Check for updates

Article

The International Journal of Robotics Research
2024, Vol. 0(0) 1–21
© The Author(s) 2024
Article reuse guidelines:
sagepub.com/journals-permissions
DOI: 10.1177/02783649241230993
journals.sagepub.com/home/ijr
Sage

Decentralized state estimation: An approach using pseudomeasurements and preintegration

Charles Champagne Cossette[1], Mohammed Ayman Shalaby[1], David Saussié[2] and James Richard Forbes[1]

Abstract
This paper addresses the problem of decentralized, collaborative state estimation in robotic teams. In particular, this paper considers problems where individual robots estimate similar physical quantities, such as each other's position relative to themselves. The use of pseudomeasurements *is introduced as a means of modeling such relationships between robots' state estimates and is shown to be a tractable way to approach the decentralized state estimation problem. Moreover, this formulation easily leads to a general-purpose observability test that simultaneously accounts for measurements that robots collect from their own sensors, as well as the communication structure within the team. Finally, input preintegration is proposed as a communication-efficient way of sharing odometry information between robots, and the entire theory is appropriate for both vector-space and Lie-group state definitions. To overcome the need for communicating preintegrated covariance information, a deep autoencoder is proposed that reconstructs the covariance information from the inputs, hence further reducing the communication requirements. The proposed framework is evaluated on three different simulated problems, and one experiment involving three quadcopters.*

Keywords
Relative position estimation, collaborative localization, Lie groups, multi-robot systems, state estimation, preintegration

1. Introduction

Decentralized state estimation is a fundamental requirement for real-world multi-robot deployments. Whether the task is collaborative mapping, relative localization, or collaborative dead-reckoning, the multi-robot estimation problem seeks to estimate the state of each robot given *all* the measurements that each robot obtains locally. This problem is made difficult by the fact that not all robots can communicate with each other and, furthermore, that high-frequency sensor measurements would require substantial communication bandwidth to simply share across the team. A robot might even have insufficient sensors to observe their own state, and hence is dependent on its neighbor's sensors to have a stable estimate. Hypothetically, an estimator that could somehow collect all these sensor measurements on each robot, and fuse them all to jointly estimate the states of every robot in one large system, would have the lowest possible estimation error variance. This is called the *centralized estimator*, but is often infeasible to implement in practice.

A common approach is for robots to share their current state and associated covariance rather than of a history of measurement values (Julier and Uhlmann (1997); Julier (2001); Carrillo-Arce et al. (2013)). This approach has the benefit of simplicity, low communication cost, and fixed message size, but suffers from a well-known issue of not being able to compute cross-correlations between the robots' state estimates (Shalaby et al. (2021b)). Furthermore, in certain problems, robots may be estimating the same physical quantities. As an example, consider two robots estimating each other's position, in addition to their own positions, as shown in Figure 1 (left). Their state vectors are both robots' positions, and therefore, both seek to estimate the same physical quantities, a situation referred to here as *full state overlap*. When robot states have similar, if not identical state definitions, it is straightforward to compute the error between their state estimates using simple subtraction. However, for more complicated problems, especially those with state definitions belonging to arbitrary Lie groups, a generalized measure of error between different robots' state estimates must be introduced.

[1]Department of Mechanical Engineering, McGill University, Montreal, QC, Canada
[2]Department of Electrical Engineering, Polytechnique Montréal, Montreal, QC, Canada

Corresponding author:
Charles Champagne Cossette, Department of Mechanical Engineering, McGill University, 817 Sherbrooke Street West, Macdonald Engineering Bld., Montréal, QC H3A 0C3, Canada.
Email: charles.cossette@mail.mcgill.ca

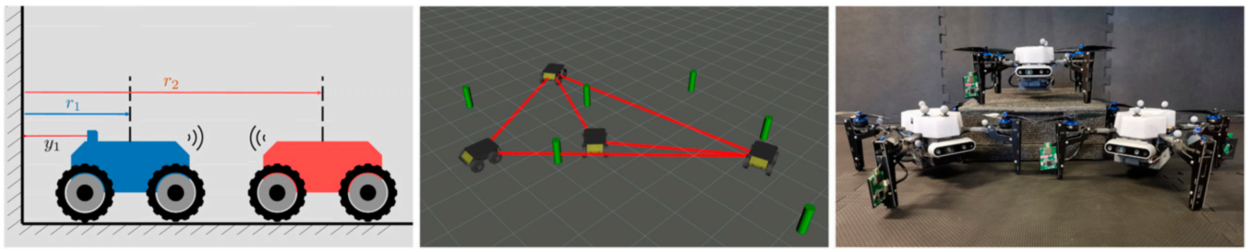

Figure 1. Three examples of decentralized estimation problems within the scope of this paper. **Left:** A toy problem with 1D robots, each estimating both of their positions. **Middle:** A problem with an incomplete communication graph. Robots observe landmarks, have range measurements to each other, and estimate their own and neighbor absolute poses. **Right**: A more complicated experimentally tested problem, where robots equipped with ultra-wideband radios estimate both their own absolute pose and relative poses of neighbors, in addition to IMU biases.

