the correlation terms impractical.

An alternative approach to solve this problem is to avoid explicit calculation of the correlation terms by applying the covariance intersection (CI) method [Julier and Uhlmann, 1997], or its generalization [Xu and Negahdaripour, 2001]. The method of CI allows consistent fusion of different, possibly correlated, sources of information while the actual correlation is unknown. However, as reported in [Bahr et al., 2009, Arambel et al., 2001], CI is incapable of handling partial updates, i. e., cases in which the measurement matrix contains only a partial representation of the state vector. Thus, although CI was used in specific applications [Julier and Uhlmann, 2007, Lazarus et al., 2008], it cannot be applied to the general MR measurement model considered in this paper.

In this work, it is proposed to explicitly calculate the required correlation terms based on the history of all the thus-far performed MR measurements. As common in many CN methods, including [Roumeliotis and Bekey, 2002, Fenwick et al., 2002, Sharma and Taylor, 2009, Bahr et al., 2009], an extended Kalman filter is used for data fusion. Our method is capable of handling different MR measurement models regardless of the thus-far performed MR measurements. The developed method utilizes a graph representation of the history of all the executed MR measurement updates for calculating the correlation terms. This graph is maintained locally by every robot in the group, and hence the developed CN method is distributed.

The proposed approach is closely related to computing the inference in general probabilistic models represented by a graph structure, and in particular to belief propagation and loopy belief propagation algorithms [Jordan et al., 1999, Jordan, 2004, Malioutov et al., 2006]. Another related work is [Kim et al., 2010], that can be considered as a particular instantiation to CN of the inference computation based on graphical models. The authors of [Kim et al., 2010] consider relative pose and two-view measurements between pairs of robots and formulate an optimization problem that involves the history of the performed measurements between pairs of robots and measurements of the proprioceptive sensors of each robot. This problem is solved each time a new measurement of any kind is received, yielding an updated pose history of all the cooperative robots, which is equivalent to computing the inference of all the random variables represented in the graphical model [Jordan et al., 1999, Jordan, 2004]. In contrast to [Kim et al., 2010], in the current paper a general MR measurement model is used, and a method for *explicit* calculation of correlation terms, required in the update step of the fusion filter, is suggested. The newly-developed method allows navigation updates without applying smoothing over the past navigation history of the cooperative robots, and is therefore computationally efficient.

Consequently, the main contributions of this paper are twofold. First, a graph-based method for an explicit calculation of cross-covariance terms, required for consistent CN, is developed. The method assumes a general MR measurement model, relating any number of robots that may contribute information from different time instances. The identities of these robots and the time instances are a priori unknown. Second, the effect of process and measurement noise on the calculated cross covariances is analyzed and a method for incorporating these noise terms into the calculated cross-covariance terms is developed.

2 Problem Description

Consider a group of N cooperative robots capable of intercommunication. Each robot is equipped with inertial navigation sensors and hence is capable of calculating its own navigation solution, comprising position, velocity and angular orientation. Denote by $\mathbf{x}_i$ and $\mathbf{x}_i^t$ the calculated and the (unknown) true navigation solutions of the ith robot, respectively, and let $\mathbf{u}_i$ represent the measurements of the robot's inertial navigation sensors. The errors in $\mathbf{u}_i$ are modeled by an unknown vector of parameters $\boldsymbol{\alpha}_i^t$. Denote by $\boldsymbol{\alpha}$ the calculated model of inertial sensor errors, used for correcting the measurements $\mathbf{u}$. For instance, the vector $\boldsymbol{\alpha}$ includes a collection of accelerometer and gyro biases.

Let

$$\boldsymbol{\zeta}_i(t_k) \doteq \begin{bmatrix} \mathbf{x}_i(t_k) \\ \boldsymbol{\alpha}_i(t_k) \end{bmatrix} \quad , \quad \boldsymbol{\zeta}_i^t(t_k) \doteq \begin{bmatrix} \mathbf{x}_i^t(t_k) \\ \boldsymbol{\alpha}_i^t(t_k) \end{bmatrix} \tag{1}$$

and $\mathcal{N} \doteq \{1, \ldots, N\}$. Then

$$\boldsymbol{\zeta}_i(t_{k+1}) = \mathbf{f}(\boldsymbol{\zeta}_i(t_k), \mathbf{u}_i(t_k)) \quad , \quad i \in \mathcal{N} \tag{2}$$

The following navigation error state vector is defined

$$\mathbf{X}_i(t) \doteq \begin{bmatrix} \mathbf{x}_i(t) - \mathbf{x}_i^t(t) \\ \boldsymbol{\alpha}_i(t) - \boldsymbol{\alpha}_i^t(t) \end{bmatrix} \equiv \boldsymbol{\zeta}_i(t) - \boldsymbol{\zeta}_i^t(t) \tag{3}$$

It is well known (see, e. g., [Farrel and Barth, 1998]) that linearization of Eq. (2) yields the following linear time-varying stochastic model for the evolution of the state vector $\mathbf{X}_i$:

$$\dot{\mathbf{X}}_i(t) = \Phi^i(t)\mathbf{X}_i(t) + \boldsymbol{\omega}^i(t) \quad , \quad i \in \mathcal{N} \tag{4}$$

where Φ^i is the continuous system matrix and $\boldsymbol{\omega}^i$ is the process noise, which is assumed to be white and zero-mean Gaussian. This continuous time model can be replaced by a discrete model

$$\mathbf{X}_i(t_b) = \Phi^i_{t_a \to t_b}\mathbf{X}_i(t_a) + \boldsymbol{\omega}^i_{t_a \to t_b} \quad , \quad i \in \mathcal{N} \tag{5}$$

where $\Phi^i_{t_a \to t_b}$ is the discrete system matrix relating the state between any two time instances t_a and t_b, $t_b > t_a$, and $\boldsymbol{\omega}^i_{t_a \to t_b}$ is the equivalent discrete process noise.

In addition to the inertial sensors, each robot is equipped with its own set of onboard exogenous sensors[1]. The readings of the exogenous sensors of the jth robot at some time instant t_a are denoted by $\mathbf{y}_j(t_a)$. These measurements are corrupted by a Gaussian white noise $\mathbf{v}_j(t_a)$. Let $\mathbf{y}_j^t(t_a) \doteq \mathbf{y}_j(t_a) - \mathbf{v}_j(t_a)$.

Consider a general measurement model that relates the navigation data and onboard sensor measurements of several robots, possibly taken at different time instances. Let j denote the identities of the robots involved in this measurement model, $j \in \mathcal{N}$.

The considered measurement model can be formulated in an implicit form as

$$\mathbf{z}(t) = \mathbf{h}(\{\boldsymbol{\zeta}_j(t_i), \mathbf{y}_j(t_i)\}_{i=1}^r) \quad , \quad j \in \mathcal{N} \tag{6}$$

where $\mathbf{z}$ is the residual measurement, which is a function of $\boldsymbol{\zeta}_j(t_i)$, representing the navigation solution $\mathbf{x}_j(t_i)$ and parametrization of the inertial sensors errors $\boldsymbol{\alpha}_j(t_i)$, and the onboard sensor readings $\mathbf{y}_j(t_i)$ of the jth robot at time t_i, with $t_i \leq t$ and t being the current time. Note that any explicit measurement model can be expressed in an implicit form, while the opposite is incorrect. The parameter r in Eq. (6) denotes the overall number of information sets $(\boldsymbol{\zeta}_j(t_i), \mathbf{y}_j(t_i))$ constituting $\mathbf{z}$. If each of the participating robots contributes only a single information set, r represents the number of robots involved in the residual measurement $\mathbf{z}$. However, in the general case, each robot may contribute information from several time instances. For example, if some robot j contributes information

[1] In this paper, inertial sensors refer to the sensors used for dead reckoning or inertial navigation, while exogenous sensors are all the other sensors of the robot. For example, accelerometers and gyroscopes are inertial sensors, while an onboard camera is an exogenous sensor.

from two time instances $t_j^1 \doteq t_1$ and $t_j^2 \doteq t_2$, then $\mathbf{z}$ will be a function of $(\boldsymbol{\zeta}_j(t_j^1), \mathbf{y}_j(t_j^1))$ and $(\boldsymbol{\zeta}_j(t_j^2), \mathbf{y}_j(t_j^2))$.

To simplify the notation, it is assumed from this point onward that the identity of the robots forming $\mathbf{z}$ is given by $1, \ldots, r$; cases in which a robot contributes information from several time instances are treated as if this information was provided by different robots. Thus, the residual measurement $\mathbf{z}$ can be written as:

$$\mathbf{z}(t) = \mathbf{h}(\{\boldsymbol{\zeta}_i(t_i), \mathbf{y}_i(t_i)\}_{i=1}^r) \tag{7}$$

Linearizing Eq. (7) about $\boldsymbol{\zeta}_i^t(t_k)$ and $\mathbf{y}_i^t(t_i)$ gives

$$\mathbf{z}(t) \approx \sum_{i=1}^r H_i(t_i)\mathbf{X}_i(t_i) + D_i(t_i)\mathbf{v}_i(t_i) \tag{8}$$

where

$$H_i(t_i) = \nabla_{\boldsymbol{\zeta}_i^t(t_i)}\mathbf{h} \quad , \quad D_i(t_i) = \nabla_{\mathbf{y}_i^t(t_i)}\mathbf{h} \tag{9}$$

since $\boldsymbol{\zeta}_i^t(t_k)$ and $\mathbf{y}_i^t(t_i)$ are unknown, the Jacobian matrices are approximated by

$$H_i(t_i) = \nabla_{\boldsymbol{\zeta}_i(t_i)}\mathbf{h} \quad , \quad D_i(t_i) = \nabla_{\mathbf{y}_i(t_i)}\mathbf{h} \tag{10}$$

The update step of the Kalman filter involves cross-covariance terms relating the different state vectors that appear in the measurement model (8). Denoting by $\tilde{\mathbf{X}}$ the estimation error of $\mathbf{X}$, the required cross-covariance terms are $E[\tilde{\mathbf{X}}_i(t_i)\tilde{\mathbf{X}}_j^T(t_j)]$ with $i, j = 1 \ldots r, i \neq j$. If these terms are known, a consistent measurement update can be employed.

In the context of a general probabilistic approach, the terms $E[\tilde{\mathbf{X}}_i(t_i)\tilde{\mathbf{X}}_j^T(t_j)]$ are related to the joint probability density function (pdf) of $\boldsymbol{\zeta}_i(t_i)$ for all the robots in the group and all the time instances $t_i \in \{t_0, \cdots, t\}$, given the measurements $\mathbf{u_i}(t_i)$ and $\mathbf{y_i}(t_i)$ from all the robots. The reader is referred to [Kim et al., 2010] for a formal definition of the joint pdf for the case of two-robot measurements ($r = 2$).

The purpose of this paper is to present an efficient method to compute the cross-covariance matrices on-demand while the identity of the involved robots, i. e. the indices i and j, and the time instances t_i and t_j are unknown a priori. It is tempting to apply the common approach, used when considering relative pose measurements for CN [Roumeliotis and Bekey, 2002], wherein an augmented covariance matrix is maintained, consisting of the covariance matrices of all the robots in the group and of cross-covariance matrices relating any pair of robots. However, this approach can be only applied when the measurement model involves concurrent information from different robots, as indeed is the case with relative pose measurements.

In the case of a general measurement model (8), in addition to the a priori unknown identity of the r robots contributing to the multi-robot (MR) measurement, the involved time instances are also unknown a priori. Therefore, maintaining all the possible cross-covariance terms is not a practical solution in terms of both computational load and storage requirements. Instead, we suggest calculating the required cross-covariance terms *on-demand* for a general MR measurement model.

3 Concept of Explicit Cross-Covariance Calculation

Before presenting the general concept behind the proposed approach, it is convinient to illustrate the calculation of cross-covariance terms in a basic example. Throughout the paper the notations $\tilde{\mathbf{a}}^-$ and $\tilde{\mathbf{a}}^+$ are used for a priori and a posteriori estimation error of $\mathbf{a}$, respectively.

3.1 A Basic Example

In this example we consider a measurement composed of information obtained from three different robots, i. e. $r = 3$. The residual measurement $\mathbf{z}$ may therefore be written as

$$\mathbf{z} \approx H_3(t_3)\mathbf{X}_3(t_3) + H_2(t_2)\mathbf{X}_2(t_2) + H_1(t_1)\mathbf{X}_1(t_1) + D\mathbf{v} \tag{11}$$

with $D \doteq \begin{bmatrix} D_3(t_3) & D_2(t_2) & D_1(t_1) \end{bmatrix}$ and $\mathbf{v} \doteq \begin{bmatrix} \mathbf{v}_3^T(t_3) & \mathbf{v}_2^T(t_2) & \mathbf{v}_1^T(t_1) \end{bmatrix}^T$.

Figure 1 shows a scenario wherein information transmitted by Robots I and II, with the current information of Robot III, is used for updating Robot III. Circles denote a priori information, while squares denote update events. Two update events are shown in the figure. While a_1, a_2 and a_3 represent information used in the first update, b_1, b_2 and b_3 represent information used in the second update. Let t_{a_i} and t_{b_i} represent the time instances corresponding to a_i and b_i, respectively, with $i = 1, 2, 3$.

Assume that the first update was carried out and that the a priori covariance matrices of the 3 robots and all the cross-covariance matrices between these robots, at the time instances t_{a_1}, t_{a_2} and t_{a_3}, were stored. Assume also that the required information for the second update is available. The key question is how to calculate the cross-covariance terms required for computing the second update, i. e. $E[\tilde{\mathbf{X}}_{III}^-(t_{b_3})\tilde{\mathbf{X}}_{II}^-(t_{b_2})]$, $E[\tilde{\mathbf{X}}_{III}^-(t_{b_3})\tilde{\mathbf{X}}_{I}^-(t_{b_1})]$ and $E[\tilde{\mathbf{X}}_{II}^-(t_{b_2})\tilde{\mathbf{X}}_{I}^-(t_{b_1})]$.

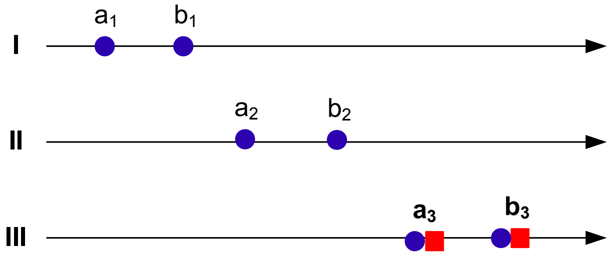


Figure 1: Measurement schedule example based on a measurement model that involves 3 robots. Robot III is updated based on information transmitted by Robots I and II. The circles denote information included in the measurement, squares indicate update events.

In particular, consider the calculation of $E[\tilde{\mathbf{X}}_{III}^-(t_{b_3})\tilde{\mathbf{X}}_{II}^-(t_{b_2})]$. Since no updates of any kind were performed between a_2 and b_2:

$$\tilde{\mathbf{X}}_{II}^-(t_{b_2}) = \Phi_{a_2 \to b_2}^{II}\tilde{\mathbf{X}}_{II}^-(t_{a_2}) + \boldsymbol{\omega}_{a_2 \to b_2}^{II} \tag{12}$$

In a similar manner, it is possible to write a transition relation between the a posteriori estimation error at a_3 and the a priori estimation error at b_3:

$$\tilde{\mathbf{X}}_{III}^-(t_{b_3}) = \Phi_{a_3 \to b_3}^{III}\tilde{\mathbf{X}}_{III}^+(t_{a_3}) + \boldsymbol{\omega}_{a_3 \to b_3}^{III} \tag{13}$$

Thus,

$$E[\tilde{\mathbf{X}}_{III}^-(t_{b_3})\tilde{\mathbf{X}}_{II}^-(t_{b_2})] = E\left[\left(\Phi_{a_3 \to b_3}^{III}\tilde{\mathbf{X}}_{III}^+(t_{a_3}) + \boldsymbol{\omega}_{a_3 \to b_3}^{III}\right)\left(\Phi_{a_2 \to b_2}^{II}\tilde{\mathbf{X}}_{II}^-(t_{a_2}) + \boldsymbol{\omega}_{a_2 \to b_2}^{II}\right)^T\right] \tag{14}$$

while the a posteriori estimation error at a_3 is given by

$$\begin{aligned} \tilde{\mathbf{X}}_{III}^+(t_{a_3}) &= (I - K_{a_3}H_{a_3})\tilde{\mathbf{X}}_{III}^-(t_{a_3}) - K_{a_3}H_{a_2}\tilde{\mathbf{X}}_{II}^-(t_{a_2}) \\ &- K_{a_3}H_{a_1}\tilde{\mathbf{X}}_{I}^-(t_{a_1}) - K_{a_3}D_a\mathbf{v}_a \end{aligned} \tag{15}$$

where K_{a_3} is the Kalman gain matrix, calculated by robot III at the first measurement update.

Since $\boldsymbol{\omega}^{II}_{a_2\to b_2}$ is statistically independent of $\tilde{\mathbf{X}}^-_{III}(t_{a_3}), \tilde{\mathbf{X}}^-_{II}(t_{a_2}), \tilde{\mathbf{X}}^-_{I}(t_{a_1})$, and since $\boldsymbol{\omega}^{III}_{a_3\to b_3}$ is statistically independent of $\tilde{\mathbf{X}}^-_{II}(t_{a_2})$ and $\boldsymbol{\omega}^{II}_{a_2\to b_2}$ (cf. Figure 1):

$$E\left[\tilde{\mathbf{X}}^+_{III}(t_{a_3})(\boldsymbol{\omega}^{II}_{a_2\to b_2})^T\right] = 0 \tag{16}$$

$$E\left[\boldsymbol{\omega}^{III}_{a_3\to b_3}\left(\Phi^{II}_{a_2\to b_2}\tilde{\mathbf{X}}^-_{II}(t_{a_2}) + \boldsymbol{\omega}^{II}_{a_2\to b_2}\right)^T\right] = 0 \tag{17}$$

In addition,

$$E\left[\mathbf{v}_a\left(\Phi^{II}_{a_2\to b_2}\tilde{\mathbf{X}}^-_{II}(t_{a_2}) + \boldsymbol{\omega}^{II}_{a_2\to b_2}\right)^T\right] = 0 \tag{18}$$

Let $\tilde{\mathbf{X}}^-_i(t_{a_i})$ be represented by $\tilde{\mathbf{X}}_{a_i}$ and denote $P_{ab} \doteq E[(\tilde{\mathbf{X}}_a)(\tilde{\mathbf{X}}_b)^T]$. Incorporating Eqs. (15)-(18) into Eq. (14) yields

$$P^-_{b_3b_2} = \Phi^{III}_{a_3\to b_3}\left\{(I - K_{a_3}H_{a_3})P^-_{a_3a_2} - K_{a_3}H_{a_2}P^-_{a_2a_2} - K_{a_3}H_{a_1}P^-_{a_1a_2}\right\}(\Phi^{II}_{a_2\to b_2})^T \tag{19}$$

Thus, $P^-_{b_3b_2}$ is expressed via the filter gain matrix, the measurement matrices, covariance and cross-covariance matrices from the past MR updates, which therefore need to be stored. The other two required cross-covariance terms in this example can be calculated using the same process, yielding an equivalent expression for $P^-_{b_3b_1}$, while $P^-_{b_2b_1} = 0$.

3.2 Overview of the Approach for a General Scenario

The approach discussed above can be generalized to any number of MR measurement updates based on the general measurement model formulated in Eq. (8).

The general cross-covariance term $E[\tilde{\mathbf{X}}_i(t_i)\tilde{\mathbf{X}}^T_j(t_j)]$ can be found by expressing each of the two state vectors $\tilde{\mathbf{X}}_i(t_i)$ and $\tilde{\mathbf{X}}_j(t_j)$ according to the history of the MR measurement updates, and then calculating $E[\tilde{\mathbf{X}}_i(t_i)\tilde{\mathbf{X}}^T_j(t_j)]$ based on the resulting expressions, while judiciously handling the involved noise terms. In contrast to the example from the previous section, in the general case the process and measurement noise terms are not necessarily statistically independent of the involved state vectors.