As a concrete example, which is featured experimentally in this paper, consider the case where quadcopters each possess GPS sensors, IMUs, and inter-robot range measurements using ultra-wideband radio. Suppose one quadcopter loses GPS functionality due to traveling under a bridge or a sensor fault, thus losing absolute positioning information. The challenge is to design an algorithm where this faulty robot maintains accurate absolute positioning estimation by appropriately sharing information with its neighbors.

1.1. Contributions

This paper has three main contributions:

1. a framework for decentralized state estimation that uses pseudomeasurements to allow for generic nonlinear relationships between robot states;
2. a preintegration-based method for constant-time, constant-memory, and constant-communication odometry sharing;
3. a theory compatible with Lie-group state definitions, including the familiar vector space.

The pseudomeasurements are shown to be a tractable and effective way to model any generic nonlinear relationship between robot state definitions, including full or partial state overlap as special cases. For example, a nonlinear state relationship is present when robots estimate each other's poses in their own body frames. A pseudomeasurement is introduced for each edge in the communication graph, and the proposed framework also naturally leads to an observability test that takes into account both the local measurements obtained by each robot *and* the communication structure between them. Usage of pseudomeasurements requires robots to communicate their states and corresponding covariances. Furthermore, the common states between robots must be at the same time step, which potentially requires odometry information to also be shared, so that states can be propagated forward to a common time.

The proposed use of preintegration allows sharing odometry information over an arbitrary duration of time, in a lossless matter. The naive alternative is for robots to share a history of odometry measurements since the last time they communicated, which has processing, memory, and communication requirements that grow linearly with the time interval between communications. Preintegration provides a constant-time, constant-memory, and constant-communication alternative that is algebraically identical to simply sharing the input measurements themselves. This makes preintegration a natural choice for multi-robot estimation problems. Moreover, preintegration preserves statistical independence assumptions that typical Kalman filtering prediction steps rely on. Preintegration is best known from the visual-inertial odometry literature (Lupton and Sukkarieh (2012); Forster et al. (2017)), where the same concept, adapted for relative pose estimation is introduced by Shalaby et al. (2023). This paper generalizes the multi-robot-preintegration concept to other common process models in robotics, presents a solution for simultaneous input bias estimation, and also further proposes a deep autoencoder to compress the associated covariance information.

Finally, the proposed solution is general to any state definition, process model, and measurement model subject to typical Gaussian noise assumptions. The complexity of the proposed estimation algorithm is identical to a standard extended Kalman filter, and the communicated messages are lightweight and of fixed length. In the experimental test demonstrated in this paper, each robot transmits information at a rate of only 53 kB/s.

This paper does not focus on the treatment of cross-correlations, and hence employs the simple, well-known covariance intersection (CI) (Julier and Uhlmann (1997); Julier (2001)) method. This allows for an arbitrary communication graph within the robot team, while remaining lightweight and avoiding cumbersome bookkeeping. The main drawback is that CI is proven to yield suboptimal estimation. However, in practice, the performance can remain adequate, and sometimes very comparable to centralized estimation (Shalaby et al. (2021b); Julier (2001)), as shown here from simulated and experimental results.

The remainder of this paper is as follows. Related work is discussed in Section 2 and mathematical preliminaries and notation are shown in Section 3. The paper then starts with a simplified “toy” problem showcasing the proposed method in Section 4, and the theory is generalized in Section 5. Preintegration sharing is presented in Section 6. Finally, Section 7 contains an application to a ground robot simulation and Section 8 applies the method to a more complicated experimental quadcopter problem.

2. Related work

There are many sub-problems associated with the overall decentralized state estimation problem. Even if communication links between robots are assumed to be lossless and have infinite bandwidth, there is still the issue of propagating information over general, incomplete communication graphs. For two robots, it is straightforward to compute centralized-equivalent estimators on each robot as done by Grime and Durrant-Whyte (1994), which use information filters to accumulate the information from a series of measurements, and then communicate this quantity. Grime and Durrant-Whyte (1994) also derived centralized-equivalent solutions for fully connected graphs, as well as tree-shaped graphs. However, they show that for generic graphs, it is impossible to obtain a centralized-equivalent estimate with only neighboring knowledge, and that more knowledge of the graph topology is required.