Clearly, sustaining the aforementioned approach requires storing the information involved in all the past MR measurement updates, including the filter gain, measurement, covariance and cross-covariance matrices. If this information is available for a *specific* sequence of MR measurement updates, the required cross-covariance terms can be calculated based on the process demonstrated in the previous section. In the following sections, however, a method for on-demand calculation of the cross-covariance terms for a *general* case is developed. The method uses a graph representation, which allows a systematical calculation of the required cross-covariance terms. The graph, locally maintained by every robot in the group, contains the information from all the past MR measurement updates.

The proposed graph topology relies upon a directed acyclic graph (DAG). Each time a new MR measurement is obtained, the graph is used for calculating the cross-covariance terms between all the r robots that participate in the measurement. These terms are then used for calculating the filter's gain and for updating the relevant robots (as explained below). Next, each robot updates its own copy of the graph with the executed measurement. The communication protocol and the actual transmitted information between the different robots in each MR update are discussed in Section 6.

While in theory every robot participating in a given MR measurement can be updated, in order to sustain the DAG assumption, only some of the robots are actually updated in the proposed approach. Before getting into further details (cf. Section 3.2.2 and Appendix A), it is important to understand the motivation for enforcing a DAG. The main reason is to avoid recursive updates, which will typically require an intensive smoothing process involving all the robots in the group [Kim et al., 2010] (and therefore extensive communication among the robots and high computational complexity). Another question that arises in such case is whether this process will eventually converge, since it is known that applying loopy belief propagation (LBP) for solving an inference problem in a graph with cycles, which is a related problem to the problem considered herein, is not guaranteed to converge [Malioutov et al., 2006]. Consequently, in this paper an acyclic graph is assumed and the identities of the updated robots are chosen so that the graph will remain as such.

At this point, it is useful to state the actual approximations of the proposed approach. Apart from the obvious approximations involved in the linearization of the process and measurement equations (Eqs. (5) and (8)), the only additional approximation is in the calculation of the Kalman gain, as discussed in Section 3.2.2 and Appendix A. The later becomes exact in scenarios in which all the r robots, involved in the MR measurement, can be updated while sustaining an acyclic graph. In all the other scenarios, updating all the r involved robots, while applying the method proposed herein for cross-covariance terms calculation, is not possible since the graph is no longer acyclic.

Before formally defining the graph structure and presenting the actual algorithm for cross-covariance calculation, we briefly overview the concept of the proposed approach and explain how to choose the robots to be updated.

3.2.1 Overview

In order to calculate a general cross-covariance term $E[\tilde{\mathbf{X}}_i(t_i)\tilde{\mathbf{X}}_j^T(t_j)]$, the nodes representing $\tilde{\mathbf{X}}_i(t_i)$ and $\tilde{\mathbf{X}}_j(t_j)$ are located in the graph. The next step is to identify all the possible paths in the graph that lead to these two nodes. This is performed by going over the graph and constructing a tree for each of these nodes; these trees contain all the relevant information in the graph that will be later used in calculation of $E[\tilde{\mathbf{X}}_i(t_i)\tilde{\mathbf{X}}_j^T(t_j)]$. The constructed two trees are actually inverse-trees, since each of them have only one leaf (representing $\tilde{\mathbf{X}}_i(t_i)$ or $\tilde{\mathbf{X}}_j(t_j)$) and possibly several root nodes. These trees are closely related to the computational trees used in the LBP algorithm [Malioutov et al., 2006].

Given these two trees, expressing $\tilde{\mathbf{X}}_i(t_i)$ and $\tilde{\mathbf{X}}_j(t_j)$ using the past MR measurements is equivalent to expanding the leaf node in each of the two trees by proceeding to connected nodes from upper levels[2]. Referring to the example from Section 3.1, the equivalent graph and the constructed two trees for $\tilde{\mathbf{X}}_i(t_i) \equiv \tilde{\mathbf{X}}_{III}^-(t_{b_3})$ and $\tilde{\mathbf{X}}_j(t_j) \equiv \tilde{\mathbf{X}}_{II}^-(t_{b_2})$ are shown in Figure 2. For instance, going up one level from the leaf nodes in each tree represents, up to the noise terms, Eqs. (12) and (13) (as explained in the sequel, the noise terms are represented in terms of noise covariance matrices).

As higher levels are processed, a check is performed whether the cross-covariance terms between the expressions obtained so far for $\tilde{\mathbf{X}}_i(t_i)$ and the expressions obtained so far for

[2] This step represents the triangulation algorithm in graphical models [Jordan et al., 1999, Jordan, 2004].

$\tilde{\mathbf{X}}_j(t_j)$ have been stored in the graph as the result of executing some MR measurement update in the past. Those cross-covariance terms that have indeed been stored in the graph, can be retrieved and used as part of the ingredients from which the required cross-covariance term $E[\tilde{\mathbf{X}}_i(t_i)\tilde{\mathbf{X}}_j^T(t_j)]$ will be eventually computed.

This brief sketch of the method's concept is further elaborated in Section 4.1, after the graph structure is formally defined.

3.2.2 Choosing What Robots to Update

Since only some of the r robots participating in the MR measurement (6) are updated, the actual filter equations need to be accordingly adjusted. Appendix A provides further details, including an approach for calculating the filter's gain matrix using the cross-covariance terms calculated by the method presented herein. In this section, we discuss what robots, among the r robots, can be updated.

Denote by t_i^{MR} the most recent time instant in which the ith robot was updated by any MR measurement. In a general MR system, the DAG topology is representative if each MR measurement is utilized for updating only the robots $i \in \{1,\ldots,r\}$, which contributed their navigation data from the time instant $t_i > t_i^{MR}$ and assuming these robots contributed a single information set $(\boldsymbol{\zeta}_i(t_i), \mathbf{y}_i(t_i))$ (cf. Section 2). In particular, the graph remains acyclic when only robots that contributed their current navigation information, i. e. $t_i = t$, are updated.

It is worth noting that if some robot i contributed $l > 1$ information sets $(\boldsymbol{\zeta}_i(t_i^1), \mathbf{y}_i(t_i^1)), (\boldsymbol{\zeta}_i(t_i^2), \mathbf{y}_i(t_i^2)), \ldots, (\boldsymbol{\zeta}_i(t_i^l), \mathbf{y}_i(t_i^l))$, with $t_i^1 < t_i^2 < \cdots < t_i^l$, to the MR measurement (6), this robot can be updated, while sustaining an acyclic graph, at the time instant t_i^l, provided that $t_i^l > t_i^{MR}$.

While in the discussion thus far several robots of the r robots were updated, for simplicity, throughout this paper we consider that only *one* robot is actually updated. Denoting by q the identity of the updated robot, its a posteriori estimation error in a general MR measurement model, formulated in Eq. (8), can be expressed as

$$\tilde{\mathbf{X}}_q^+(t_q) = (I - K_q H_q)\,\tilde{\mathbf{X}}_q^-(t_q) - K_q \sum_{i=1\ ,\ i\neq q}^{r} H_i \tilde{\mathbf{X}}_i^-(t_i) - K_q \sum_{i=1}^{r} D_i \mathbf{v}_i(t_i) \tag{20}$$

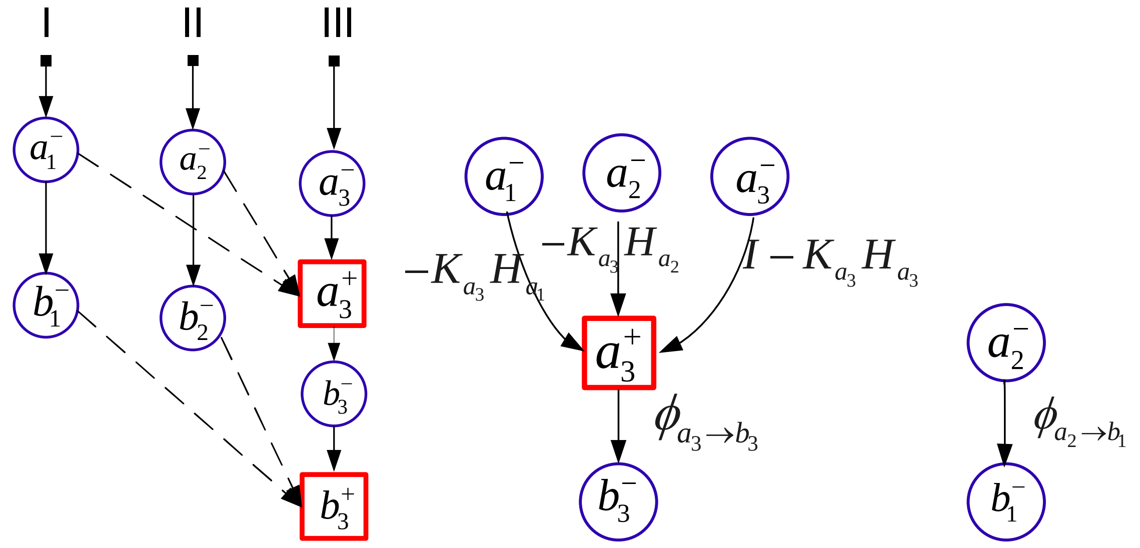
where K_q is the Kalman gain matrix computed for the qth robot. The a priori estimation error of some robot i, based on Eq. (5), is given by

$$\tilde{\mathbf{X}}_i^-(t_b) = \Phi^i_{t_a \to t_b} \tilde{\mathbf{X}}_i^-(t_a) + \boldsymbol{\omega}^i_{t_a \to t_b} \tag{21}$$

3.3 Graph Representation

Every robot in the group locally maintains its own copy of the DAG $G = (V, E)$, where V is the set of nodes and E is the set of directed weighted arcs. The weight of each arc reflects the information flow between the two connected nodes.

Two type of nodes exist in V. Nodes of the first type represent a priori information obtained from different robots in the group, constituting the MR measurements. These nodes are called *a priori nodes*. A single such node represents, therefore, $\boldsymbol{\zeta}_i(t_i)$ and $\mathbf{y}_i(t_i)$ – navigation data and readings of onboard sensors of the ith robot from time instant t_i, respectively. This information is transmitted by the ith robot to the updated robot q at the current time t (cf. Eq. (7)). In the general case, $t_i \leq t$. Nodes of the second type represent update events, i. e. the a posteriori information of the updated robot. Such nodes are called *a posteriori nodes*. Thus, each MR measurement update is represented by $r+1$ nodes. Figure 2(a) shows the graph obtained for the 3-robot measurement example considered in Section 3.1. A priori nodes are indicated in the graph by circles, while a posteriori nodes are designated by squares.



(a) (b)

Figure 2: (a) Graph representation for the scenario shown in Figure 1. (b) The trees $T_{b_3^-}$ and $T_{b_1^-}$ required for calculating $P^-_{b_3b_1}$.

We proceed by presenting the following definitions.

Definition 1. *A thread of the ith robot is a sub-graph of G, containing all the nodes in V that represent information of the ith robot and arcs in E connecting between these nodes.*

Each robot in the group has its own thread in G.

Definition 2. *The transition relation is given by*

$$\tilde{\mathbf{X}}_b = \Phi^i_{t_a\to t_b}\tilde{\mathbf{X}}_a + \boldsymbol{\omega}^i_{t_a\to t_b} \tag{22}$$

where $a, b \in V$ are any two adjacent a priori nodes in the ith thread, representing $\tilde{\mathbf{X}}_i^-(t_a)$ and $\tilde{\mathbf{X}}_i^-(t_b)$, respectively.

The transition relation connects between the a priori estimation errors of the ith robot at two different time instances t_a and t_b, as expressed by Eq. (21). The nodes a and b, both located in thread i, are connected by an arc, weighted by the transition matrix $w(a,b) = \Phi^i_{t_a\to t_b}$. The noise process covariance matrix $Q^i_{t_a\to t_b} \doteq E[\boldsymbol{\omega}^i_{t_a\to t_b}(\boldsymbol{\omega}^i_{t_a\to t_b})^T]$ is associated to this arc as well. For example, the nodes a_1^- and b_1^- in Figure 2(a) are connected by an arc representing a transition relation.

Each thread in G can also contain a posteriori nodes. In such a case, G will contain r a priori nodes that are connected to an a posteriori node, located in the thread of the updated robot q, by an update relation, defined as follows (cf. also Eq. (20)).

Definition 3. *Denote by α the a posteriori node, representing $\tilde{\mathbf{X}}_q^+(t_\alpha)$, and by β_i the a priori nodes, representing $\tilde{\mathbf{X}}_i^-(t_{\beta_i})$, with $i = 1, \ldots, r$. The update relation is given by:*

$$\tilde{\mathbf{X}}_\alpha = \left(I - K_\alpha H_{\beta_q}\right) \tilde{\mathbf{X}}_{\beta_q} - K_\alpha \sum_{i=1\ ,\ i \neq q}^{r} H_{\beta_i} \tilde{\mathbf{X}}_{\beta_i} - K_\alpha \sum_{i=1}^{r} D_{\beta_i} \mathbf{v}_{\beta_i} \tag{23}$$

where K_α is the Kalman gain computed by the updated robot.

The transition and update relations are illustrated in Figures 3(a) and 3(b), respectively.

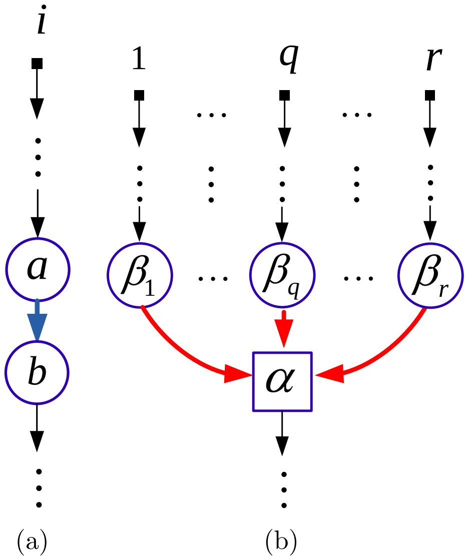


Figure 3: (a) The node a is connected to the node b via a transition relation. (b) The nodes β_i, with $i = 1, \ldots, r$, are connected to the node α via an update relation.

The arc weight $w(\beta_i, \alpha)$, connecting the a priori node β_i with the a posteriori node α is

$$w(\beta_i, \alpha) = \begin{cases} I - K_\alpha H_{\beta_q} & \text{if } i = q \\ -K_\alpha H_{\beta_i} & \text{else} \end{cases} \tag{24}$$

In addition, each arc is associated with a measurement noise covariance matrix $K_\alpha D_{\beta_i} R_{\beta_i} (K_\alpha D_{\beta_i})^T$, with $i = 1, \ldots, r$ and $R_{\beta_i} \doteq E[\mathbf{v}_{\beta_i} \mathbf{v}_{\beta_i}^T]$.

For instance, in Figure 2(a), the a priori information stored in the nodes a_1^-, a_2^- and a_3^- is connected to the node a_3^+ that represents a posteriori information.

As mentioned in Section 3.2, the a priori and a posteriori covariance and cross-covariance terms between the nodes, which participated in the *same* MR update in the past, are known (this information can be stored in the nodes themselves). The construction process of the graph and the communication protocol among the robots is discussed in [Indelman et al., 2011].

4 Graph-based Calculation of Cross-Covariance Terms

For a given DAG G, we wish to calculate $E[\tilde{\mathbf{X}}_i(t_i)\tilde{\mathbf{X}}_j^T(t_j)]$, the cross-covariance between the ith robot at t_i and the jth robot at t_j. In this section we use the notation $\tilde{\mathbf{X}}_a$ as an alternative to $\tilde{\mathbf{X}}_i^-(t_i)$ or $\tilde{\mathbf{X}}_i^+(t_i)$, where a is an a priori or a posteriori node in G, respectively. Let the nodes c and d in G represent $\tilde{\mathbf{X}}_i(t_i)$ and $\tilde{\mathbf{X}}_j(t_j)$, respectively. Thus, the goal here is to calculate $E[\tilde{\mathbf{X}}_c\tilde{\mathbf{X}}_d^T]$, which is equivalent to calculating $E[\tilde{\mathbf{X}}_i(t_i)\tilde{\mathbf{X}}_j^T(t_j)]$.

4.1 Rationale

As mentioned in Section 3.2.1, the first step is to construct two inverse-trees $T_c = (V_{T_c}, E_{T_c})$ and $T_d = (V_{T_d}, E_{T_d})$, containing all the possible paths in G to each of the nodes c and d. This can be performed as follows. The first tree, T_c, is initialized with the node c. Each next level is composed of the parents of the nodes that reside in the previous level, as determined from G. For example, the second level of T_c contains all the nodes in G that are directly connected to c. The same process is executed for constructing a tree T_d for the node d. Note that every node in T_c and T_d has only one child but may have one or r parents. In the latter case, the node represents an MR update event. Figure 2(b) shows an example of such trees, constructed based on the graph shown in Figure 2(a) for calculating the cross-covariance $E[\tilde{\mathbf{X}}_{b_3^-}\tilde{\mathbf{X}}_{b_1^-}^T]$, i. e. $c \equiv b_3^-$ and $d \equiv b_1^-$.

The convention used here is that if some node a_i has several parents, the jth parent is denoted as a_{i+1}^j. Also, $a \equiv a_1$, as shown in Figure 4.

Given the two trees T_c and T_d, the cross-covariance term $E[\tilde{\mathbf{X}}_c\tilde{\mathbf{X}}_d^T]$ can be computed by expressing $\tilde{\mathbf{X}}_c$ and $\tilde{\mathbf{X}}_d$ using information stored in the nodes from upper levels in the two trees. We start with the first level in the two trees, which is composed of the node c in T_c, and the node d in T_d. Since the cross-covariance $E[\tilde{\mathbf{X}}_c\tilde{\mathbf{X}}_d^T]$ is unknown, we proceed to the parents of these nodes, i. e. to the next level in the trees, according to the relation type represented by the arc weights.

Having reached the second level, the term $E[\tilde{\mathbf{X}}_c\tilde{\mathbf{X}}_d^T]$ can be expressed using information stored in nodes from the current (second) level and lower levels. For example, assuming a transition relation (22) connecting the first two levels in the two trees, $E[\tilde{\mathbf{X}}_c\tilde{\mathbf{X}}_d^T]$ can be written, according to Eq. (22), in three different forms:

$$\left\{\begin{array}{l} E\left[\tilde{\mathbf{X}}_c\left(\Phi_{d_2\to d}\tilde{\mathbf{X}}_{d_2} + \boldsymbol{\omega}_{d_2\to d}\right)^T\right] \\ E\left[\left(\Phi_{c_2\to c}\tilde{\mathbf{X}}_{c_2} + \boldsymbol{\omega}_{c_2\to c}\right)\tilde{\mathbf{X}}_d^T\right] \\ E\left[\left(\Phi_{c_2\to c}\tilde{\mathbf{X}}_{c_2} + \boldsymbol{\omega}_{c_2\to c}\right)\left(\Phi_{d_2\to d}\tilde{\mathbf{X}}_{d_2} + \boldsymbol{\omega}_{d_2\to d}\right)^T\right] \end{array}\right. \tag{25}$$

where c_2 and d_2 are the parents of c and d, respectively.

Since the expression from the previous (first) level was already checked, it is now required to examine whether any of the expressions involving nodes from the current level are known. In other words, the question is whether any of the pairs $E[\tilde{\mathbf{X}}_c\tilde{\mathbf{X}}_{d_2}^T]$, $E[\tilde{\mathbf{X}}_{c_2}\tilde{\mathbf{X}}_d^T]$ and $E[\tilde{\mathbf{X}}_{c_2}\tilde{\mathbf{X}}_{d_2}^T]$ are known. In addition, it is also required to know the correlation between the noise terms and the state vectors.

Since, in general, these pairs are unknown, we proceed to the next (third) level in the trees according to the relation type represented by the arc weights. Now, each of the expressions for $E[\tilde{\mathbf{X}}_c\tilde{\mathbf{X}}_d^T]$ obtained while processing the previous (second) level, may be further expanded using information stored in the nodes of the current (third) level.

Continuing the previous example, assume the second and third levels are connected by a transition relation (22) in T_c and an update relation (23) in T_d, and assume the third robot is updated ($q=3$). Then one of the possible expressions for $E[\tilde{\mathbf{X}}_c\tilde{\mathbf{X}}_d^T]$ would be obtained from $\tilde{\mathbf{X}}_c = \Phi_{c_2\to c}\tilde{\mathbf{X}}_{c_2} + \boldsymbol{\omega}_{c_2\to c}$ and

$$\tilde{\mathbf{X}}_d = \Phi_{d_2\to d}\left[\left(I - K_{d_2}H_{d_3^3}\right)\tilde{\mathbf{X}}_{d_3^3} - K_{d_2}\sum_{i=1\ ,\ i\neq 3}^{r} H_{d_3^i}\tilde{\mathbf{X}}_{d_3^i} - K_{d_2}\sum_{i=1}^{r} D_{d_3^i}\mathbf{v}_{d_3^i}\right] + \boldsymbol{\omega}_{d_2\to d} \quad (26)$$

Note that, compared to Eq. (23), $\alpha \equiv d_2$ and $\beta_i = d_3^i$.

Once again, the question is whether the different cross-covariance terms that appear in the new expressions involving current and lower levels are known (had been stored in G in the past). All the expressions from the previous level (the second level) were already analyzed. Ignoring for the moment terms that involve noise, it is obvious that less terms are to be analyzed when nodes closer to c or d are considered. Therefore, it is preferred to start analyzing from the lower level upward.

If, for example, $E[\tilde{\mathbf{X}}_{c_3}\tilde{\mathbf{X}}_{d_3^1}^T]$, is known, then the nodes $c_3 \in V_{T_c}$ and $d_3^1 \in V_{T_d}$ are either identical ($c_3 \equiv d_3^1$) or represent state vectors that had been used in the same MR measurement. Otherwise, $E[\tilde{\mathbf{X}}_{c_3}\tilde{\mathbf{X}}_{d_3^1}^T]$ would not have been stored in G. In any case, the known term $E[\tilde{\mathbf{X}}_{c_3}\tilde{\mathbf{X}}_{d_3^1}^T]$, properly weighted, is part of $E[\tilde{\mathbf{X}}_c\tilde{\mathbf{X}}_d^T]$. Having a known term also means that there is no need to proceed to nodes of higher levels related to this term.

The procedure continues to higher levels in the two trees until either all the terms required for calculating the cross-covariance $E[\tilde{\mathbf{X}}_c\tilde{\mathbf{X}}_d^T]$ are known, or the top level in both trees has been reached. In the latter case, the unknown terms of the cross-covariance are zero.

The process noise terms are assumed to be statistically independent, $E[\boldsymbol{\omega}_{i_1\to i_2}\boldsymbol{\omega}_{j_1\to j_2}^T] = 0$, if $\boldsymbol{\omega}_{i_1\to i_2}$ and $\boldsymbol{\omega}_{j_1\to j_2}$ belong to different robots, or, if $\boldsymbol{\omega}_{i_1\to i_2}$ and $\boldsymbol{\omega}_{j_1\to j_2}$ belong to the same robot at non-coinciding time instances, i. e., $(t_{i_1}, t_{i_2}) \cap (t_{j_1}, t_{j_2}) = \phi$. The measurement noise is assumed to be statistically independent of the state vectors involved in the measurement. On the other hand, the process and measurement noise terms may be statistically *dependent* on the involved state vectors (see Section 4.2.3).

In the following sections, the above rationale is transformed into an algorithm for calculating the cross-covariance $E[\tilde{\mathbf{X}}_c\tilde{\mathbf{X}}_d^T]$ in a *general* scenario.

4.2 Algorithm for Explicit Cross-Covariance Calculation

Let $T_b = (V_{T_b}, E_{T_b})$ be a tree containing all the paths in $G = (V, E)$ to some node $b \in V$, and let $a \in V_{T_b}$ and $\alpha, \beta \in V$. The following notations are used in the remainder of this paper:

$\pi_b(a)$	Parents of node a in tree T_b
$\mathcal{A}_b(a)$	Ancestors of node a in tree T_b
$\mathcal{D}_b(a)$	Descendants of node a in tree T_b
$a_k \overset{T_b}{\Longrightarrow} a$	Path $a_k \to \cdots \to a_2 \to a$ in tree T_b
$\{a_k \overset{T_b}{\Longrightarrow} a\}$	Group of nodes in the path $a_k \overset{T_b}{\Longrightarrow} a$

Definition 4. *A pair of nodes (α, β) is said to be known, if $E[\tilde{\mathbf{X}}_\alpha\tilde{\mathbf{X}}_\beta^T]$ is known, i. e., if it can be retrieved from the data stored in G. A known pair (α, β) is denoted by $\odot(\alpha, \beta)$.*

Definition 5. *Given the location of node a in the tree T_b, $(T_b)^a$ is defined as the sub-tree of T_b, containing all the ancestors of a in T_b and the node a itself.*

Let $T_c = (V_{T_c}, E_{T_c})$ and $T_d = (V_{T_d}, E_{T_d})$ be two trees constructed from G, and let $c_\delta, c_\rho \in V_{T_c}$ and $d_\eta, d_\zeta \in V_{T_d}$, where the indices $\delta, \rho, \eta, \zeta$ indicate the level in which each node is located.

Definition 6. *The pair (c_δ, d_η) is said to be younger than the pair (c_ρ, d_ζ) if*

$$\min(\delta, \eta) < \min(\rho, \zeta) \tag{27}$$

The algorithm for calculating cross covariance terms gradually processes pair permutations between nodes in $T_c = (V_{T_c}, E_{T_c})$ and nodes in $T_d = (V_{T_d}, E_{T_d})$ at different levels, starting from the first level. The permutation set of the kth level is denoted by $\mathcal{M}_k$, with $\mathcal{M}_1 \doteq \{(c, d)\}$. The next sections describe an algorithm for calculating $E[\tilde{\mathbf{X}}_c \tilde{\mathbf{X}}_d^T]$ based on $\mathcal{M}_k$ from different levels. The value of $E[\tilde{\mathbf{X}}_c \tilde{\mathbf{X}}_d^T]$ is initialized to zero.

4.2.1 Processing a single member of $\mathcal{M}_k$

In the general case, when processing the permutation set $\mathcal{M}_k$ from level k, all the nodes on the path to the leaf (which is $c \in V_{T_c}$ and $d \in V_{T_d}$) should be considered, starting from the leaf and going up until reaching the current level k. For example, assume that for some member $(c_k, d_k) \in \mathcal{M}_k$, the paths to the leaf nodes are $c_k \overset{T_c}{\Longrightarrow} c$ and $d_k \overset{T_d}{\Longrightarrow} d$. Figure 4(a) schematically illustrates a general path $c_k \overset{T_c}{\Longrightarrow} c$. Start by checking whether (c_k, d) or (c, d_k) are known in the sense of Definition 4, i. e., whether $\odot(c_k, d)$ or $\odot(c, d_k)$. If not, then check whether $\odot(c_k, d_2)$ or $\odot(c_2, d_k)$, and so on. The procedure ends when a known pair of nodes is found, or when reaching and analyzing the pair (c_k, d_k). When a known couple of nodes is discovered, its contribution to the cross-covariance $E[\tilde{\mathbf{X}}_c \tilde{\mathbf{X}}_d^T]$ is calculated.

Denote the overall weight of the paths $c_k \overset{T_c}{\Longrightarrow} c$ and $d_k \overset{T_d}{\Longrightarrow} d$ by $W_c(c_k)$ and $W_d(d_k)$, respectively. If $\odot(c_j, d_k)$, with $1 \leq j \leq k$, then $E[\tilde{\mathbf{X}}_c \tilde{\mathbf{X}}_d^T]$ is updated according to:

$$E[\tilde{\mathbf{X}}_c \tilde{\mathbf{X}}_d^T] \leftarrow E[\tilde{\mathbf{X}}_c \tilde{\mathbf{X}}_d^T] + W_c(c_j) E[\tilde{\mathbf{X}}_{c_j} \tilde{\mathbf{X}}_{d_k}^T] W_d^T(d_k) + Q_{c_j d_k} \tag{28}$$

Similarly, if $\odot(c_k, d_j)$, with $1 \leq j \leq k$, then $E[\tilde{\mathbf{X}}_c \tilde{\mathbf{X}}_d^T]$ is updated according to:

$$E[\tilde{\mathbf{X}}_c \tilde{\mathbf{X}}_d^T] \leftarrow E[\tilde{\mathbf{X}}_c \tilde{\mathbf{X}}_d^T] + W_c(c_k) E[\tilde{\mathbf{X}}_{c_k} \tilde{\mathbf{X}}_{d_j}^T] W_d^T(d_j) + Q_{c_k d_j} \tag{29}$$

The noise covariances $Q_{c_k d_j}$ and $Q_{c_j d_k}$ are analyzed in Section 4.2.3. If $w(a, b)$ is the arc weight connecting the node a to node b in G, then

$$\begin{aligned} W_c(c_k) &= \Pi_{i=2}^k w(c_i, c_{i-1}) && (30) \\ W_d(d_k) &= \Pi_{i=2}^k w(d_i, d_{i-1}) && (31) \end{aligned}$$

After finishing analyzing the member $(c_k, d_k) \in \mathcal{M}_k$, the permutation set $\mathcal{M}_k$ is updated as follows.

$$\mathcal{M}_k \leftarrow \mathcal{M}_k \diagdown \begin{cases} \{(c', d_k) \mid c' \in \pi_c(c_j)\ ,\ (c', d_k) \in \mathcal{M}_k\} & \text{if } \odot(c_j, d_k) \\ \{(c_k, d') \mid d' \in \pi_d(d_j)\ ,\ (c_k, d') \in \mathcal{M}_k\} & \text{if } \odot(c_k, d_j) \end{cases} \tag{32}$$

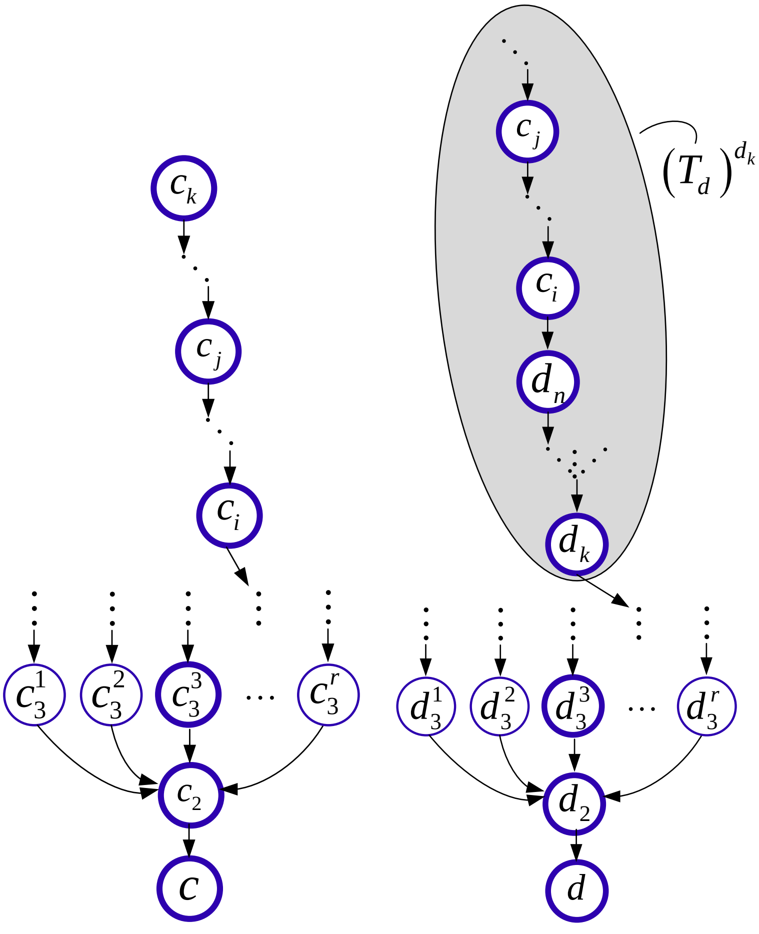
4.2.2 Calculation of $\mathcal{M}_{k+1}$

Having described how each level in the trees T_c and T_d is handled, the next step is to address the mechanism for advancing to the next level. After finishing processing all the members in $\mathcal{M}_k$, as discussed in Section 4.2.1, the only members left in $\mathcal{M}_k$ are those for whom the procedure did not find any known pair. If $\mathcal{M}_k = \phi$, the algorithm terminates.

The set of permutations in the next level, $\mathcal{M}_{k+1}$, is constructed based on the parents of each of the nodes that appear in $\mathcal{M}_k$: For each member $(a, b) \in \mathcal{M}_k$, the groups $\pi_c(a)$ and $\pi_d(b)$ are obtained. Then, a set of all the possible pair permutations between $\pi_c(a)$ and $\pi_d(b)$ is constructed and added to $\mathcal{M}_{k+1}$:

$$\mathcal{M}_{k+1} = \left\{(c^s_{k+1}, d^t_{k+1}) \mid c^s_{k+1} \in \pi_c(a)\ , d^t_{k+1} \in \pi_d(b)\ ,\ \forall (a, b) \in \mathcal{M}_k\right\} \tag{33}$$

where s and t distinguish between several parents.



(a) The tree T_c (b) The tree T_d

Figure 4: The node c_j in T_c has descendants that appear as ancestors of d_k in the sub-tree $(T_d)^{d_k}$, therefore contributing noise terms to the calculated $E[\tilde{\mathbf{X}}_c \tilde{\mathbf{X}}_d^T]$. Update-nodes are not explicitly marked.

4.2.3 Effect of Noise Terms

In this section, we discuss the effect of process and measurement noise terms on the cross-covariance $E[\tilde{\mathbf{X}}_c \tilde{\mathbf{X}}_d^T]$, when expressing $E[\tilde{\mathbf{X}}_c \tilde{\mathbf{X}}_d^T]$ via $\tilde{\mathbf{X}}_{c_k}$ and $\tilde{\mathbf{X}}_{d_k}$.

Let $T_a = (V_{T_a}, E_{T_a})$ be a tree constructed for some node $a \in V$, and let $a_l, a_{l-1} \in V_{T_a}$ be some nodes from levels l and $l-1$, respectively. These nodes are connected either by a transition relation (22) or an update relation (23). In the first case, the two nodes belong to the same thread, while in the second case, the nodes may be from different threads.

Denote by $\boldsymbol{\eta}_{a_l:a_{l-1}}$ the noise related to expressing $\tilde{\mathbf{X}}_{a_{l-1}}$ via $\tilde{\mathbf{X}}_{a_l}$. Then $\boldsymbol{\eta}_{a_l:a_{l-1}}$ can be either process or measurement noise, depending on the relation type:

$$\boldsymbol{\eta}_{a_l:a_{l-1}} = \left\{ \begin{array}{cl} \boldsymbol{\omega}_{a_l \to a_{l-1}} & \text{transition relation} \\ -K_{a_{l-1}} D_{a_l} \mathbf{v}_{a_l} & \text{update relation} \end{array} \right. \tag{34}$$

Let c_m and d_p be some nodes in the trees T_c and T_d, respectively, and recall Definition 5.

Lemma 1. *If $(T_d)^{d_p}$ does not contain any nodes from the path $c_m \to \cdots \to c_r \to \cdots \to c$ in T_c, then $\boldsymbol{\eta}_{c_\gamma:c_{\gamma-1}}$ and $\tilde{\mathbf{X}}_{d_p}$ are statistically independent for any $\gamma \in \{1, \ldots, m\}$.*

The proof of this and other Lemmas to follow, can be found in Appendix B.

Corollary 1. *If T_d does not contain any nodes from the path $c_m \xRightarrow{T_c} c$, then $\boldsymbol{\eta}_{c_\gamma:c_{\gamma-1}}$ and $\tilde{\mathbf{X}}_d$ are statistically independent for any $\gamma \in \{1, \ldots, m\}$.*

Lemma 1 and Corollary 1 are also valid, with the proper adjustments, when considering $(T_c)^{c_m}$ and T_c, respectively.

At this point assume, without loss of generality, that in the process of analyzing the member (c_k, d_k), described in Section 4.2.1, the pair (c_j, d_k) was discovered as known in the sense of Definition 4. Since nodes from lower levels are analyzed first, no other known pair (c_r, d_k) or (c_k, d_r) exists with $r < j$.

Lemma 2. *The path $d_k \xRightarrow{T_d} d$ does not contain any node c_r from the path $c_j \to \cdots \to c_r \to \cdots \to c$ in T_c for any $1 \leq r < j$. If $r = j$, the node $c_r = c_j$ can only appear in the path $d_k \xRightarrow{T_d} d$ as d_k.*

Lemmas 1 and 2 lead to the following corollary.

Corollary 2. *If $(T_d)^{d_k}$ does* not *contain any nodes from the path $c_j \xRightarrow{T_c} c$, then $\boldsymbol{\eta}_{c_\gamma:c_{\gamma-1}}$, for any $\gamma \in \{1, \ldots, m\}$, is statistically independent of all the states represented by the nodes $\{d_k \xRightarrow{T_d} d\} \cup (T_d)^{d_k}$.*

Note that $\boldsymbol{\eta}_{c_\gamma:c_{\gamma-1}}$ may still be statistically *dependent*, for at least a single value of $\gamma \in \{1, \ldots, m\}$, on states represented by the nodes in $T_d \diagdown \{d_k \xRightarrow{T_d} d\} \diagdown (T_d)^{d_k}$, if among these nodes there is at least one node from the path $c_j \xRightarrow{T_c} c$. This leads to the following corollary.

Corollary 3. *If for all the discovered pairs $\odot(a, b)$ with $a \in V_{T_c}$ and $b \in V_{T_d}$*

$$\mathcal{D}_c(a) \cap \mathcal{A}_d(b) = \phi \tag{35}$$

then all the noise terms from T_c, involved in the calculation of $E[\tilde{\mathbf{X}}_c \tilde{\mathbf{X}}_d^T]$, are statistically-independent of $\tilde{\mathbf{X}}_d$, and all the involved noise terms from T_d are statistically-independent of $\tilde{\mathbf{X}}_c$.

In other words, when the conditions of Corollary 3 are satisfied for all members in $\mathcal{M}_k$, for *all* considered k, the calculated cross-covariance $E[\tilde{\mathbf{X}}_c\tilde{\mathbf{X}}_d^T]$ will not contain any noise terms. However, when the conditions of Corollary 3 are not satisfied, $E[\tilde{\mathbf{X}}_c\tilde{\mathbf{X}}_d^T]$ will contain noise covariances from different time instances and robots. Returning to the discovered pair $\odot(c_j, d_k)$, we now assume that there are descendants of c_j in T_c that appear as ancestors of d_k in T_d: $\mathcal{D}_c(c_j) \cap \mathcal{A}_d(d_k) \neq \phi$. Consequently, $Q_{c_jd_k} \neq 0$ and, thus, the objective in the remainder of this section is to calculate $Q_{c_jd_k}$ (cf. Eq. (28)).

Among the nodes in $\mathcal{D}_c(c_j) \cap \mathcal{A}_d(d_k)$, denote by c_i, $1 < i < j$, the descendant of c_j that is closest to c, as illustrated in Figure 4. The child of c_i in T_d is denoted by d_n.

Lemma 3. *The path $c_j \overset{T_c}{\Longrightarrow} c_i$ appears in $(T_d)^{d_k}$.*

Observe that Lemma 3 is also valid for any sub-path $c_j \overset{T_c}{\Longrightarrow} c_{i'}$ of the path $c_j \overset{T_c}{\Longrightarrow} c_i$, with $i \leq i' < j$. Furthermore, $(T_d)^{d_k}$ might contain several appearances of the sub-paths $c_j \overset{T_c}{\Longrightarrow} c_{i'}$.

Now we analyze the correlation between the noise term $\boldsymbol{\eta}_{c_l:c_{l-1}}$, related to any two adjacent nodes c_l and c_{l-1} in the path $c_j \to \cdots \to c_l \to c_{l-1} \to \cdots \to c_i$, and $\tilde{\mathbf{X}}_{d_k}$. The term $E[\boldsymbol{\eta}_{c_l:c_{l-1}}\tilde{\mathbf{X}}_{d_k}^T]$, with $i+1 \leq l \leq j$, may be calculated as follows.