Leung et al. (2010) present a general centralized-equivalent algorithm for arbitrary time-varying graphs, and is formulated over distributions directly, hence allowing inference using any algorithm such as an extended or sigma-point Kalman filter. Robots must still share raw measurements with each other, therefore requiring substantial bookkeeping. The approach is extended to the SLAM problem by Leung et al. (2012). Roumeliotis and Bekey (2002) decompose the centralized Kalman filter equations using a singular value decomposition to generate independent equations that each robot can compute. Provided that robots have broadcasting ability, and obtain direct pose measurements of their neighbors, a centralized-equivalent solution can be obtained. The consensus Kalman filter (Olfati-Saber and Murray (2004); Olfati-Saber (2005)) aims to asymptotically send the state of n arbitrary nodes to a common value, which is effectively a problem with full state overlap. Battistelli et al. (2015) proposes alternate consensus approaches such as “consensus on information” or “consensus on measurements.” However, the problem does not consider the fact that robots collect their own, separate odometry measurements that are not necessarily available to neighboring robots.

As previously mentioned, one of the simplest solutions to the decentralized estimation problem is to use covariance intersection (Julier and Uhlmann (1997); Julier (2001)). CI conservatively assumes maximum correlation between robot estimates. Although the performance is theoretically suboptimal, the implementation is extremely simple, and imposes no constraints whatsoever on the communication frequency or graph topology. Carrillo-Arce et al. (2013) apply CI to a collaborative localization problem, where each robot estimates their own absolute state given direct relative pose measurements to other robots. Meanwhile, Arambel et al. (2001) present a decentralized state estimation algorithm for multiple spacecraft, where each spacecraft estimates the full state of all vehicles and then utilizes CI to fuse the neighbors’ full state. Recently, split-CI has been introduced to separate states into groups of correlated and independent substates (Li and Nashashibi (2013)), while Li and Yang (2021) exploit CI for the fusion of poses on Lie groups.

When employing CI, a user-defined weighting parameter has to be chosen, which affects the level of inflation of the block-diagonal components of the covariance matrix. Zhu and Kia (2019) formulate an optimization problem where the logarithm of the determinant of the posterior covariance matrix is minimized as a function of the CI weighting parameter, alongside an alternative linear-matrix-inequality approach that estimates the most conservative posterior covariance matrix. Meanwhile, Luft et al. (2018) use an EKF-like filter for decentralized estimation where cross-correlations are also explicitly tracked for both the prediction step and the fusion of local measurements. When relative measurements are encountered, an improved approximation to the joint covariance matrix is developed, which outperforms CI. The approach of Luft et al. (2018) assumes that process model inputs between robots are uncorrelated, which is not applicable in some of the problems in this paper. The work by Jung et al. (2020) builds off of Luft et al. (2018) to solve a full 3D collaborative state estimation problem where each robot has a camera and an IMU.

Another approach using scattering theory has recently been presented for two robots (Allak et al. (2019; 2022)), with the objective of reducing the communication cost associated with high-rate sensor measurements. Also making reference to the IMU preintegration technique (Lupton and Sukkarieh (2012); Forster et al. (2017)), covariance pre-computations are derived by Allak et al. (2019) and later extended to also include the mean (Allak et al. (2022)). It is shown that by sharing pre-computed matrices with twice the size as the state vector, a centralized-equivalent state estimate can be directly obtained with no measurement reprocessing. However, the generalization to more robots does not seem straightforward.

A variety of optimization-based approaches can be seen in the literature, especially when applied to multi-robot simultaneous localization and mapping (SLAM). Tian et al. (2022) have released Kimera-Multi, which uses a distributed pose-graph optimization algorithm to perform metric-semantic SLAM. Lajoie and Beltrame (2023) propose Swarm-SLAM, which performs multi-robot SLAM with an emphasis on using sparsity to minimize the number of data exchanges. However, these distributed SLAM methods are appropriate for situations where each robot has sufficiently rich sensor information via cameras or LIDARs and can perform individual SLAM in the first place. The method of this paper does not impose such a requirement.

3. Preliminaries

This paper will address problems where an individual robot's process model $\mathbf{f}(\cdot)$ and measurement model $\mathbf{g}(\cdot)$ are modeled in the standard form of

$$\begin{aligned} \mathcal{X}_{i_k} &= \mathbf{f}(\mathcal{X}_{i_{k-1}}, \mathbf{u}_{i_{k-1}}, \mathbf{w}_{i_{k-1}}), & \mathbf{w}_{i_{k-1}} &\sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_{i_{k-1}}), \\ \mathbf{y}_{i_k} &= \mathbf{g}(\mathcal{X}_{i_k}) + \mathbf{v}_{i_k}, & \mathbf{v}_{i_k} &\sim \mathcal{N}(\mathbf{0}, \mathbf{R}_{i_k}), \end{aligned} \tag{1}$$

for Robot i, where $\mathbf{u}_{i_k} \in \mathbb{R}^{n_u}$ is the process input at time step k, $\mathbf{y}_{i_k} \in \mathbb{R}^{n_y}$ are the measurements, and $\mathcal{X}_{i_k} \in G$ denotes the robot state belonging to any Lie group G. As a notational convenience, the shorthand $\mathcal{X}_{i:j} = \{\mathcal{X}_i \ \dots \ \mathcal{X}_j\}$ will refer to a collection of arbitrary objects with indices in the range $[i, j]$.