Assume for the moment that $(T_d)^{d_k}$ contains only a single appearance of $c_l \to c_{l-1}$. Then $\tilde{\mathbf{X}}_{d_k}$ is given by (cf. Figure 4)

$$\begin{aligned}\tilde{\mathbf{X}}_{d_k} &= W_{d_k}(c_l)\tilde{\mathbf{X}}_{c_l} + \sum_{r=k}^{n-1} W_{d_k}(d_r)\boldsymbol{\eta}_{d_{r+1}:d_r} + \\ &+ W_{d_k}(d_n)\boldsymbol{\eta}_{c_i:d_n} + \sum_{r=i}^{l-1} W_{d_k}(c_r)\boldsymbol{\eta}_{c_{r+1}:c_r} + \boldsymbol{\nu}_d \end{aligned} \tag{36}$$

where $\boldsymbol{\nu}_d$ is composed of state vectors and noise terms represented by nodes in $(T_d)^{d_k} \diagdown \{c_l \to c_{l-1} \to \cdots \to d_k\}$. Here, $W_{d_k}(a)$ is the overall weight of the path $a \to \ldots \to d_k$ in $(T_d)^{d_k}$. Since it was assumed that $c_l \to c_{l-1}$ appears only once in $(T_d)^{d_k}$, $(T_d)^{d_k} \diagdown \{c_l \to c_{l-1} \to \cdots \to d_k\}$ does not contain $c_l \to c_{l-1}$. Therefore, according to Lemma 1, $\boldsymbol{\eta}_{c_l:c_{l-1}}$ and $\boldsymbol{\nu}_d$ are statistically independent and thus, from Eq. (36),

$$E[\boldsymbol{\eta}_{c_l:c_{l-1}}\tilde{\mathbf{X}}_{d_k}^T] = E[\boldsymbol{\eta}_{c_l:c_{l-1}}\boldsymbol{\eta}_{c_l:c_{l-1}}^T]W_{d_k}^T(c_{l-1}) \tag{37}$$

The term $E[\boldsymbol{\eta}_{c_l:c_{l-1}}\boldsymbol{\eta}_{c_l:c_{l-1}}^T]$ is equal to the process or measurement noise covariances, depending on the relation type between $\tilde{\mathbf{X}}_{c_{l-1}}$ and $\tilde{\mathbf{X}}_{c_l}$ (cf. Eq. (34)):

$$E[\boldsymbol{\eta}_{c_l:c_{l-1}}\boldsymbol{\eta}_{c_l:c_{l-1}}^T] = \left\{ \begin{array}{rl} Q_{c_l:c_{l-1}} & \text{transition relation} \\ K_{c_{l-1}}D_{c_l}R_{c_l:c_{l-1}}(K_{c_{l-1}}D_{c_l})^T & \text{update relation} \end{array} \right. \tag{38}$$

Recall that the matrices in Eq. (38) were stored as part of the arc weights (cf. Section 3).

In the general case, $(T_d)^{d_k}$ may contain several appearances of $c_l \to c_{l-1}$, each appearance with its own path $c_l \to c_{l-1} \to \cdots \to d_k$. Letting u distinguish between these different appearances of $c_l \to c_{l-1}$ in $(T_d)^{d_k}$, and denoting by $W_b^u(a)$ the overall weight of the uth path $a \overset{T_b}{\Longrightarrow} b$, Eq. (37) becomes:

$$E[\boldsymbol{\eta}_{c_l:c_{l-1}}\tilde{\mathbf{X}}_{d_k}^T] = E[\boldsymbol{\eta}_{c_l:c_{l-1}}\boldsymbol{\eta}_{c_l:c_{l-1}}^T]\sum_u \left(W_{d_k}^u(c_{l-1})\right)^T \tag{39}$$

Furthermore, when considering the whole tree T_d, $c_l \rightarrow c_{l-1}$ may appear not only in $(T_d)^{d_k}$. According to Lemma 2, $c_l \rightarrow c_{l-1} \not\subset d_k \xrightarrow{T_d} d$. Thus, in addition to $(T_d)^{d_k}$, $c_l \rightarrow c_{l-1}$ may be also found only in $T_d \diagdown (T_d)^{d_k} \diagdown \{d_k \xrightarrow{T_d} d\}$. However, the contribution of the correlation between $\boldsymbol{\eta}_{c_l:c_{l-1}}$ and the state vectors represented by nodes in $T_d \diagdown (T_d)^{d_k} \diagdown \{d_k \xrightarrow{T_d} d\}$ will be calculated when processing other members in $\mathcal{M}_k$.
In a similar manner to Eq. (36), $\tilde{\mathbf{X}}_c$ can be expressed as (cf. Figure 4)

$$\tilde{\mathbf{X}}_c = W_c(c_j)\tilde{\mathbf{X}}_{c_j} + \sum_{r=i}^{l-1} W_c(c_r)\boldsymbol{\eta}_{c_{r+1}:c_r} + \boldsymbol{\nu}_c \tag{40}$$

where $\boldsymbol{\nu}_c$ is composed of state vectors and noise terms outside the path $c_j \xrightarrow{T_c} c_i \xrightarrow{T_c} c$. Therefore, the contribution of the noise term $\boldsymbol{\eta}_{c_l:c_{l-1}}$ to $E[\tilde{\mathbf{X}}_c \tilde{\mathbf{X}}_d^T]$, due to the nodes in $\mathcal{D}_c(c_j) \cap \mathcal{A}_d(d_k)$, is:

$$\bar{Q}_1(l) \doteq W_c(c_{l-1}) E[\boldsymbol{\eta}_{c_l:c_{l-1}} \boldsymbol{\eta}_{c_l:c_{l-1}}^T] \sum_u (W_d^u(c_{l-1}))^T \tag{41}$$

for *each* $i+1 \leq l \leq j$.

Yet, in addition to the above, the nodes $\mathcal{D}_d(d_k) \cap \mathcal{A}_c(c_j)$ also appear in expressions that constitute $Q_{c_j d_k}$. This situation may be handled in a similar manner. Among all the nodes in $\mathcal{D}_d(d_k) \cap \mathcal{A}_c(c_j)$, denote by d_s, $1 < s < k$, the node that is closest to d. Thus, the contribution of noise terms to $E[\tilde{\mathbf{X}}_c \tilde{\mathbf{X}}_d^T]$, due to the nodes $\mathcal{D}_d(d_k) \cap \mathcal{A}_c(c_j)$, is:

$$\bar{Q}_2(m) \doteq \sum_u W_c^u(d_{m-1}) E[\boldsymbol{\eta}_{d_m:d_{m-1}} \boldsymbol{\eta}_{d_m:d_{m-1}}^T] W_d^T(d_{m-1}) \tag{42}$$

for *each* $s+1 \leq m \leq k$.

In conclusion, the noise covariance $Q_{c_j d_k}$ for a discovered $\odot(c_j, d_k)$ is:

$$Q_{c_j d_k} \doteq \sum_{l=i+1}^{j} \bar{Q}_1(l) + \sum_{m=s+1}^{k} \bar{Q}_2(m) \tag{43}$$

In practice, the calculation of $Q_{c_j d_k}$ requires processing all the nodes in $(T_d)^{d_k}$, checking if they appear in $c_j \xrightarrow{T_c} c$, and processing all the nodes in $(T_c)^{c_j}$, checking if these nodes appear in $d_k \xrightarrow{T_d} d$. If such nodes were found, the contribution of the involved noise terms is computed using Eq. (43). A similar process should be carried out for calculating $Q_{c_k d_j}$ in case $\odot(c_k, d_j)$ is discovered (cf. Eq. (29)).

The above calculations are required only upon discovering a known pair. A formal algorithm for calculating $Q_{c^* d^*}$ for some discovered pair $\odot(c^*, d^*)$ is given in the next section.

4.3 Formal Algorithms

Algorithm 1 summarizes the developed approach for calculating the cross covariance $E[\tilde{\mathbf{X}}_c \tilde{\mathbf{X}}_d^T]$ given the trees T_c and T_d. The notation card(A) denotes the cardinality of the set A.

The process of analyzing a single permutation (c_k, d_k) from $\mathcal{M}_k$, discussed in Section 4.2.1, is presented in Algorithm 2, while Algorithm 3 implements the technique, developed in Section 4.2.3, for calculating the effect of the noise terms on the calculated cross covariance $E[\tilde{\mathbf{X}}_c \tilde{\mathbf{X}}_d^T]$.

Algorithm 1 Calculation of $E[(\tilde{\mathbf{X}}_c)(\tilde{\mathbf{X}}_d)^T]$

```
**Input**: Trees $T_c, T_d$. $h_c \doteq \text{height}(T_c), h_d \doteq \text{height}(T_d)$
**Initialization**: $k = 1$, $E[(\tilde{\mathbf{X}}_c)(\tilde{\mathbf{X}}_d)^T] = 0$, $\mathcal{M}_1 = \{(c, d)\}$.
**while** $k \leq \max(h_c, h_d)$ **do**
    **for** $r = 1$ to $\text{card}(\mathcal{M}_k)$ **do**
        Let $(c_k, d_k) \doteq \mathcal{M}(r)$. Execute Algorithm 2 on $(c_k, d_k)$. Let the output be
        $c^*, d^*, W_{c^*d^*}$, flag.
        **if** flag **then**
            $E[(\tilde{\mathbf{X}}_c)(\tilde{\mathbf{X}}_d)^T] = E[(\tilde{\mathbf{X}}_c)(\tilde{\mathbf{X}}_d)^T] + W_{c^*d^*}$
            Update $\mathcal{M}_k$ according to Eq. (32)
        **end if**
    **end for**
    **if** $\mathcal{M}_k$ is empty **then**
        **return** $E[(\tilde{\mathbf{X}}_c)(\tilde{\mathbf{X}}_d)^T]$
    **else**
        Construct $\mathcal{M}_{k+1}$ based on Eq. (33)
        $k = k + 1$
    **end if**
**end while**
**return** $E[(\tilde{\mathbf{X}}_c)(\tilde{\mathbf{X}}_d)^T]$
```

Algorithm 2 Processing a single member (c_k, d_k) from $\mathcal{M}_k$

```
**Input**: Trees $T_c, T_d$, node $c_k$ in $T_c$ and node $d_k$ in $T_d$
**Initialization**: $l = 1$, $c^* = d^* = W_{c^*d^*} = \{\}$, flag $= 0$
**while** $l \leq k$ **do**
    **if** $E[\tilde{\mathbf{X}}_{c_k}\tilde{\mathbf{X}}_{d_l}^T]$ is known, i. e., $\odot(c_k, d_l)$ **then**
        $c^* \doteq c_k$, $d^* \doteq d_l$, flag $= 1$
        break
    **end if**
    **if** $E[\tilde{\mathbf{X}}_{c_l}\tilde{\mathbf{X}}_{d_k}^T]$ is known, i. e., $\odot(c_l, d_k)$ **then**
        $c^* \doteq c_l$, $d^* \doteq d_k$, flag $= 1$
        break
    **end if**
    $l = l + 1$
**end while**
**if** flag **then**
    Calculate $Q_{c^*d^*}$ by executing Algorithm 3
    $W_{c^*d^*} = W_c(c^*)E[\tilde{\mathbf{X}}_{c^*}\tilde{\mathbf{X}}_{d^*}^T]W_d^T(d^*) + Q_{c^*d^*}$
**end if**
**return** $c^*, d^*, W_{c^*d^*}$, flag
```

Algorithm 3 Calculation of $Q_{c^*d^*}$.

```
1:  Input: $T_c, T_d, c^*, d^*$, s.t. $E[\tilde{\mathbf{X}}_{c^*}\tilde{\mathbf{X}}_{d^*}^T]$ is known.
2:  Initialization: $U_{d^*} \doteq (T_d)^{d^*}$, $U_{c^*} \doteq (T_c)^{c^*}$, $Q_{c^*d^*} = 0$.
3:  while $U_{d^*}$ is not empty do
4:      $U_{d^*} = U_{d^*} \diagdown \{l_i\}$, where $\{l_i\}$ are the leafs of $U_{d^*}$.
5:      Check if any leafs of $U_{d^*}$ appear in $c^* \xRightarrow{T_c} c$.
6:      for each such leaf $\beta$ of $U_{d^*}$ do
7:          Denote $c^* \to \cdots \to \beta$ as $u_s \to \cdots \to u_1$, then $E[\boldsymbol{\eta}_{c^*\to\beta}\boldsymbol{\eta}_{c^*\to\beta}^T] = \sum_{\zeta=2}^{s} W_\beta(u_{\zeta-1})E[\boldsymbol{\eta}_{\zeta\to\zeta-1}\boldsymbol{\eta}_{\zeta\to\zeta-1}^T]\left(W_\beta(u_{\zeta-1})\right)^T$
8:          $Q_{c^*d^*} = Q_{c^*d^*} + W_c(\beta)E[\boldsymbol{\eta}_{c^*\to\beta}\boldsymbol{\eta}_{c^*\to\beta}^T]W_d^T(\beta)$
9:          $U_{d^*} = U_{d^*} \diagdown (T_d)^\beta$
10:     end for
11: end while
12: Repeat Steps 3-11, replacing: $U_{d^*}$ by $U_{c^*}$; $c^*$ by $d^*$; $c$ by $d$; $T_c$ by $T_d$; instead of Step 8 perform $Q_{c^*d^*} = Q_{c^*d^*} + W_c(\beta)E[\boldsymbol{\eta}_{d^*\to\beta}\boldsymbol{\eta}_{d^*\to\beta}^T]W_d^T(\beta)$.
13: return $Q_{c^*d^*}$
```

5 Computational Complexity

As seen in Section 3, the computational complexity depends on the particular scenario being considered. In this section, an analysis of the computational complexity is provided. It is shown that the worst-case computational complexity is bounded by $O\left(n^2 log(rn)\right)$, where n is the number of the performed MR measurement updates, represented in G. Section 5.2 suggests an efficient implementation method, which allows to considerably reduce the actual computational complexity.

If a robot has limited computational resources, it is possible to approximate the true cross-covariance terms by maintaining a limited history of the MR measurement updates. In this case, the graph G may be treated as a constant-size buffer, where upon reaching a maximum size, the nodes representing information contained in old MR measurement updates[3] are removed from the graph G, thereby neglecting the contribution of those updates on the cross-covariance terms to be computed in the future.

5.1 Computational Complexity Analysis

Assume that $n-1$ MR update events have been carried out and currently the nth update event should be performed. Since each MR measurement is represented in the graph G by $r+1$ nodes, prior to the nth update event the graph G will contain $(r+1)(n-1) = (r+1)n - r - 1$ nodes. These nodes are scattered among the robot threads in G. Since each node in the two trees may have one or r parents, the number of nodes in the ith level is bounded by r^{i-1}.

A tighter bound can be obtained by noting that at least one level should separate between two update-event nodes. Therefore, if a node has r parents, each of these parents nodes will have only one parent. Consequently, the number of nodes in the ith level is bounded by $r^{\lfloor 0.5(i-1)\rfloor}$.

The analyzed worst case is composed of the following assumptions: (i) The number

[3] Different logic may be applied for choosing the nodes to be removed from the graph.

of nodes in each level i in the two trees is r^{i-1}; (ii) known pairs of nodes, in the sense of Definition 4, are found only upon reaching the top level in both trees, thereby ensuring maximum-size permutation sets $\mathcal{M}_k$ and that all the levels are processed by Algorithm 1; (iii) the computational cost of checking whether $\odot(a, b)$, i. e. whether $E[\tilde{\mathbf{X}}_a\tilde{\mathbf{X}}_b^T]$ is known, is $O(1)$.

Following these assumptions, the height h of each of the two trees can be calculated from

$$(r+1)n - r - 1 = \sum_{i=1}^{h} r^{i-1} = r^h - 1 \tag{44}$$

which implies

$$h \approx log_r(rn+n) \tag{45}$$

In addition,

$$\text{card}(\mathcal{M}_k) \leq r^{2(k-1)} \tag{46}$$

The complexity of processing a single member from $\mathcal{M}_k$ is bounded by $2i$. Thus, without taking into account the involved complexity of Algorithm 3 for calculating the contribution of noise terms, the overall computational complexity is bounded by

$$\sum_{i=1}^{h} r^{2(i-1)} \cdot 2i = r^{-2} \sum_{i=1}^{h} r^{2i} \cdot 2i \tag{47}$$

Letting $j \doteq 2i$,

$$r^{-2} \sum_{i=1}^{h} r^{2i} \cdot 2i = r^{-2} \sum_{j=1}^{2h} jr^j - r^{-1} \tag{48}$$

Now, using the relation

$$\sum_{i=1}^{m} ir^i = \frac{r}{(r-1)^2}\left(1 - r^m - mr^m + mr^{1+m}\right) \approx \frac{1}{r}\left[mr^m(r-1) - (r^m - 1)\right] \tag{49}$$

and recalling that $h = log_r(rn+n)$ gives

$$\begin{aligned} r^{-2} \sum_{j=1}^{2h} jr^j - r^{-1} &< r^{-2} \sum_{j=1}^{2h} jr^j \\ &= r^{-3}\left[log_r(rn+n)^2 \cdot (rn+n)^2(r-1) - ((rn+n)^2 - 1)\right] \\ &\approx r^{-3}(r+1)^2(r-1)n^2 log_r(rn+n)^2 \sim O\left(n^2 log(rn)\right) \end{aligned} \tag{50}$$

The computational complexity cost of calculating the contribution of the noise terms to the cross-covariance (Algorithm 3) can be bounded as follows. It is assumed that Algorithm 3 is carried out each time a pair from $\mathcal{M}_k$ is processed. Note that in practice, Algorithm 3 should be executed only upon finding a known pair of nodes. A single execution of this algorithm for a pair of nodes (c_i, d_i) from the ith level requires checking for each $a \in \mathcal{D}_c(c_i)$ whether $a \in \mathcal{A}_d(d_i)$, and for each $b \in \mathcal{D}_d(d_i)$ whether $b \in \mathcal{D}_c(c_i)$. This procedure is therefore bounded by $2ir^{h-i}$. Thus, processing a single member from

$\mathcal{M}_i$ is now bounded by $2i + 2ir^{h-i}$ instead of $2i$. The overall computational complexity, including the complexity of Algorithm 3, is therefore bounded by

$$\sum_{i=1}^{h} r^{2(i-1)} \cdot (2i + 2ir^{h-i}) \sim O\left(n^2 log(rn)\right) \tag{51}$$

In conclusion, the worst-case complexity of calculating a cross-covariance term in a general scenario is bounded by $O\left(n^2 log(rn)\right)$.

5.2 Efficient Implementation

The computational load can be significantly reduced by efficient implementation methods. One possible implementation is described next.

A meta-structure $\mathcal{H}$ is created and maintained when constructing the two trees T_c and T_d. This structure is composed of a header containing the details of all the nodes participating in either of the two trees. Each cell in the header, representing some node b, has also a flag indicating whether b appears in both of the trees. In addition, each cell points towards a structure that contains the following fields: The name of the tree in which b appears; height of the node b; link to the location of b in the tree. The structure contains also pointers to nodes $u^1, \ldots u^{r-1}$, if such nodes exist, such that b and the nodes $u^1, \ldots u^{r-1}$ belong to the same MR measurement update (and therefore, $E[\tilde{\mathbf{X}}_b \tilde{\mathbf{X}}_{u^i}^T]$, $i = 1, \ldots, r-1$, are known). If b appears in the trees several times, a linked list is used in which each cell is a structure representing a single appearance of b in the trees. Figure 5 shows schematically such a structure for $r = 3$.

This implementation allows processing each member $(c_k, d_k) \in \mathcal{M}_k$ more efficiently, although the worst-case computational complexity does not change. Instead of looking for $\odot(c_k, d_j)$ or $\odot(c_j, d_k)$, by going over the nodes in $c_k \overset{T_c}{\Longrightarrow} c$ and $d_k \overset{T_d}{\Longrightarrow} d$, the following may be performed: Check in the meta-structure $\mathcal{H}$ whether c_k is linked to any other nodes, which were part of the same MR update. For each such node u (there are only $r-1$ such nodes), check if $u \in V_{T_d}$ by going over the linked list of u in $\mathcal{H}$. For each appearance $u \in V_{T_d}$, check if $h_d(u) < k$, and then check if $d_k \in \mathcal{A}_d(u)$. Choose the node u with the smallest height. Repeat the process for d_k with the proper adjustments.

Assume that $\odot(c_j, d_k)$. When computing the contribution of the noise terms (cf. Section 4.2.3), instead of processing all the nodes[4] in $(T_d)^{d_k}$ and $(T_c)^{c_j}$, checking whether they appear in $c_j \overset{T_c}{\Longrightarrow} c$ and $d_k \overset{T_d}{\Longrightarrow} d$, respectively, the following may be performed. For each node $c_r \in c_j \overset{T_c}{\Longrightarrow} c$, check in $\mathcal{H}$ whether it appears in T_d (indicated by flag = 1). If it does, go over the linked list of c_r in $\mathcal{H}$ and choose only the appearances of c_r in T_d which are higher than k. For each chosen appearance of c_r, verify that d_k is a descendant. Repeat the process for $d_k \overset{T_d}{\Longrightarrow} d$ (with respect to T_c).