3.1. Lie groups

A Lie group G is a smooth manifold whose elements, given a group operation $\circ$: $G \times G \rightarrow G$, satisfy the group axioms (Solà et al. (2018)). The application of this operation to two arbitrary group elements $\mathcal{X}, \mathcal{Y} \in G$ is written as $\mathcal{X} \circ \mathcal{Y} \in G$. For any G, there exists an associated Lie algebra $\mathfrak{g}$, a vector space identifiable with elements of $\mathbb{R}^m$, where m is referred to as the degrees of freedom of G. Lie algebra elements are related to group elements through the exponential and logarithmic maps, denoted $\exp : \mathfrak{g} \rightarrow G$ and $\log : G \rightarrow \mathfrak{g}$. The "vee" and "wedge" operators are denoted $(\cdot)^\vee : \mathfrak{g} \rightarrow \mathbb{R}^m$ and $(\cdot)^\wedge : \mathbb{R}^m \rightarrow \mathfrak{g}$, which can be used to associate Lie algebra elements with vectors. Composing these operators, group elements can be associated with vectors using

$$\mathcal{X} = \exp(\xi^\wedge) \triangleq \operatorname{Exp}(\xi), \quad \xi = \log(\mathcal{X})^\vee \triangleq \operatorname{Log}(\mathcal{X}),$$

where $\mathcal{X} \in G, \xi \in \mathbb{R}^m$, and the shorthand notation $\operatorname{Exp} : \mathbb{R}^m \rightarrow G$ and $\operatorname{Log} : G \rightarrow \mathbb{R}^m$ has been defined. Following Solà et al. (2018), the adjoint matrix representation of an element $\mathcal{X} \in G$ is denoted $\mathbf{Ad} : G \rightarrow \mathbb{R}^{m \times m}$ and defined such that

$$\mathbf{Ad}(\mathcal{X})\xi = \left(\mathcal{X}\xi^\wedge \mathcal{X}^{-1}\right)^\vee.$$

The most common Lie groups appearing in robotics are $SO(n)$, representing rotations in n-dimensional space, $SE(n)$, representing poses, and $SE_2(3)$ representing "extended" poses that also contain velocity information. In these cases, the elements $\mathcal{X}$ are invertible matrices and the group operation $\circ$ is regular matrix multiplication.

3.1.1. $\oplus$ and $\ominus$ operators. Estimation theory for vector-space states and Lie groups can be elegantly aggregated into a single mathematical treatment by defining generalized "addition" $\oplus : G \times \mathbb{R}^m \rightarrow G$ and "subtraction" $\ominus : G \times G \rightarrow \mathbb{R}^m$ operators, whose precise definitions will depend on the problem at hand. For example, possible implementations include

$$\begin{aligned} \mathcal{X} \oplus \delta\mathbf{x} &= \mathcal{X} \circ \operatorname{Exp}(\delta\mathbf{x}) && \text{(Lie group right)}, \\ \mathcal{X} \oplus \delta\mathbf{x} &= \operatorname{Exp}(\delta\mathbf{x}) \circ \mathcal{X} && \text{(Lie group left)}, \\ \mathbf{x} \oplus \delta\mathbf{x} &= \mathbf{x} + \delta\mathbf{x} && \text{(vector space)}, \end{aligned}$$

for addition and, correspondingly,

$$\begin{aligned} \mathcal{X} \ominus \mathcal{Y} &= \operatorname{Log}(\mathcal{Y}^{-1} \circ \mathcal{X}) && \text{(Lie group right)}, \\ \mathcal{X} \ominus \mathcal{Y} &= \operatorname{Log}(\mathcal{X} \circ \mathcal{Y}^{-1}) && \text{(Lie group left)}, \\ \mathbf{x} \ominus \mathbf{y} &= \mathbf{x} - \mathbf{y} && \text{(vector space)}, \end{aligned}$$

for subtraction. This abstraction is natural since a vector space technically qualifies as a Lie group with regular addition + as the group operation.

3.1.2. Gaussian distributions on Lie groups. As an example use of this abstraction, consider defining a normally distributed Lie group element with mean $\overline{\mathcal{X}}$ and covariance $\mathbf{\Sigma}$, as done by Barfoot and Furgale (2014), with

$$\mathcal{X} = \overline{\mathcal{X}} \circ \operatorname{Exp}(\delta\mathbf{x}), \delta\mathbf{x} \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}),$$

when using a right parameterization, or a similar definition for left parameterizations. This can alternatively be written in an abstract way, applicable to any group or vector space, with

$$\mathcal{X} = \overline{\mathcal{X}} \oplus \delta\mathbf{x}, \delta\mathbf{x} \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}). \tag{2}$$

Moreover, given that $\delta\mathbf{x} = \mathcal{X} \ominus \overline{\mathcal{X}}$, it follows from (2) that

$$p(\mathcal{X}) = \eta \exp\left(-\frac{1}{2}(\mathcal{X} \ominus \overline{\mathcal{X}})^{\mathsf{T}} \mathbf{\Sigma}^{-1} (\mathcal{X} \ominus \overline{\mathcal{X}})\right) \triangleq \mathcal{N}_L(\overline{\mathcal{X}}, \mathbf{\Sigma}),$$

where the reader should note the definition of the generalized Gaussian $\mathcal{N}_L(\overline{\mathcal{X}}, \mathbf{\Sigma})$ (Bourmaud et al. (2016)).

3.1.3. Derivatives on Lie groups. Again following Solà et al. (2018), the Jacobian of a function f: $G \rightarrow G$, taken with respect to $\mathcal{X}$ can be defined as

$$\left.\frac{Df(\mathcal{X})}{D\mathcal{X}}\right|_{\overline{\mathcal{X}}} \triangleq \left.\frac{\partial f(\overline{\mathcal{X}} \oplus \delta\mathbf{x}) \ominus f(\overline{\mathcal{X}})}{\partial \delta\mathbf{x}}\right|_{\delta\mathbf{x}=\mathbf{0}}, \tag{3}$$

where it should be noted that the function $f(\overline{\mathcal{X}} \oplus \delta\mathbf{x}) \ominus f(\overline{\mathcal{X}})$ of $\delta\mathbf{x}$ has $\mathbb{R}^m$ as both its domain and codomain, and can thus be differentiated using any standard technique. With the above general definition of a derivative, it is easy to define the so-called *Jacobian of G* as $\mathbf{J} = D\operatorname{Exp}(\mathbf{x})/D\mathbf{x}$, where left/right group Jacobians are obtained with left/right definitions of $\oplus$ and $\ominus$.

3.1.4. Composite groups. A *composite groups* is simply the concatenation of N other Lie groups $G_1, \dots, G_N$ (Solà et al. (2018)), with elements of the form

$$\boldsymbol{\mathcal{X}} = (\mathcal{X}_1, \dots \mathcal{X}_N) \in G_1 \times \cdots \times G_N.$$

The group operation, inverse, and identity are defined elementwise. For example,

$$\boldsymbol{\mathcal{X}} \circ \boldsymbol{\mathcal{Y}} = (\mathcal{X}_1 \circ \mathcal{Y}_1, \ldots, \mathcal{X}_N \circ \mathcal{Y}_N).$$

Furthermore, defining $\delta \mathbf{x} = [\delta \mathbf{x}_1^{\mathrm{T}} \ \ldots \ \delta \mathbf{x}_N^{\mathrm{T}}]^{\mathrm{T}}$ the $\oplus$, operator is given by

$$\boldsymbol{\mathcal{X}} \oplus \delta \mathbf{x} = (\mathcal{X}_1 \oplus \delta \mathbf{x}_1, \ldots, \mathcal{X}_N \oplus \delta \mathbf{x}_N)$$

and a similar definition applies to $\ominus$.

3.2. Maximum a posteriori

Maximum a posteriori (MAP) is the standard approach taken in the robotics literature. Popular algorithms such as the extended Kalman filter (EKF), iterated EKF, sliding-window filter, and batch estimator can all be derived from a MAP approach, thus unifying them under a common theory. Given a statistically independent measurement $\mathbf{y}$ with a standard model as in (1), as well as a prior distribution $p(\mathcal{X}) = \mathcal{N}_L(\check{\mathcal{X}}, \check{\mathbf{P}})$, the estimate $\hat{\mathcal{X}}$ produced by the MAP approach is

$$\begin{aligned}\hat{\mathcal{X}} &= \underset{\mathcal{X}}{\operatorname{argmax}}\, p(\mathcal{X}|\mathbf{y}) \\ &= \underset{\mathcal{X}}{\operatorname{argmax}}\, \eta \ \ p(\mathbf{y}|\mathcal{X}) p(\mathcal{X}) \\ &= \underset{\mathcal{X}}{\operatorname{argmax}}\ \eta \ \ \mathcal{N}(\mathbf{g}(\mathcal{X}), \mathbf{R}) \ \ \mathcal{N}_L\left(\check{\mathcal{X}}, \check{\mathbf{P}}\right)\end{aligned}$$

where η is a normalization constant. Equivalently, minimizing the negative logarithm yields a nonlinear least-squares problem of the form