6 Communication Protocol and Communication Cost

This section describes the communication protocol required for carrying out a single MR measurement update, given a graph G representing all the MR measurements executed in the past. An upper bound on the communication cost is given as well. It is assumed that

[4] The number of nodes in $(T_a)^b$ is bounded by r^{h-h_b}, where h is the height of the tree T_a, and h_b is the height of node b in T_a.

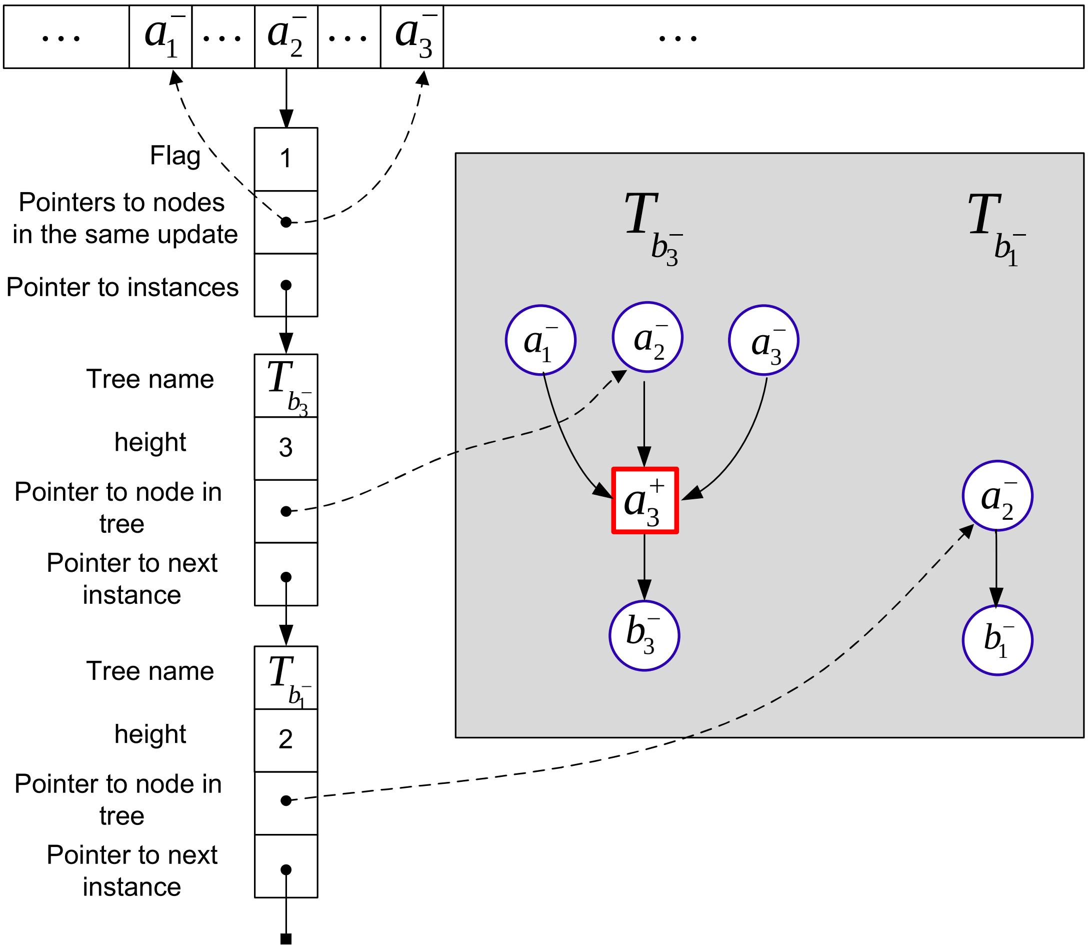


Figure 5: Schematic illustration of a possible implementation of $\mathcal{H}$ for the scenario shown in Figure 2. Only the structure for the node a_2^- is shown. Note that flag's value is 1 since a_2^- appears in both trees.

prior to executing the current MR measurement update, all the local graphs maintained by the robots in the group are identical. The communication protocol assumes all the robots are capable of communicating with each other. Scenarios in which some of the robots can not communicate with other robots are not currently handled and are left for future research. Recall that the identity of the updated robot was denoted by q.

6.1 Communication Protocol

To instantiate an MR measurement, robot q broadcasts its current navigation solution and its covariance to all the other robots in the group, asking each of these robots to transmit back information sets $(\boldsymbol{\zeta}_i(t_i), \mathbf{y}_i(t_i))$ (cf. Section 2) that satisfy a certain predefined criteria. One possible criterion is proximity of the position estimation, i. e.: checking whether the position of the ith robot at the time instant t_i is close enough to the current position of the qth robot. Another alternative is to check if the same scene is observed by the two robots (robot i at time instant t_i and robot q at the current time). Naturally, the involved uncertainty estimation, compared to the covariance obtained from robot q should also be taken into account. Note that t_i can be some time instant from the past $(t_i < t)$, in which case the transmitting robot i needs to extract the information set

$(\boldsymbol{\zeta}_i(t_i), \mathbf{y}_i(t_i))$ from a repository, maintained during the mission. It is also possible that several information sets (from different time instances) are transmitted by the same robot to robot q.

Before some robot i transmits an information set $(\boldsymbol{\zeta}_i(t_i), \mathbf{y}_i(t_i))$, it locates its position in the ith thread in the graph, if a node representing this information set was to be added to the graph. Denote this node by β, and the adjacent nodes in the same thread by α and γ. Then, robot i calculates the transition and noise covariance matrices $\Phi^i_{t_\alpha \to t_\beta}$, $Q^i_{t_\alpha \to t_\beta}$, $\Phi^i_{t_\beta \to t_\gamma}$, $Q^i_{t_\beta \to t_\gamma}$ and transmits these matrices, along with the information set $(\boldsymbol{\zeta}_i(t_i), \mathbf{y}_i(t_i))$, to robot q. In addition, robot i also transmits its covariance matrix $E[\tilde{\mathbf{X}}_i(t_i)\tilde{\mathbf{X}}_i^T(t_i)]$ and the measurement covariance matrix $R_i(t_i)$, which is typically associated to the onboard sensors that produced the measurements $\mathbf{y}_i(t_i)$.

After receiving all the information sets $(\boldsymbol{\zeta}_i(t_i), \mathbf{y}_i(t_i))$ from the rest of the robots in the group, robot q chooses the best $r-1$ sets, which together with the current information set of robot q yields $\{(\boldsymbol{\zeta}_i(t_i), \mathbf{y}_i(t_i))\}_{i=1}^r$. Based on these information sets, robot q calculates the MR measurement $\mathbf{z}$ (cf. Eq. (7)). The decision regarding which sets are better than others can be taken using different criteria and logic. Next, robot q calculates the Jacobian matrices $H_i(t_i)$ and $D_i(t_i)$ for $i \in \{1, \cdots, r\}$. Among these matrices are also $H_q(t)$ and $D_q(t)$, which are computed based on the local information of robot q.

Now, in order to update robot q with the MR measurement $\mathbf{z}$, it is required to calculate the cross-covariance terms $E[\tilde{\mathbf{X}}_i(t_i)\tilde{\mathbf{X}}_j^T(t_j)]$ for all $i, j \in \{1, \cdots, r\}, i \neq j$. To accomplish this step, r a priori nodes are added to appropriate threads and locations in the local graph G (of robot q), representing each of the r information sets. Each such node is connected to adjacent nodes in the same thread by an arc representing a transition relation (cf. Definition 2), where the required transition and process noise covariance matrices were already transmitted (see above). In addition, each node is associated with the transmitted a priori covariance matrix, i.e. $E[\tilde{\mathbf{X}}_i(t_i)\tilde{\mathbf{X}}_i^T(t_i)]$ for a node representing $\tilde{\mathbf{X}}_i(t_i)$. After adding r a priori nodes to the local graph, robot q calculates the required cross-covariance terms by executing Algorithm 1.

Once all the cross-covariance terms are computed, it is possible to calculate the Kalman gain matrix K_q and update robot q (cf. Section 3.2), followed by an update of the graph: The calculated cross-covariance terms are stored in the relevant a priori nodes (e. g., the cross-covariance $E[\tilde{\mathbf{X}}_i(t_i)\tilde{\mathbf{X}}_j^T(t_j)]$ is stored in the two nodes representing $\tilde{\mathbf{X}}_i(t_i)$ and $\tilde{\mathbf{X}}_j(t_j)$), and a new node is added to the thread of robot q in the graph. This is an update node, which is connected to r a priori nodes by arcs representing an update relation (cf. Definition 3).

The final step is to broadcast the update information to the rest of the robots in the group so they could update their local graphs. Therefore, robot q broadcasts the information that participated in the MR update: 1) Identities of the involved $r-1$ robots and its own identity; 2) Gain matrix K_q, time instances t_i, a priori covariance matrices $E[\tilde{\mathbf{X}}_i(t_i)\tilde{\mathbf{X}}_i^T(t_i)]$ and Jacobian matrices $H_i(t_i), D_i(t_i)$ for $i \in \{1, \cdots, r\}$; 3) All the calculated cross-covariance matrices $E[\tilde{\mathbf{X}}_i(t_i)\tilde{\mathbf{X}}_j^T(t_j)]$ with $i, j \in \{1, \cdots, r\}, i \neq j$; 4) The appropriate transition matrices and process and measurement noise matrices for each of the r information sets (see above).

Once this information is obtained by any robot in the group, it follows the same steps as described above for updating its local copy of the graph (all the required quantities are available, thus no computations should be performed). Consequently, at the end of this process all the robots in the group remain with the same graph. The different steps of

Table 1: Communication Protocol and Cost

#	Action	What is transmitted	Overall Cost
1.	Robot q broadcasts its current navigation solution	$\boldsymbol{\zeta}_q(t), E[\tilde{\mathbf{X}}_q(t)\tilde{\mathbf{X}}_q^T(t)]$	$O(n_\zeta^2) \leq O(n^2)$
2.	Reply from other robots	$(\boldsymbol{\zeta}_i(t_i), \mathbf{y}_i(t_i))$, $E[\tilde{\mathbf{X}}_i(t_i)\tilde{\mathbf{X}}_i^T(t_i)]$, $R_i(t_i)$, $\Phi^i_{t_\alpha \to t_\beta}$, $Q^i_{t_\alpha \to t_\beta}$, $\Phi^i_{t_\beta \to t_\gamma}$, $Q^i_{t_\beta \to t_\gamma}$ for $i \in \{1, \cdots, N\}, i \neq q$	$N \cdot O(n^2)$
3.	Robot q broadcasts update information	Identities of the involved robots, K_q, t_i, $H_i(t_i), D_i(t_i)$, $R_i(t_i)$, $\Phi^i_{t_\alpha \to t_\beta}$, $Q^i_{t_\alpha \to t_\beta}$, $\Phi^i_{t_\beta \to t_\gamma}$, $Q^i_{t_\beta \to t_\gamma}$, $E[\tilde{\mathbf{X}}_i(t_i)\tilde{\mathbf{X}}_j^T(t_j)]$ for $i, j \in \{1, \cdots, r\}$	$r^2 O(n_\zeta^2) + rO(n_y^2)$ $\leq r^2 O(n^2)$

the described-above communication protocol are summarized in Table 1.

6.2 Communication Cost Analysis

The communication cost is analyzed in this section assuming the cost involved in transmitting an $n \times m$ matrix is $O(nm)$. Recall that N represents the number of robots in the group and denote by n_{ζ_i} and n_{y_i} the cardinality of $\boldsymbol{\zeta}_i$ and $\mathbf{y}_i$, respectively. Let also,

$$n_\zeta \doteq \max(n_{\zeta_1}, \cdots, n_{\zeta_N}) \quad , \quad n_y \doteq \max(n_{y_1}, \cdots, n_{y_N}) \quad , \quad n \doteq \max(n_\zeta, n_y)$$

Table 1 presents the involved communication cost for the different steps of the communication protocol (cf. Section 6.1) assuming a broadcast involves a single transmission. In the overall, the communication cost is bounded by $(N + r^2)O(n^2)$.

In case a broadcast requires a number of transmissions, an upper bound for the communication cost can be obtained by assuming that a broadcast requires a transmission to each robot in the group. Consequently, steps 1 and 3 in Table 1 should be carried out N times, and therefore the bound on the communication cost is $Nr^2O(n^2)$.

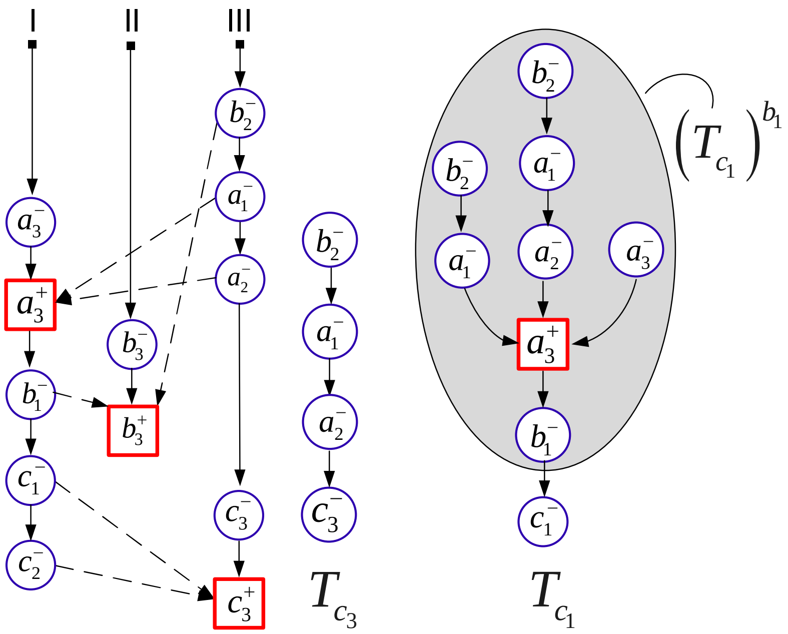
7 Results

In this section, the proposed method is demonstrated for vision-based three-view MR updates [Indelman et al., 2011], [Indelman et al., 2012] in a theoretical example, in a statistical simulation using synthetic imagery, and in an experiment involving real navigation and imagery data. The statistical simulation environment is also used to compare the method with a centralized smoothing approach.

The three-view MR measurements are formulated whenever the same scene is observed by three different views, possibly captured by different robots, i. e., $r = 3$. A short description of the three-view MR measurement model is given in Appendix C. In this case, the residual measurement $\mathbf{z}$ is a function of the robots' navigation solutions and of the matching line-of-sight vectors, calculated based on the measured feature points from the overlapping three images [Indelman et al., 2011]. The residual measurement $\mathbf{z}$ is given by Eq. (11).

7.1 Example

Consider the problem of calculating the term $E[\tilde{\mathbf{X}}_{c_3}^-(\tilde{\mathbf{X}}_{c_1}^-)^T]$ in the example shown in Figure 6(a). The trees $T_{c_3^-}$ and $T_{c_1^-}$ are shown in Figure 6(b). In this example, $E[\tilde{\mathbf{X}}_{c_3}^-(\tilde{\mathbf{X}}_{c_1}^-)^T]$ may be calculated based on the known term $E[\tilde{\mathbf{X}}_{b_2}^-(\tilde{\mathbf{X}}_{b_1}^-)^T]$, which is analyzed upon reaching the fourth level in the two trees. As can be seen, $a_1^-, a_2^- \in \mathcal{D}_{c_3^-}(b_2^-)$ and also $a_1^-, a_2^- \in \mathcal{A}_{c_1^-}(b_1^-)$. Thus, according to Section 4.2.3, the noise terms associated with the path $b_2^- \to a_1^- \to a_2^-$ are *not* statistically independent of $\tilde{\mathbf{X}}_{b_1}^-$.



(a) Graph (b) The trees T_{c_3} and T_{c_1}

Figure 6: An example assuming three-view measurements.

Applying the proposed algorithm, the term $E[\tilde{\mathbf{X}}_{c_3}^-(\tilde{\mathbf{X}}_{c_1}^-)^T]$ is calculated as

$$\begin{aligned} E[\tilde{\mathbf{X}}_{c_3}^-(\tilde{\mathbf{X}}_{c_1}^-)^T] &= \Phi^{III}_{b_2\to c_3} E[\tilde{\mathbf{X}}_{b_2}^-(\tilde{\mathbf{X}}_{b_1}^-)^T](\Phi^{I}_{b_1\to c_1})^T + \Phi^{III}_{a_2\to c_3} Q^{I}_{a_1\to a_2} A_2^T (\Phi^{I}_{a_3\to c_1})^T + \\ &+ \Phi^{III}_{a_1\to c_3} Q^{III}_{b_2\to a_1} (A_1 + A_2 \Phi^{III}_{a_1\to a_2})^T (\Phi^{I}_{a_3\to c_1})^T \end{aligned} \tag{52}$$

with $A_1 = -K_{a_3} H_{a_1}$ and $A_2 = -K_{a_3} H_{a_2}$.

7.2 Statistical Simulation Results

The proposed method is demonstrated in this section in a statistical simulation over the three-view MR updates. A formation comprising two aerial robots is examined. Each of the robots is equipped with its own navigation system and onboard camera.

The following two coordinate systems are defined: 1) *NED*: North-East-Down coordinate system (also known as local-level local-north system). The x, y and z axes are north, east and down, respectively. 2) *Body*: Body-fixed reference frame. The x axis points towards the robot's front, y points right when viewed from above and z completes the setup to yield a Cartesian right hand system.

The navigation solution of each robot is represented by

$$\mathbf{x}_i \doteq \begin{bmatrix} \mathbf{P}_i^T & \mathbf{V}_i^T & \mathbf{\Psi}_i^T \end{bmatrix}^T \quad , \quad i \in \{1, 2\} \tag{53}$$

with $\mathbf{P}, \mathbf{V}, \mathbf{\Psi}$ representing position, velocity and Euler angles, respectively. A basic model $\boldsymbol{\alpha}_i$ of inertial sensor errors (cf. Section 2) is assumed

$$\boldsymbol{\alpha}_i = \begin{bmatrix} \mathbf{d}^T & \mathbf{b}^T \end{bmatrix}^T \tag{54}$$

where $\mathbf{d}$ is the gyro drift and $\mathbf{b}$ is the accelerometer bias. Consequently, the navigation error state vector is composed of (cf. Eq. (3))

$$\mathbf{X}_i = \begin{bmatrix} \Delta\mathbf{x}_i^T & \Delta\boldsymbol{\alpha}_i^T \end{bmatrix}^T \doteq \begin{bmatrix} \Delta\mathbf{P}_i^T & \Delta\mathbf{V}_i^T & \Delta\mathbf{\Psi}_i^T & \Delta\mathbf{d}_i^T & \Delta\mathbf{b}_i^T \end{bmatrix}^T \quad , \quad i \in \{1, 2\} \tag{55}$$

with $\Delta\mathbf{a}$ denoting the error of $\mathbf{a}$. Thus, $\Delta\mathbf{P}, \Delta\mathbf{V}$ are the position and velocity errors, given in the NED system, $\Delta\mathbf{\Psi}$ are the Euler angle errors, and $\Delta\mathbf{d}, \Delta\mathbf{b}$ are the residual drift and bias vectors (i. e., the difference between the true and estimated drift and bias). The continuous system matrix Φ^i satisfying Eq. (4) for the above-defined state vector (Eq. (55)) is given by [Indelman, 2011]

$$\Phi^i = \begin{bmatrix} 0_{3\times3} & I_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & A_s & 0_{3\times3} & C_{NED}^{Body} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & -C_{NED}^{Body} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \end{bmatrix} \tag{56}$$

where the matrix C_{NED}^{Body} is a directional cosine matrix (DCM) transforming from body system to NED system and A_s is a skew-symmetric matrix of the specific force vector $\mathbf{f} = \begin{pmatrix} f_x & f_y & f_z \end{pmatrix}^T$, measured by the accelerometers and expressed in the NED system:

$$A_s = \begin{bmatrix} 0 & -f_D & f_E \\ f_D & 0 & -f_N \\ -f_E & f_N & 0 \end{bmatrix} \qquad \begin{bmatrix} f_N \\ f_E \\ f_D \end{bmatrix} = C_{NED}^{Body} \begin{bmatrix} f_x \\ f_y \\ f_z \end{bmatrix} \tag{57}$$

The above model of Φ^i is valid for short periods of operation [Titterton and Weston, 2004], significantly smaller than the Schuler period (around 84 minutes), which is indeed the case in the considered scenario herein.