$$\hat{\mathcal{X}} = \underset{\mathcal{X}}{\operatorname{argmax}} \frac{1}{2} \mathbf{e}(\mathcal{X})^{\mathrm{T}} \mathbf{W} \mathbf{e}(\mathcal{X}), \tag{4}$$

$$\mathbf{e}(\mathcal{X}) = \begin{bmatrix} \mathcal{X} \ominus \check{\mathcal{X}} \\ \mathbf{y} - \mathbf{g}(\mathcal{X}) \end{bmatrix},$$

where $\mathbf{W} = \operatorname{diag}(\check{\mathbf{P}}^{-1}, \mathbf{R}^{-1})$. Using an on-manifold optimization approach, (4) can be solved by first parameterizing the state with $\mathcal{X} = \hat{\mathcal{X}} \oplus \delta \mathbf{x}$ and solving the problem

$$\delta \hat{\mathbf{x}} = \underset{\delta \mathbf{x}}{\operatorname{argmax}} \frac{1}{2} \mathbf{e}\left(\hat{\mathcal{X}} \oplus \delta \mathbf{x}\right)^{\mathrm{T}} \mathbf{W} \mathbf{e}\left(\hat{\mathcal{X}} \oplus \delta \mathbf{x}\right). \tag{5}$$

Using (3), the Jacobian of the error vector is given by

$$\mathbf{H} \triangleq \left.\frac{D\mathbf{e}(\mathcal{X})}{D\mathcal{X}}\right|_{\hat{\mathcal{X}}} = \left.\frac{\partial \mathbf{e}\left(\hat{\mathcal{X}} \oplus \delta \mathbf{x}\right)}{\partial \delta \mathbf{x}}\right|_{\delta \mathbf{x} = \mathbf{0}},$$

and an approximate solution to (5) can be obtained by solving the Gauss–Newton system

$$\left(\mathbf{H}^{\mathrm{T}} \mathbf{W} \mathbf{H}\right) \delta \hat{\mathbf{x}} = \mathbf{H}^{\mathrm{T}} \mathbf{W} \mathbf{e}\left(\hat{\mathcal{X}}\right).$$

The above is iterated with $\hat{\mathcal{X}} \leftarrow \check{\mathcal{X}} \oplus \delta \hat{\mathbf{x}}$ and initialized with $\hat{\mathcal{X}} \leftarrow \check{\mathcal{X}}$. A common approximation for the posterior covariance $\hat{\mathbf{P}}$ where $p(\mathcal{X}|\mathbf{y}) \approx \mathcal{N}_L(\hat{\mathcal{X}}, \hat{\mathbf{P}})$ is given by $\hat{\mathbf{P}} = (\mathbf{H}^{\mathrm{T}} \mathbf{W} \mathbf{H})^{-1}$ with $\mathbf{H}$ evaluated at $\hat{\mathcal{X}}$.

3.3. Covariance intersection

Covariance intersection (CI) is a tool introduced by Julier and Uhlmann (1997) for the purposes of decentralized data fusion under unknown cross-correlations, and can be summarized with the following lemma.

Lemma 1. Consistency of Covariance Intersection. *The inequality*

$$\begin{bmatrix} \frac{1}{w} \boldsymbol{\Sigma}_{xx} & \mathbf{0} \\ \mathbf{0} & \frac{1}{1-w} \boldsymbol{\Sigma}_{yy} \end{bmatrix} \geq \begin{bmatrix} \boldsymbol{\Sigma}_{xx} & \boldsymbol{\Sigma}_{xy} \\ \boldsymbol{\Sigma}_{xy}^{\mathrm{T}} & \boldsymbol{\Sigma}_{yy} \end{bmatrix}, \tag{6}$$

which applies in the positive definite sense, holds for all $w \in (0, 1)$*, where* $\boldsymbol{\Sigma}_{xx}$, $\boldsymbol{\Sigma}_{yy}$*, and the right-hand-side of* (6) *are positive definite.*

There are several known strategies for choosing w (Julier (2001)). Following Shalaby et al. (2021b), a fixed value of $w = 0.99$ is chosen for all the results shown in this paper, as it is a simple approach that yields acceptable results.

4. A toy problem

Consider first one of the simplest multi-robot estimation problems, shown on the left of Figure 1. Two robots are located at positions r_1 and r_2, respectively, and both robots seek to estimate both robots' positions. By design, each robot carries distinct, conceptually independent estimates, even though their states represent the same true *physical* variables. This mimics exactly what will occur in implementation, as each robot's processor will have a live estimate of both robots' positions. Their state vectors can therefore be defined as

$$\mathbf{x}_1 = \begin{bmatrix} r_1^{[1]} & r_2^{[1]} \end{bmatrix}^{\mathrm{T}}, \mathbf{x}_2 = \begin{bmatrix} r_1^{[2]} & r_2^{[2]} \end{bmatrix}^{\mathrm{T}},$$

where the square bracket superscript $(\cdot)^{[i]}$ is used when necessary to denote Robot i's estimate or "instance" of a common physical variable. Each robot also collects local measurements from its sensors. Robot 1 is capable of measuring its own position,

$$y_1 = \mathbf{G}_1 \mathbf{x}_1 + v_1, \ \ v_1 \sim \mathcal{N}(0, R_1), \ \ \mathbf{G}_1 = [1 \quad 0],$$

while Robot 2 is only capable of measuring its position relative to Robot 1,

$$y_2 = \mathbf{G}_2 \mathbf{x}_2 + v_2, \ \ v_2 \sim \mathcal{N}(0, R_2), \ \ \mathbf{G}_2 = [-1 \quad 1].$$

To keep things simple for this demonstrative problem, robots are assumed to have access to each other's input measurements, such as wheel odometry. This allows them to predict their state forward in time using a conventional Kalman filter. However, a more communication-efficient solution will be proposed in Section 6. Neither robot is capable of estimating their full state vector from local measurements only, meaning that some form of communication will be required.

To reflect the knowledge that the two robots' state vectors are physically the same, a key design choice of this paper is to incorporate a *pseudomeasurement* of the form

$$\mathbf{y}_{12} = \mathbf{x}_1 - \mathbf{x}_2 + \mathbf{v}_{12}, \ \mathbf{v}_{12} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Psi}),$$

whose "measured" value is always exactly zero. This pseudomeasurement can be viewed as a soft constraint on the problem, inversely weighted by the arbitrary pseudo-measurement covariance $\boldsymbol{\Psi}$. The estimation problem is now to compute, as accurately as possible, the posterior distribution

$$p(\mathbf{x}_1, \mathbf{x}_2 | y_1, y_2, \mathbf{y}_{12}).$$

4.1. Solution via MAP

Applying MAP to this simplified problem is to say that

$$\widehat{\mathbf{x}}_1, \widehat{\mathbf{x}}_2 = \underset{\mathbf{x}_1, \mathbf{x}_2}{\operatorname{argma} x} \, p(\mathbf{x}_1, \mathbf{x}_2 | y_1, y_2, \mathbf{y}_{12}). \tag{7}$$

Assuming that v_1, v_2, $\mathbf{v}_{12}$ are all independent random variables, that is, that $p(v_1, v_2, \mathbf{v}_{12}) = p(v_1)p(v_2)p(\mathbf{v}_{12})$, allows the use of Bayes' rule to write

$$\begin{aligned} p(\mathbf{x}_1, \mathbf{x}_2 | y_1, y_2, \mathbf{y}_{12}) = \eta p(y_1|\mathbf{x}_1) p(y_2|\mathbf{x}_2) \\ \times p(\mathbf{y}_{12}|\mathbf{x}_1, \mathbf{x}_2) p(\mathbf{x}_1, \mathbf{x}_2), \end{aligned} \tag{8}$$

where η is a normalization constant that does not depend on $\mathbf{x}_1$ or $\mathbf{x}_2$. Next, assume that the prior distributions of the robots are independent and Gaussian, possibly as a result of using CI,

$$p(\mathbf{x}_1, \mathbf{x}_2) = p(\mathbf{x}_1)p(\mathbf{x}_2) = \mathcal{N}\left(\check{\mathbf{x}}_1, \check{\mathbf{P}}_1\right)\mathcal{N}\left(\check{\mathbf{x}}_2, \check{\mathbf{P}}_2\right). \tag{9}$$

Substituting (9) into (8) and grouping terms into those available to each robot yields

$$\begin{aligned} &p(\mathbf{x}_1, \mathbf{x}_2 | y_1, y_2, \mathbf{y}_{12}) = \\ &\eta p(\mathbf{y}_{12}|\mathbf{x}_1, \mathbf{x}_2)(p(y_1|\mathbf{x}_1)p(\mathbf{x}_1))(p(y_2|\mathbf{x}_2)p(\mathbf{x}_2)). \end{aligned} \tag{10}$$

Since the local measurement models are linear, it is straightforward to exactly compute the terms

$$p(y_i|\mathbf{x}_i)p(\mathbf{x}_i) = \eta_i p(\mathbf{x}_i|y_i) = \eta_i \mathcal{N}\left(\tilde{\mathbf{x}}_i, \tilde{\mathbf{P}}_i\right), i = 1, 2, \tag{11}$$

using the regular Kalman filter equations. The means and covariances $\tilde{\mathbf{x}}_i, \tilde{\mathbf{P}}_i$ (with tildes) represent the distribution of each robot's state conditioned on only local measurements, without the information that the robots' states are physically the same. Substituting (11) into (10) yields a simplified expression for the posterior, and the optimization problem (7) now leads to the least-squares problem