The navigation system of Robot I is of a better quality, compared to the navigation system of Robot II. Table 2 presents the assumed initial navigation errors and the errors of the inertial measurement units (IMU) of the two robots.

The two robots performed the same straight and level north-heading trajectory, with Robot I being 2 seconds ahead of Robot II. In the considered scenario, Robot I transmitted information (images and navigation data) to Robot II, thereby allowing updating the navigation system of Robot II using the three-view measurement approach [Indelman et al., 2011, 2012]. Synthetic imagery was used in the simulation with an image noise standard deviation of 1 pixel. The navigation system of Robot I is not updated, and it therefore develops inertial navigation errors over time. Figure 7 summarizes the assumed measurement schedule.

Table 2: Initial Navigation Errors and IMU Errors

Parameter Description	Robot I	Robot II	Units
Initial position error (1σ)	$(10, 10, 10)^T$	$(100, 100, 100)^T$	m
Initial velocity error (1σ)	$(0.1, 0.1, 0.1)^T$	$(0.3, 0.3, 0.3)^T$	m/s
Initial attitude error (1σ)	$(0.1, 0.1, 0.1)^T$	$(0.1, 0.1, 0.1)^T$	deg
IMU drift (1σ)	$(1, 1, 1)^T$	$(10, 10, 10)^T$	deg/hr
IMU bias (1σ)	$(1, 1, 1)^T$	$(10, 10, 10)^T$	mg

Even though the considered scenario may seem rather simple, different time instances are involved in each MR measurement and the time differences between these time instances are not fixed, e. g. $t_{a_2} - t_{a_1} \neq t_{b_2} - t_{b_1}$ (cf. Figure 7). Consequently, this scenario cannot be handled by many of the existing techniques, such as [Roumeliotis and Bekey, 2002, Bahr et al., 2009], without assuming that the time instances are a priori known and the time differences are fixed. It should be noted that, the method by [Bahr et al., 2009] can be also applied[5] in this scenario, however, as mentioned in Section 1, this method requires refining the whole navigation history for all the robots for each new measurement (including IMU or odometry measurements).

Figure 8(a) shows the equivalent graph that was used for calculating the cross-covariance terms in each update event of Robot II, applying Algorithm 1. For example, the two trees $T_{b_3^-}$ and $T_{b_1^-}$, constructed for calculating $P_{b_3^- b_1^-}$ are given in Figure 8(b). In the considered scenario, the conditions of Corollary 3 are satisfied, as in particular can be seen in Figure 8(b). Therefore, the computed cross covariances do *not* involve any noise terms. The obtained cross-covariance terms in the considered scenario maintain a constant structure regardless of how many MR updates were performed so far. For example, the cross-covariance term $P^-_{b_3 b_2}$, required for the second MR update, is similar to Eq. (19):

$$P^-_{b_3 b_2} = \Phi^{II}_{a_3 \to b_3} \left\{ (I - K_{a_3} H_{a_3}) P^-_{a_3 a_2} - K_{a_3} H_{a_2} P^-_{a_2 a_2} - K_{a_3} H_{a_1} P^-_{a_1 a_2} \right\} (\Phi^I_{a_2 \to b_2})^T \tag{58}$$

while the terms $P^-_{b_3 b_1}$ and $P^-_{b_2 b_1}$ are given by

$$P^-_{b_3 b_1} = \Phi^{II}_{a_3 \to b_3} \left\{ (I - K_{a_3} H_{a_3}) P^-_{a_3 a_2} - K_{a_3} H_{a_2} P^-_{a_2 a_2} - K_{a_3} H_{a_1} P^-_{a_1 a_2} \right\} (\Phi^I_{a_2 \to b_1})^T \tag{59}$$

$$P^-_{b_2 b_1} = \Phi^I_{b_1 \to b_2} P^-_{b_1 b_1} \tag{60}$$

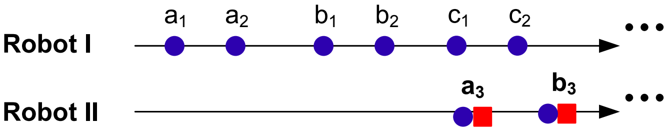


Figure 7: Three-view measurements schedule assumed in the simulation runs.

[5] The method formulation in [Bahr et al., 2009] is given for a general two-robot measurement model. However, to the best of the authors' understanding, their method can be generalized to the general MR measurement model Eq. (6) considered in this paper and in particular to the three-view measurement model considered in this section.

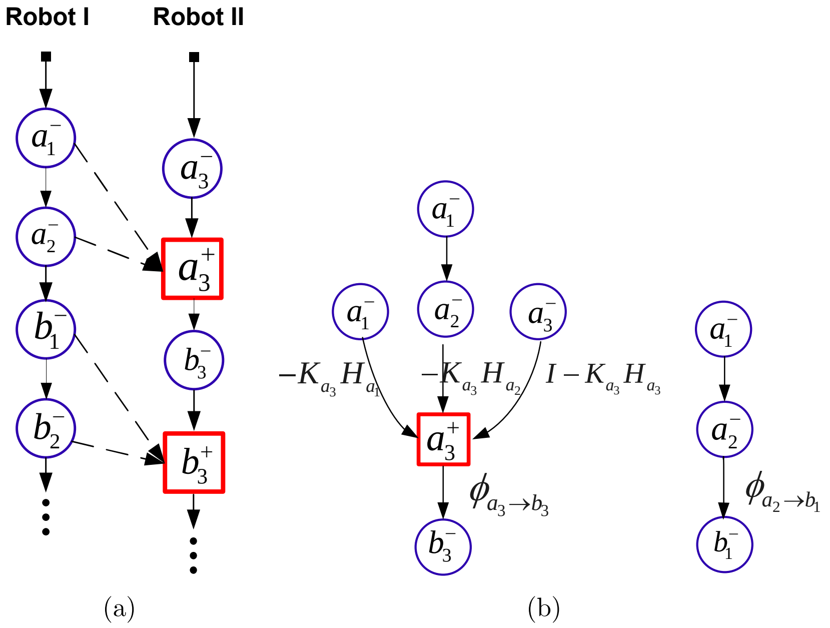


Figure 8: (a) Equivalent graph for the scenario shown in Figure 7. (b) The trees $T_{b_3^-}$ and $T_{b_1^-}$ required for calculating $P_{b_3^- b_1^-}$.

Figures 9 and 10 present Monte-Carlo results (1000 runs) of position and velocity errors of Robot II, compared to Robot I errors. The notations V_N, V_E and V_D, used in Figures 10 and 12, represent velocity errors in the north, east and down directions, respectively. Four curves are shown: mean navigation error (μ), standard deviation (σ), square-root covariance of the filter and the standard deviation of Robot I.

As seen, the errors are considerably reduced in all axes upon each MR update, while maintaining a consistent, unbiased performance. The importance of the cross-covariance terms is clearly evident from Figures 11 and 12. In these figures, the cross-covariance terms were neglected, leading to a biased and inconsistent estimation along the motion heading.

Note that although Robot II is equipped with an inferior navigation system, its performance is not inferior to the performance of Robot I. After several updates, Robot II actually outperforms Robot I. For example, the position errors of Robot II are smaller than the position errors of Robot I. The reason for this phenomenon is that while the measurement is based upon three images, which were obtained from two robots, only one of the robots is actually updated. Updating both robots would yield an improvement in both robots [Roumeliotis and Bekey, 2002]. Referring to Section 3.2, since Robot I contributes two sets of information to each MR measurement (e. g., at t_{a_1} and t_{a_2}), the graph will remain acyclic, if Robot I is updated at t_{a_2} and t_{b_2} (and not at t_{a_1} and t_{b_1}).

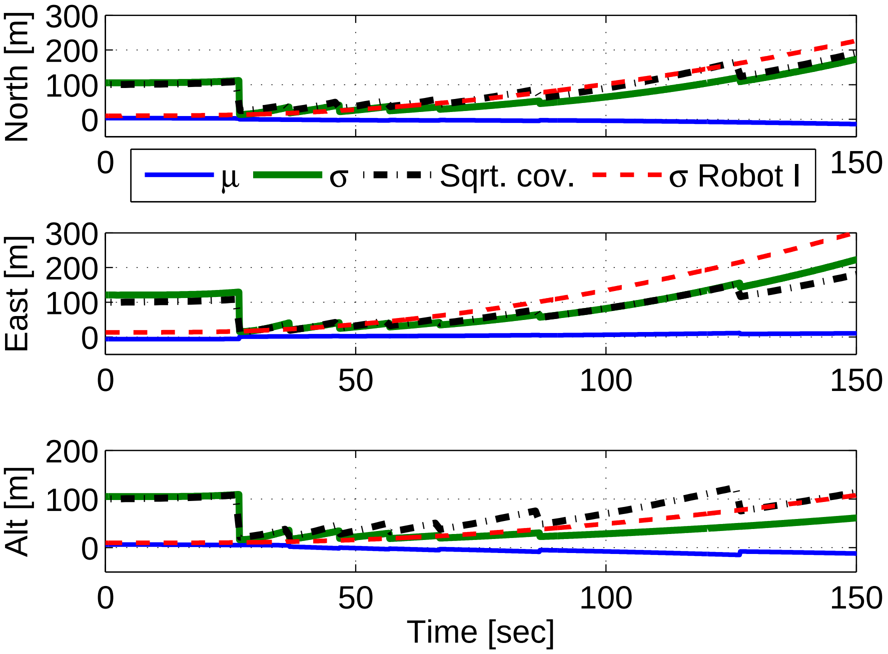


Figure 9: Position errors of Robot II compared to position errors of Robot I in a formation scenario. $\boldsymbol{\mu}, \boldsymbol{\sigma}$ and Sqrt. cov. denote mean error, standard deviation of the error and the square root covariance of the filter, respectively. $\boldsymbol{\sigma}$ Robot I denotes the standard deviation of the error obtained by Robot I. Position errors of Robot II are unbiased ($\boldsymbol{\mu} \approx \mathbf{0}$) and reduced in all axes.

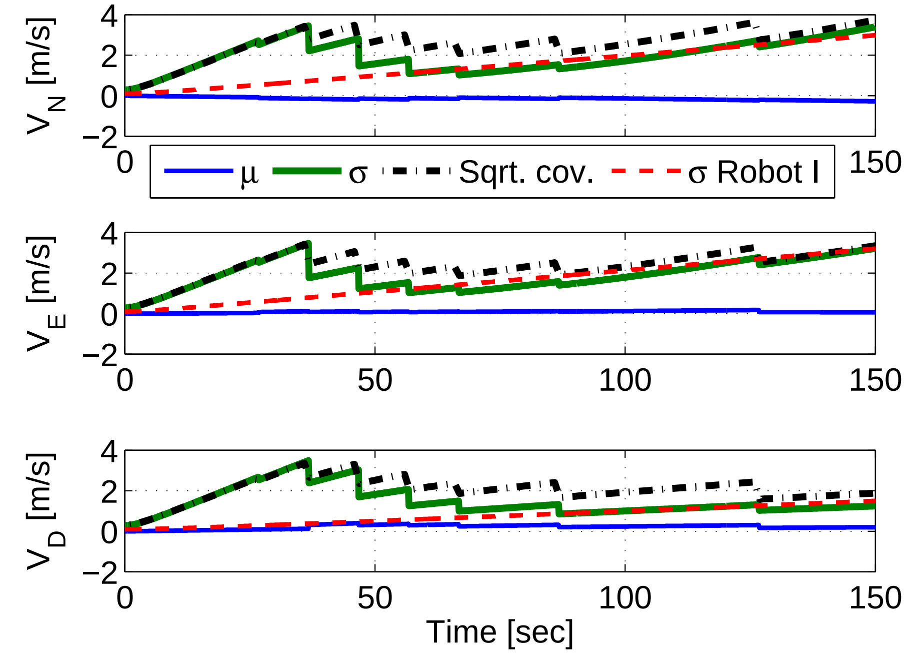


Figure 10: Velocity errors of Robot II compared to velocity errors of Robot I in a formation scenario: Velocity errors of Robot II are unbiased ($\boldsymbol{\mu} \approx \mathbf{0}$) and reduced in all axes. See text and caption of Figure 9 for an explanation of the legend notations.

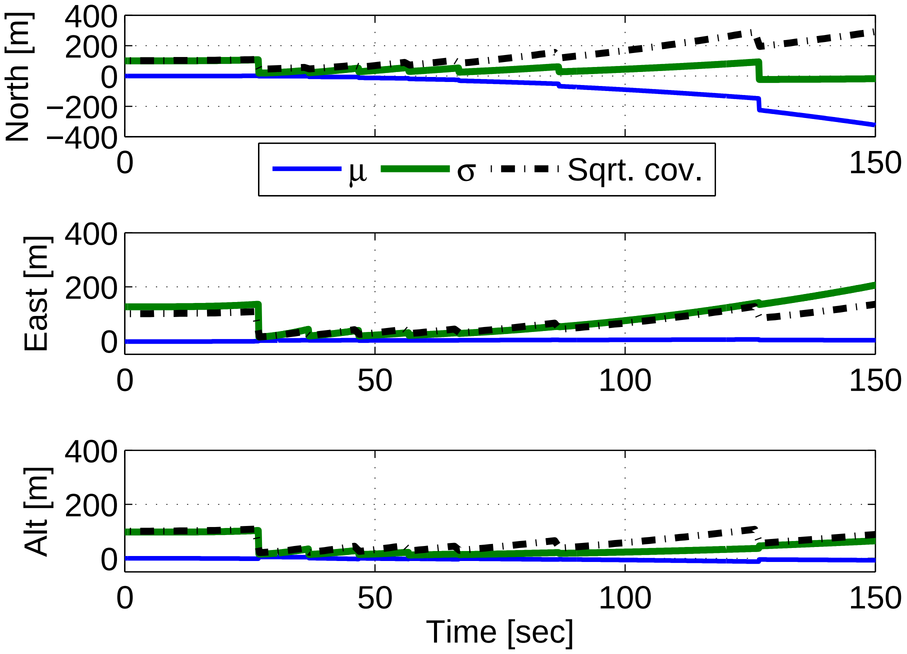


Figure 11: Position errors of Robot II when cross-covariance terms are neglected: Biased estimation ($\boldsymbol{\mu} \neq \mathbf{0}$) is obtained along the motion heading. See text and caption of Figure 9 for an explanation of the legend notations.

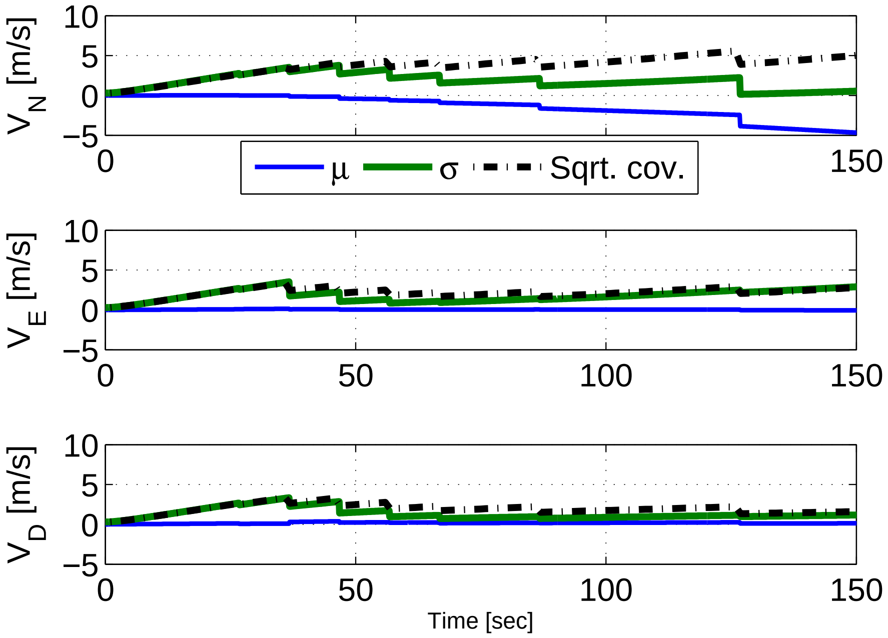


Figure 12: Velocity errors of Robot II when cross-covariance terms are neglected: Biased estimation ($\boldsymbol{\mu} \neq \mathbf{0}$) is obtained along the motion heading. See text and caption of Figure 9 for an explanation of the legend notations.

7.3 Comparison to a Fixed-Lag Centralized Smoothing Approach

In this section the proposed method for cross-covariance calculation is compared to a centralized approach, in which a single state vector comprising information from all the robots in the group is maintained (cf., e. g., [Nerurkar et al., 2009]). The considered scenario consists of two robots that share visual and navigation information and perform a straight and level trajectory as in Section 7.2.

Since the three-view MR measurement model (11) involves different time instances, a fixed-lag centralized smoothing was applied with a state vector and a covariance matrix defined as:

$$\breve{\mathbf{X}}(t_k) \doteq \begin{bmatrix} \breve{\mathbf{X}}_{\mathrm{I}}^T(t_k) & \breve{\mathbf{X}}_{\mathrm{II}}^T(t_k) \end{bmatrix}^T \quad , \quad \breve{P}(t_k) \doteq E[\breve{\mathbf{X}}(t_k)\breve{\mathbf{X}}^T(t_k)] \tag{61}$$

with

$$\breve{\mathbf{X}}_i(t_k) \doteq \begin{bmatrix} \mathbf{X}_i^T(t_k) & \mathbf{X}_i^T(t_{k-1}) & \ldots & \mathbf{X}_i^T(t_{k-w+1}) \end{bmatrix}^T \quad i \in \{\mathrm{I}, \mathrm{II}\} \tag{62}$$

where $\mathbf{X}_i(t_k)$ is the state vector of the i-th robot at time t_k, as defined in Eq. (55), and w is the size of the smoothing lag. The smoothing lag should be long enough to contain the information from all the time instances involved in each MR measurement, which therefore should be known *a priori* (as opposed to the proposed graph-based approach). Thus, since in this section $\mathbf{X}_i \in \mathbb{R}^{15\times 1}$ (cf. Eq. (55)) and recalling that M is the number of robots in the group, the overall dimensions of the centralized smoothing state vector $\breve{\mathbf{X}}(t_k)$ and covariance matrix $\breve{P}(t_k)$ are:

$$\breve{\mathbf{X}}(t_k) \in \mathbb{R}^{15wM\times 1} \quad , \quad \breve{P}(t_k) \in \mathbb{R}^{15wM\times 15wM} \tag{63}$$

In the specific scenario considered in this section, only 2 robots were involved ($M = 2$) and the minimum smoothing window lag to accommodate the measurement schedule (schematically) shown in Figure 7 is $w = 30$, corresponding to storing information of about 30 seconds for each robot in the state vector $\breve{\mathbf{X}}$.

Figures 13-16 present the comparison results obtained from the statistical performance study (as described in Section 7.2). The results are shown in terms of the standard deviation error (σ), while the mean error is zero in both cases (cf. Figures 9 and 10). The results are shown mainly for Robot II, since Robot I develops inertially in the proposed method as it is not updated by the three-view MR measurements in the considered scenario (cf. Section 7.2). While Robot I is updated in the centralized approach, these updates are marginal, as shown for example, in Figure 16 for the velocity state: because Robot I has a better navigation system than Robot II (cf. Table 2), the calculated filter gain matrix distributes the innovation from the three-view MR measurements mainly to Robot II.

As can be seen from Figures 13-15, the overall performance obtained for Robot II using the graph-based approach is similar to the performance of the centralized smoothing approach, although the latter provides better results in some of the components in the navigation state. In particular, position errors (Figure 13) normal to the motion heading (north) are very similar, while the errors along the motion heading are smaller in the centralized approach. This behavior was also obtained in the velocity errors (not shown). Estimation errors of Euler angles and accelerometer bias are very much alike in the two methods (Figures 14 and 15), while the drift state is poorly estimated in both cases (not shown).

The fact that the graph-based approach is somewhat inferior to the centralized smoothing approach is not surprising, since the centralized smoothing approach essentially represents the optimal solution (given the linearization assumptions), while the graph-based approach makes an additional approximation, in the process of calculating the Kalman gain matrix, as mentioned in Section 3.2 (this approximation is made whenever only part of the involved robots are updated, as indeed is the case in this section).

However, while the centralized smoothing approach provides up-to-linearization optimal solution, it has two significant deficiencies compared to the proposed graph-based approach. First, the fixed-lag centralized smoothing approach supports only MR measurements that involve time instances within the fixed lag window. Using a large (or unlimited) smoothing lag is not practical, since the dimensions of the augmented state vector and the augmented covariance matrix grow rapidly as a function of the smoothing lag size (cf. Eq. (63)). Therefore, choosing the actual size of the lag window requires *a priori* knowledge of the MR measurements to be performed. For example, in the scenario considered herein, it was possible to set the smoothing lag to 30 only because the three-view MR measurements schedule was assumed to be known. In contrast to this, the proposed graph-based approach does not require the MR measurement schedule to be known a priori: even if the graph structure has a limited capacity, it can be used to process any MR measurement schedule, provided there is enough space to accommodate the involved information[6]. Moreover, as mentioned in Section 5, upon reaching full capacity, it is possible to apply different considerations, such as identifying low-correlation bonds, when deciding what old information should be removed from the graph. Second, the augmented state vector and covariance matrix should be constantly propagated and updated by the filter, resulting in a continuous high computational load, whereas in the proposed graph-based approach, a basic local state vector is maintained by each robot and the appropriate cross-covariance terms are calculated only when required.

[6] As detailed in [Indelman, 2011], additional navigation information in-between the time instances, participating in the MR measurements schedule, supports construction of the graph. This information is stored in repositories that are locally maintained by the robots in the group.

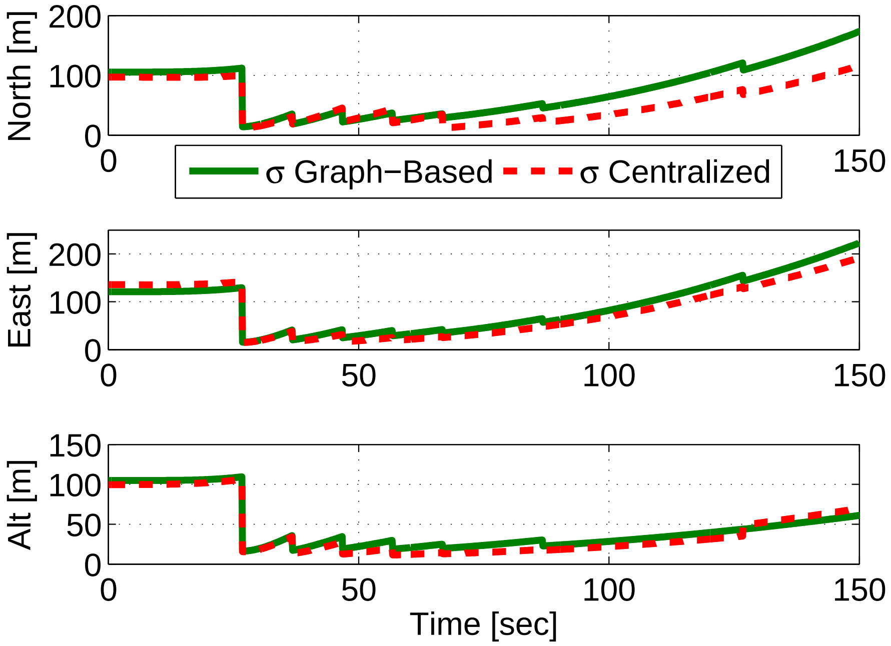


Figure 13: Position errors of Robot II in the two compared methods.

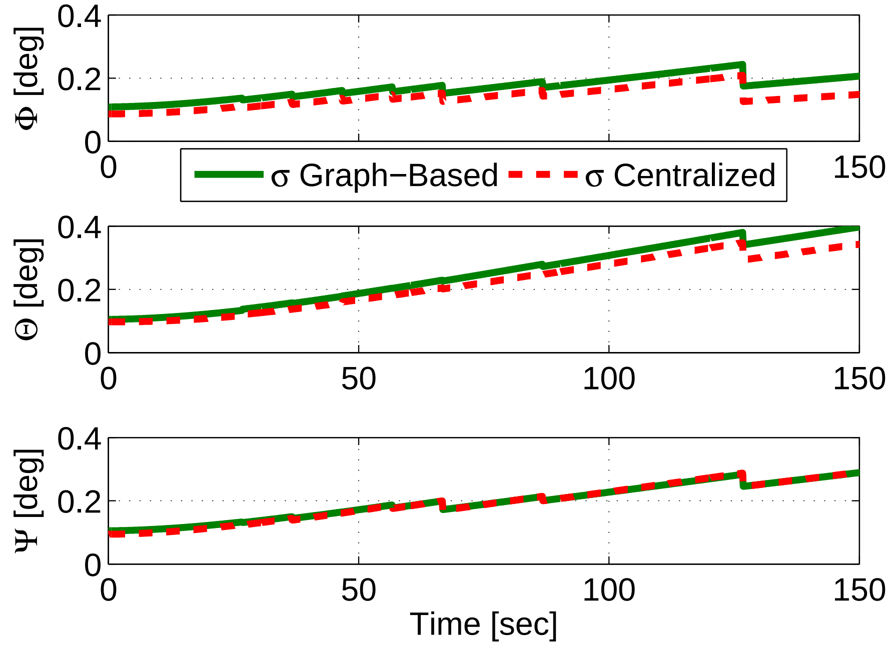


Figure 14: Euler angles errors of Robot II in the two compared methods.

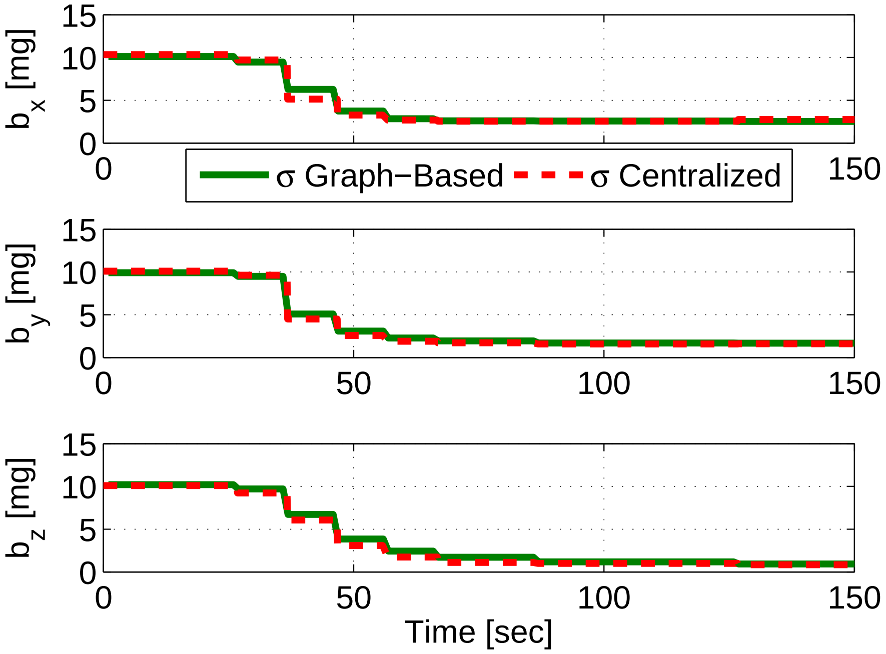


Figure 15: Bias estimation errors of Robot II in the two compared methods.

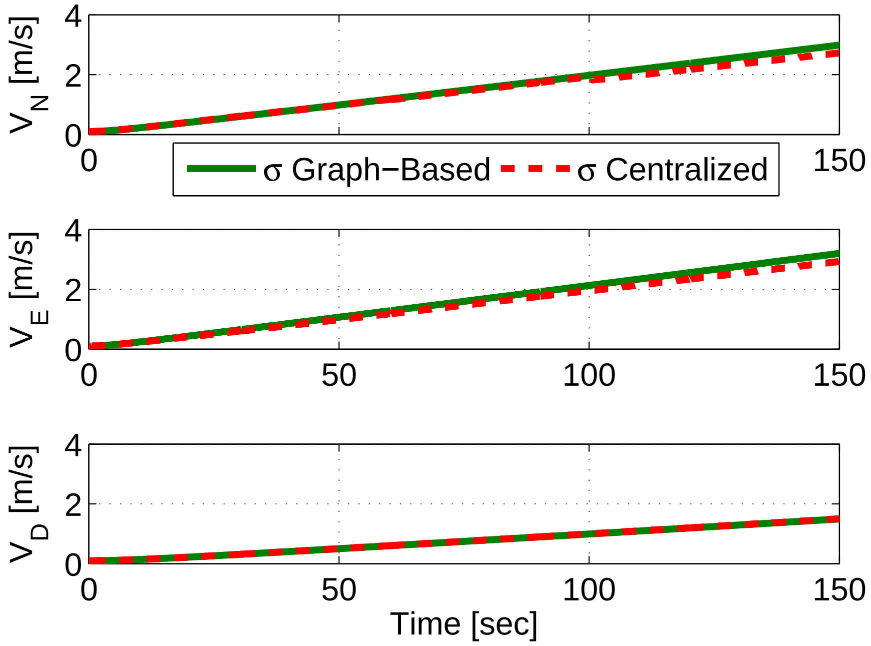


Figure 16: Velocity errors of Robot I in the two compared methods.

Table 3: Measurement schedule details in the experiment: Update type, the involved time instances t_1, t_2, t_3 and the robots identities (I,II) in each three-view measurement update.

Type	t_1 [sec]	t_2 [sec]	t_3 [sec]
MR update	8.4/I	14.2/I	32.6/II
MR update	35.9/I	39.1/I	53.2/II
MR update	2.3/II	5.6/II	60.0/I
MR update	47.9/I	49.2/I	60.6/II
MR update	10.3/II	12.1/II	66.8/I
Self update	0.3/I	1.3/I	81.1/I
Self update	22.8/I	24.3/I	97.0/I
Self update	54.3/I	55.6/I	124.7/I
Self update	70.8/I	72.1/I	142.0/I

7.4 Experiment Results

In this section, the proposed graph-based method for consistent information fusion is demonstrated, in conjunction with the three-view MR measurements, in an experiment. A detailed description of the experiment setup, the involved processing and some of the results have been previously reported in [Indelman et al., 2011] and [Indelman, 2011]. Therefore, this section mainly focuses on analyzing the performance of the graph-based approach.

The experiment setup consists of a single manually-driven ground robot, attached with an IMU and a network camera, capturing images normal to the motion heading. The IMU and imagery data were synchronized and stored for offline processing. Two different trajectories, representing a holding pattern scenario, were performed by the same robot as shown in Figure 17. Since the IMU and the camera were turned off in-between the two trajectories, it is possible to consider a two-robot scenario, i. e. as if the trajectories were performed by two different robots equipped with a similar hardware. Referring to Figure 17, Robot I performed the shown trajectory twice, while Robot II performed this trajectory once, starting from a different location along the shown trajectory. Robot II reached the starting point of Robot I after about 26 seconds. The only available ground-truth data in this experiment is the manually measured location, denoted by diamond and square markings in the figure (the experiment was carried out indoors and thus GPS was unavailable).

Table 3 shows the schedule of the three-view measurements performed in the experiment in terms of the involved time instances (t_1, t_2, t_3) and robot's identities (I,II) in each update. As can be seen, in some cases all the three images in a given three-view measurement were contributed by the same robot, while in other cases the first two images and the third image were captured by different robots. The former case is denoted as "Self update", while the later case is denoted as "MR update". Examples of captured imagery in the experiment and further discussion regarding the image processing phase can be found in [Indelman et al., 2011].

The experiment results are given in Figures 18-19, showing the developing position errors for each of the two robots, calculated as the difference between the reference and

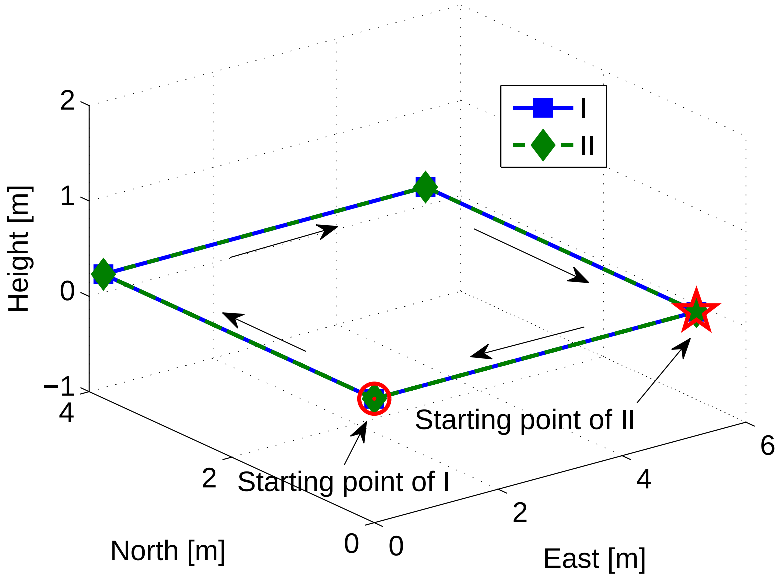


Figure 17: Trajectories of the two robots in an experiment. Each robot has started moving from a different position. Diamond and square markings denote manually-measured robot locations.

the estimated position. The process of position estimation utilized the appropriate cross-covariance terms that were calculated according to the proposed graph-based approach. For a comparison, results are shown when the required cross-covariance terms were neglected.

While the overall performance is significantly improved compared to the inertial scenario [Indelman et al., 2011], in this section the focus is on analyzing the difference in results obtained with and without the cross-covariance terms. As can be seen, the estimation error in the first three-view measurement updates is similar, regardless to whether the involved cross-covariance terms were calculated or neglected, since during these first updates the involved correlations are still insignificant[7]. As more three-view updates are performed, the results do begin to differ.

At this point it is useful to recall that, conceptually, each three-view measurement update is capable of reducing the current position errors (at t_3) only to the error levels that were present at t_1 and t_2, when the first two images were captured [Indelman et al., 2012]. Since no external or absolute information is involved, the a posteriori position error cannot reach substantially lower levels (see also statistical results in Section 7.2). In the general case, the estimation of all navigation states at these two time instances will have some influence on the a posteriori error at t_3, depending on the appropriate cross-covariance terms. Using this insight and since the involved time instances for each three-view measurement are known (cf. Table 3), it is possible to evaluate the expected reduction in the position errors.

As an example, the update of Robot I at $t = 124.7$ sec is analyzed. From Figure 18 it can be seen that the north position error is smaller when using the calculated cross-

[7] In particular, it can be shown that since each three-view measurement is used to update only one robot, at the time of the first update ($t = 32.6$ sec for Robot I, and $t = 60.0$ sec for Robot II) the involved information is statistically independent.

covariance terms, while the east position error is smaller when neglecting these terms (e. g., the east position error is around 5 and 20 meters when the cross-covariance terms are neglected or calculated, respectively). However, the three-view measurement at this time instant ($t_3 = 124.7$) is based on imagery and navigation data from $t_1 = 54.3$ sec and $t_2 = 55.6$ sec, obtained from Robot I (cf. Table 3). The (north, east, alt) position errors at these first two time instances are around $(5, 38, 6.5)$ meters, while the position error at $t_3 = 124.7$ is updated to $(30, 20, 6.5)$ meters and $(45, 5, 5.5)$ meters when using the calculated cross-covariance terms or when neglecting these terms, respectively. Having in mind the above-mentioned insight, one can conclude that the former case, that makes use of the calculated cross-covariance terms, is the more reasonable result: in particular, observe the east position error obtained when neglecting the cross-covariance terms, which drops to around 5 meters, while the a priori error at t_1 and t_2 is around 38 meters.

The same rational can be used to explain other differences between the two shown results, leading to the conclusion that the results are more reasonable when using the graph-based calculated cross-covariance terms over neglecting these terms.

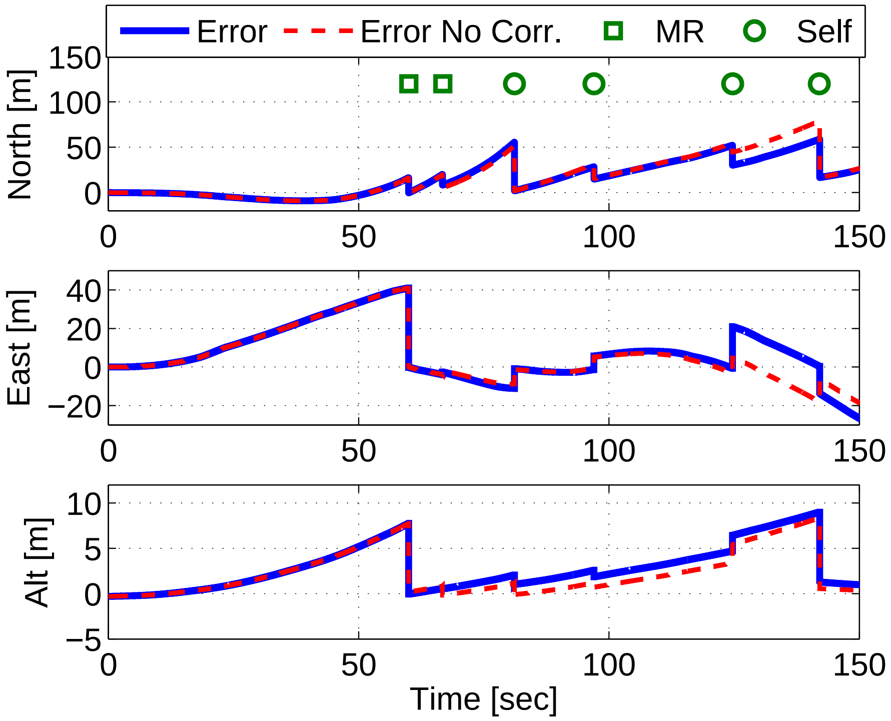


Figure 18: Position errors of Robot I in an experiment.

8 Conclusions

In this paper, a new method was proposed for on-demand, explicit calculation of correlation terms, required for consistent extended Kalman filter-based data fusion in distributed cooperative navigation. The method assumed a general multi-robot model, involving navigation information and readings of onboard sensors of any number of robots, possibly obtained at different time instances.

Each robot in the group maintained a state vector comprised only of its own navigation parameters, while the required correlation terms with other robots were calculated based

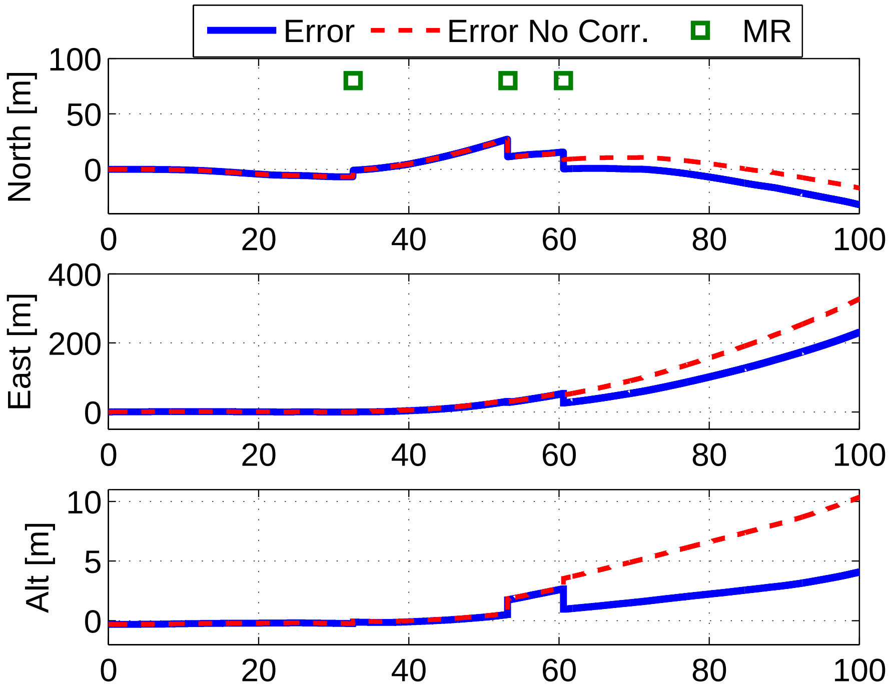


Figure 19: Position errors of Robot II in an experiment.

on a graph, representing all the multi-robot measurement updates performed thus far. This graph was locally maintained by every robot in the group. The developed method is capable of handling the most general scenarios of multi-robot measurements by properly taking into account the involved process and measurement noise terms.

The proposed method was demonstrated in a synthetic example, in a statistical simulation and in an experiment based on a three-view vision-based measurement model, in which a measurement is formulated whenever the same scene is observed by three images. These images can be captured by different robots at different time instances. The presented simulation results were generated using a formation involving two robots, wherein the first robot was equipped with a much better navigation system than the navigation system of the second robot. Applying the proposed method for calculating the correlation terms allowed to obtain consistent, unbiased estimation, rendering the performance of the second robot similar to the performance of the first robot. It was shown that neglecting the correlation terms yields biased and inconsistent estimations of position and velocity. A comparison of the proposed method to a fixed-lag centralized smoothing approach was also presented.

A holding-pattern scenario was demonstrated in the experiment, in which two robots performed the same basic trajectory several times. Experiment results were presented in which the three-view measurements were applied while using the cross-covariance terms, calculated by the proposed method, and while neglecting these terms.

It is our belief that the shown results demonstrate the essential behavior of the proposed method, and thus the performance evaluation in large-scale scenarios is left for future research.

Appendix A: Filter Gain Matrix Calculation

This appendix presents an approach for calculating the filter's gain matrix in case only some of the r robots participating in the MR measurement (8) are updated. If all of the involved robots were updated, it was possible to rewrite the MR measurement equation (8) as

$$\mathbf{z}(t) = \mathcal{H}\boldsymbol{\chi} + \boldsymbol{\nu} \tag{64}$$

with

$$\begin{aligned}
\boldsymbol{\chi} &\doteq \begin{bmatrix}\mathbf{X}_1^T(t_1) & \cdots & \mathbf{X}_r^T(t_r)\end{bmatrix}^T & (65)\\
\mathcal{H} &\doteq \begin{bmatrix}H_1(t_1) & \cdots & H_r(t_r)\end{bmatrix} & (66)\\
\boldsymbol{\nu} &\doteq \sum_{i=1}^{r} D_i(t_i)\mathbf{v}_i(t_i) & (67)
\end{aligned}$$

and calculate the Kalman gain matrix $\mathcal{K}$ as

$$\mathcal{K} = \mathcal{P}\mathcal{H}^T\left(\mathcal{H}\mathcal{P}\mathcal{H}^T + \mathcal{R}\right)^{-1} \tag{68}$$

where $\mathcal{R} = E\left[\boldsymbol{\nu}\boldsymbol{\nu}^T\right]$ and $\mathcal{P} = E\left[\boldsymbol{\chi}\boldsymbol{\chi}^T\right]$. The matrix $\mathcal{P}$ contains cross-covariance terms between the different r robots. However, since the involved time instances $t_1, \cdots, t_r$ are a priori unknown, it is impractical to maintain the overall augmented state vector $\boldsymbol{\chi}$ and its covariance matrix $\mathcal{P}$ (thereby invalidating a centralized solution to the problem). While the covariance terms from all the relevant time instances $E[\mathbf{X}_i(t_i)\mathbf{X}_i(t_i)^T]$ (with $i = 1, \cdots, r$) can be actually stored, the cross-covariance terms $E[\mathbf{X}_i(t_i)\mathbf{X}_i(t_j)^T]$ (with $i \neq j$) should be calculated upon-demand.

The proposed method in this paper calculates these terms assuming an acyclic graph. This assumption is enforced by updating only part of the involved robots (cf. Section 3.2), which essentially means that the MR measurement equation (8) cannot be written in the form of Eq. (64). Assuming that only the first $r_u \leq r$ robots are updated and defining $\breve{\boldsymbol{\chi}}_1$ and $\breve{\boldsymbol{\chi}}_2$ as

$$\begin{aligned}
\breve{\boldsymbol{\chi}}_1 &\doteq \begin{bmatrix}\mathbf{X}_1^T(t_1) & \cdots & \mathbf{X}_{r_u}^T(t_{r_u})\end{bmatrix}^T & (69)\\
\breve{\boldsymbol{\chi}}_2 &\doteq \begin{bmatrix}\mathbf{X}_{r_u+1}^T(t_{r_u+1}) & \cdots & \mathbf{X}_r^T(t_r)\end{bmatrix}^T & (70)
\end{aligned}$$

the measurement equation can be expressed instead as

$$\mathbf{z}(t) = \breve{\mathcal{H}}_1\breve{\boldsymbol{\chi}}_1 + \breve{\boldsymbol{\nu}} \tag{71}$$

with

$$\begin{aligned}
\breve{\mathcal{H}}_1 &\doteq \begin{bmatrix}H_1(t_1) & \cdots & H_{r_u}(t_{r_u})\end{bmatrix} & (72)\\
\breve{\mathcal{H}}_2 &\doteq \begin{bmatrix}H_{r_u+1}(t_{r_u+1}) & \cdots & H_r(t_r)\end{bmatrix} & (73)\\
\breve{\boldsymbol{\nu}} &\doteq \breve{\mathcal{H}}_2\breve{\boldsymbol{\chi}}_2 + \sum_{i=1}^{r} D_i(t_i)\mathbf{v}_i(t_i) & (74)
\end{aligned}$$

One may note that, due to the cross-covariance terms between $\breve{\boldsymbol{\chi}}_1$ and $\breve{\boldsymbol{\chi}}_2$, the equivalent measurement noise $\breve{\boldsymbol{\nu}}$ is no longer statistically independent with the estimated state vector

$\breve{\boldsymbol{\chi}}_1$. Therefore, the basic assumption of the Kalman filter is contradicted. The actual Kalman gain for the r_u robots,

$$\breve{\mathcal{K}} \equiv \begin{bmatrix} K_1^T & \cdots & K_{r_u}^T \end{bmatrix}^T \tag{75}$$

can be calculated using an ad-hoc approach [Indelman et al., 2012] as follows.

$$K_i \doteq P_{\mathbf{X}_i(t_i)\mathbf{z}} P_{\mathbf{z}}^{-1} \quad , \quad i \in \{1, \cdots, r_u\} \tag{76}$$

with

$$P_{\mathbf{X}_i(t_i)\mathbf{z}} = \sum_{j=1}^{r} P_{ij} H_j^T \tag{77}$$

$$P_{\mathbf{z}} = \breve{\mathcal{H}}_1 \breve{\mathcal{P}}_1 \breve{\mathcal{H}}_1^T + \breve{\mathcal{H}}_2 \breve{\mathcal{P}}_2 \breve{\mathcal{H}}_2^T + \mathcal{R} \tag{78}$$

here $\breve{\mathcal{P}}_1 \doteq E[\breve{\boldsymbol{\chi}}_1 \breve{\boldsymbol{\chi}}_1^T]$, $\breve{\mathcal{P}}_2 \doteq E[\breve{\boldsymbol{\chi}}_2 \breve{\boldsymbol{\chi}}_2^T]$ and $P_{ij} = E\left[\mathbf{X}_i(t_i)\mathbf{X}_j^T(t_j)\right]$. Note that when $r_u = r$, i. e. all the involved robots in the MR measurement (8) are updated, the calculated gain matrix is the exact Kalman gain ($\breve{\mathcal{K}} = \mathcal{K}$). The actual update equations of the r_u robots are the standard equations of the extended Kalman filter.

Appendix B: Proofs

This appendix contains proofs of the Lemmas from Section 4.2.3.

Proof of Lemma 1 Suppose that $\boldsymbol{\eta}_{c_\gamma:c_{\gamma-1}}$ and $\tilde{\mathbf{X}}_{d_p}$ are statistically dependent for at least a single value of $\gamma \in \{1, \ldots, m\}$. Then there must exist some node c_r on the path $c_m \to \cdots \to c_r \to \cdots \to c$ in T_c, representing $\tilde{\mathbf{X}}_{c_r}$, such that $\tilde{\mathbf{X}}_{d_p}$ can be expressed in terms of $\tilde{\mathbf{X}}_{c_r}$, and perhaps other state vectors, i. e. $\tilde{\mathbf{X}}_{d_p}$ is a descendant of $\tilde{\mathbf{X}}_{c_r}$. Thus, c_r is an ancestor of d_p, and therefore will appear in $(T_d)^{d_p}$, thereby contradicting the assumption. ■

Proof of Lemma 2 Suppose that the path $d_k \overset{T_d}{\Longrightarrow} d$ does contain a node c_r from the path $c_j \overset{T_c}{\Longrightarrow} c$, with $1 \leq r < j$. Thus, there is a pair of nodes (a, b), with $a = c_r \in V_{T_c}$ and $b = c_r \in V_{T_d}$ such that $E[\tilde{\mathbf{X}}_a \tilde{\mathbf{X}}_b^T] \equiv E[\tilde{\mathbf{X}}_{c_r} \tilde{\mathbf{X}}_{c_r}^T]$ is known.[8]

However, since $r < j$, c_r is closer to c than c_j. Therefore, the pair (a, b) is younger, in the sense of Definition 6, than the pair (c_j, d_k), and thus should have been found while the algorithm processed the rth level. Consequently, this member would have been removed from the permutation set of the rth level, $\mathcal{M}_r$ (cf. Section 4.2.1). Hence, if such a pair indeed existed, then upon reaching the kth level, the permutation set $\mathcal{M}_k$ would not have contained the member (c_k, d_k), since $c_k \in \mathcal{A}_c(a) \equiv \mathcal{A}_c(c_r)$, and $d_k \in \mathcal{A}_d(b) \equiv \mathcal{A}_d(c_r)$ (cf. Eq. (33) for calculating $\mathcal{M}_k$). Since it is given that $(c_k, d_k) \in \mathcal{M}_k$, the node c_r does not exist.

Using the same reasoning, when $r = j$, the node $c_r = c_j$ cannot appear in the path $d_{k-1} \to \cdots \to d$. However, it is possible that $c_j = d_k$, since each node in G may have two children (and only one child in each of the trees). In this case, one of the children is located in T_c, while the other is located in T_d. ■

Proof of Lemma 3 $c_i \in \mathcal{A}_d(d_k)$, and therefore $c_i \in (T_d)^{d_k}$. Since c_i may be reached from any node on the path $c_j \to \cdots \to c_i$, and c_i leads to d_k, any node from $c_j \to \cdots \to c_i$ also leads to d_k. Therefore, $c_j \to \cdots \to c_i$ appears in $(T_d)^{d_k}$. ■

[8] Recall that the covariance of each of the nodes in G is stored (cf. Section 3.3).

Appendix C: Three-View MR Update

The three-view MR updates are based on constraints stemming from observing a static scene from three distinct views, which may be captured by different robots, not necessarily at the same time. These constraints are given by [Indelman et al., 2011]

$$\begin{aligned}
\mathbf{q}_1^T(\mathbf{T}_{12} \times \mathbf{q}_2) &= 0 && \text{(79a)}\\
\mathbf{q}_2^T(\mathbf{T}_{23} \times \mathbf{q}_3) &= 0 && \text{(79b)}\\
(\mathbf{q}_2 \times \mathbf{q}_1)^T(\mathbf{q}_3 \times \mathbf{T}_{23}) &= (\mathbf{q}_1 \times \mathbf{T}_{12})^T(\mathbf{q}_3 \times \mathbf{q}_2) && \text{(79c)}
\end{aligned}$$

where $\mathbf{q}_i$ is a line-of-sight (LOS) vector of the ith view to a static landmark, observed in the three views, and $\mathbf{T}_{ij}$ is the translation vector from the ith view to the jth view, with $i, j \in \{1, 2, 3\}$ and $i \neq j$. All the vectors appearing in Eqs. (79) are expressed in the same coordinate system using the appropriate rotation matrices.

In practice, the three views will have more than a single static landmark in common. Letting N_{ij} be the number of common landmarks observed by views i and j, $i, j \in \{1, 2, 3\}$, $i \neq j$, and denoting by N_{123} the number of landmarks observed by all the three views, the constraints (79) turn into

$$\begin{bmatrix} U \\ F \\ 0 \end{bmatrix}_{N\times 3} \mathbf{T}_{23} = \begin{bmatrix} W \\ 0 \\ G \end{bmatrix}_{N\times 3} \mathbf{T}_{12} \tag{80}$$

where $N \doteq N_{12} + N_{23} + N_{123}$ and

$$\begin{aligned}
U &= \begin{bmatrix}\mathbf{u}_1 & \dots & \mathbf{u}_{N_{123}}\end{bmatrix}^T \quad W = \begin{bmatrix}\mathbf{w}_1 & \dots & \mathbf{w}_{N_{123}}\end{bmatrix}^T && (81)\\
F &= \begin{bmatrix}\mathbf{f}_1 & \dots & \mathbf{f}_{N_{23}}\end{bmatrix}^T \quad G = \begin{bmatrix}\mathbf{g}_1 & \dots & \mathbf{g}_{N_{12}}\end{bmatrix}^T && (82)
\end{aligned}$$

while the vectors $\mathbf{f}, \mathbf{g}, \mathbf{u}, \mathbf{w} \in \mathbb{R}^{3\times 1}$ are defined as

$$\begin{aligned}
\mathbf{f}^T &\doteq (\mathbf{q}_2 \times \mathbf{q}_3)^T && (83)\\
\mathbf{g}^T &\doteq (\mathbf{q}_1 \times \mathbf{q}_2)^T && (84)\\
\mathbf{u}^T &\doteq (\mathbf{q}_1 \times \mathbf{q}_2)^T[\mathbf{q}_3]_\times = \mathbf{g}^T[\mathbf{q}_3]_\times && (85)\\
\mathbf{w}^T &\doteq (\mathbf{q}_2 \times \mathbf{q}_3)^T[\mathbf{q}_1]_\times = \mathbf{f}^T[\mathbf{q}_1]_\times && (86)
\end{aligned}$$

Since navigation and imagery data is imperfect, Eq. (80) will not be satisfied. Therefore, a residual measurement $\mathbf{z}$ is defined as

$$\mathbf{z} \doteq \begin{bmatrix} U \\ F \\ 0 \end{bmatrix}_{N\times 3} \mathbf{T}_{23} - \begin{bmatrix} W \\ 0 \\ G \end{bmatrix}_{N\times 3} \mathbf{T}_{12} \tag{87}$$

The residual measurement $\mathbf{z}$ is a nonlinear function of the navigation solutions attached to the three views, and of the LOS vectors. In a similar manner to Eq. (7) we can write:

$$\mathbf{z}(t) = \mathbf{h}(\{\boldsymbol{\zeta}_i(t_i), \mathbf{y}_i(t_i)\}_{i=1}^r) \tag{88}$$

where, in this case, $\mathbf{y}_i$ are the LOS vectors of view i, appearing in Eq. (87). Linearizing Eq. (88) about $\boldsymbol{\zeta}_i(t_i)$ and $\mathbf{y}_i(t_i)$ gives an expression similar to Eq. (11):

$$\mathbf{z} \approx H_3(t_3)\mathbf{X}_3(t_3) + H_2(t_2)\mathbf{X}_2(t_2) + H_1(t_1)\mathbf{X}_1(t_1) + D\mathbf{v} \tag{89}$$

Acknowledgements

This work was partially supported by the Gordon Center for Systems Engineering at the Technion – Israel Institute of Technology.

References

P. O. Arambel, C. Rago, and R. K. Mehra. Covariance intersection algorithm for distributed spacecraft state estimation. In *Proceedings of the American Control Conference*, pages 4398–4403, Arlington, VA, USA, June 2001.

A. Bahr, M. R. Walter, and J. J. Leonard. Consistent cooperative localization. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3415–3422, Kobe, Japan, May 2009.

J. A. Farrel and M. Barth. *The Global Positioning System and Inertial Navigation.* McGraw-Hill, 1998.

J. W. Fenwick, P. M. Newman, and J. J. Leonard. Cooperative concurrent mapping and localization. In *Proceedings of the IEEE International Conference on Robotics and Automation*, Washington, USA, May 2002.

A. Howard, M. J. Mataric, and G. S. Sukhatme. Putting the 'i' in 'team' - an egocentric approach to cooperative localization. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 868–874, Taipei, Taiwan, 2003.

V. Indelman. Navigation performance enhancement using online mosaicking. Technion, Israel, 2011.

V. Indelman, P. Gurfil, E. Rivlin, and H. Rotstein. Distributed vision-aided cooperative localization and navigation based on three-view geometry. In *Proceedings of the IEEE Aerospace Conference*, Montata, USA, March 2011.

V. Indelman, P. Gurfil, E. Rivlin, and H. Rotstein. Real-time vision-aided localization and navigation based on three-view geometry. *IEEE Transactions on Aerospace and Electronic Systems*, 48(2), April 2012.

M. I. Jordan. Graphical models. *Statistical Science*, 19(1):140–155, 2004.

M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233, 1999.

S. J. Julier and J. K. Uhlmann. A non-divergent estimation algorithm in the presence of unknown correlations. In *Proceedings of the American Control Conference*, pages 2369–2373, Albuquerque, New Mexico, June 1997.

S. J. Julier and J. K. Uhlmann. Using covariance intersection for slam. *Elsevier Robotics and Autonomous Systems*, 55:3–20, 2007.

B. Kim, M. Kaess, L. Fletcher, J. Leonard, A. Bachrach, N. Roy, and S. Teller. Multiple relative pose graphs for robust cooperative mapping. In *Proceedings of the IEEE International Conference on Robotics and Automation*, Anchorage, Alaska, May 2010.

J. Knuth and P. Barooah. Distributed collaborative localization of multiple vehicles from relative pose measurements. In *Forty-Seventh Annual Allerton Conference*, pages 314–321, Illinois, USA, 2009.

R. Kurazume, S. Nagata, and S. Hirose. Cooperative positioning with multiple robots. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1250–1257, San Diego, CA, May 1994.

M. T. Lazaro and J. A. Castellanos. Localization of probabilistic robot formations in slam. In *Proceedings of the IEEE International Conference on Robotics and Automation*, Anchorage, Alaska, May 2010.

S. B. Lazarus, A. Tsourdos, P. Silson, B. White, and R. Zbikowski. Unmanned aerial vehicle navigation and mapping. *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, 222(4):531–548, 2008.

D. M. Malioutov, J. K. Johnson, and A. S. Willsky. Walk-sums and belief propagation in gaussian graphical models. *Journal of Machine Learning Research*, 7:2031–2064, 2006.

L. Merino, J. Wiklund, F. Caballero, A. Moe, J. Ramiro, E. Forssen, K. Nordberg, and A. Ollero. Vision-based multi-uav position estimation. In *IEEE Robotics and Automation Magazine*, pages 53–62, September 2006.

A. I. Mourikis, S. I. Roumeliotis, and J. W. Burdick. Sc-kf mobile robot localization: A stochastic cloning kalman filter for processing relative-state measurements. *IEEE Transactions on Robotics*, 23(4):717–730, 2007.

E. D. Nerurkar, S. I. Roumeliotis, and A. Martinelli. Distributed maximum a posteriori estimation for multi-robot cooperative localization. In *in Proceedings of the IEEE International Conference on Robotics and Automation*, 2009.

S. I. Roumeliotis and G. A. Bekey. Distributed multirobot localization. *IEEE Transactions on Robotics and Automation*, 18(5):781–795, 2002.

R. Sharma and C. N. Taylor. Vision based distributed cooperative navigation for mavs in gps denied areas. In *Proceedings of the AIAA Infotech@Aerospace Conference*, Washington, USA, April 2009.

C. Smaili, M. E. E. Najjar, and F. Charpillet. Multi-sensor fusion method using bayesian network for precise multi-vehicle localization. In *Proceedings of the IEEE International Conference on Intelligent Transportation Systems*, pages 906–911, Beijing, China, 2008.

D. H. Titterton and J. L. Weston. *Strapdown Inertial Navigation Technology*. AIAA, 2004.

X. Xu and S. Negahdaripour. Application of extended covariance intersection principle for mosaic-based optical positioning and navigation of underwater vehicles. In *Proceedings of the IEEE International Conference on Robotics and Automation*, Seoul, Korea, May 2001.