$$\widehat{\mathbf{x}}_1, \widehat{\mathbf{x}}_2 = \underset{\mathbf{x}_1, \mathbf{x}_2}{\operatorname{argmin}} \frac{1}{2} \mathbf{e}(\mathbf{x}_1, \mathbf{x}_2)^{\mathrm{T}} \mathbf{W} \mathbf{e}(\mathbf{x}_1, \mathbf{x}_2),$$

where

$$\mathbf{e}(\mathbf{x}_1, \mathbf{x}_2) = \begin{bmatrix} \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} \\ \mathbf{1} & -\mathbf{1} \end{bmatrix} \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix} - \begin{bmatrix} \tilde{\mathbf{x}}_1 \\ \tilde{\mathbf{x}}_2 \\ \mathbf{0} \end{bmatrix} \triangleq \mathbf{H}\mathbf{x} - \mathbf{z},$$

$$\mathbf{W} = \operatorname{diag}\left(\tilde{\mathbf{P}}_1^{-1}, \tilde{\mathbf{P}}_2^{-1}, \boldsymbol{\Psi}^{-1}\right).$$

In this linear case the unique solution $\widehat{\mathbf{x}}$ is given by

$$\begin{bmatrix} \widehat{\mathbf{x}}_1 \\ \widehat{\mathbf{x}}_2 \end{bmatrix} = \left(\mathbf{H}^{\mathrm{T}}\mathbf{W}\mathbf{H}\right)^{-1}\mathbf{H}^{\mathrm{T}}\mathbf{W}\mathbf{z}, \tag{12}$$

which is also known to be the mean. The relevant matrices expand to

$$\begin{aligned} \mathbf{H}^{\mathrm{T}}\mathbf{W}\mathbf{H} &= \begin{bmatrix} \tilde{\mathbf{P}}_1^{-1} + \boldsymbol{\Psi}^{-1} & -\boldsymbol{\Psi}^{-1} \\ -\boldsymbol{\Psi}^{-1} & \tilde{\mathbf{P}}_2^{-1} + \boldsymbol{\Psi}^{-1} \end{bmatrix}, \\ \mathbf{H}^{\mathrm{T}}\mathbf{W}\mathbf{z} &= \begin{bmatrix} \tilde{\mathbf{P}}_1^{-1}\tilde{\mathbf{x}}_1 \\ \tilde{\mathbf{P}}_2^{-1}\tilde{\mathbf{x}}_2 \end{bmatrix}. \end{aligned}$$

The next steps involve various applications of the Sherman-Morrison-Woodbury (SMW) identities to analytically invert the inverse covariance matrix $\mathbf{H}^{\mathrm{T}}\mathbf{W}\mathbf{H}$, as well as solve for the solution using (12). The derivation details are omitted for brevity but follow the same steps as (Barfoot 2023, Ch. 3.3.2). The eventual result is

$$\begin{aligned} \widehat{\mathbf{x}} &\triangleq \begin{bmatrix} \widehat{\mathbf{x}}_1 \\ \widehat{\mathbf{x}}_2 \end{bmatrix} = \begin{bmatrix} \tilde{\mathbf{x}}_1 + \mathbf{K}_1(\tilde{\mathbf{x}}_2 - \tilde{\mathbf{x}}_1) \\ \tilde{\mathbf{x}}_2 + \mathbf{K}_2(\tilde{\mathbf{x}}_1 - \tilde{\mathbf{x}}_2) \end{bmatrix}, \\ \widehat{\mathbf{P}} &\triangleq \left(\mathbf{H}^{\mathrm{T}}\mathbf{W}\mathbf{H}\right)^{-1} \\ &= \begin{bmatrix} (\mathbf{1} - \mathbf{K}_1)\tilde{\mathbf{P}}_1 & -\mathbf{K}_1\tilde{\mathbf{P}}_2 \\ -\mathbf{K}_2\tilde{\mathbf{P}}_1 & (\mathbf{1} - \mathbf{K}_2)\tilde{\mathbf{P}}_2 \end{bmatrix}, \\ \mathbf{K}_1 &\triangleq \tilde{\mathbf{P}}_1\left(\boldsymbol{\Psi} + \tilde{\mathbf{P}}_2 + \tilde{\mathbf{P}}_1\right)^{-1}, \\ \mathbf{K}_2 &\triangleq \tilde{\mathbf{P}}_2\left(\boldsymbol{\Psi} + \tilde{\mathbf{P}}_2 + \tilde{\mathbf{P}}_1\right)^{-1}, \end{aligned} \tag{13}$$

and furthermore $p(\mathbf{x}_1, \mathbf{x}_2 | \mathbf{y}_1, \mathbf{y}_2, \mathbf{y}_{12}) = \mathcal{N}(\widehat{\mathbf{x}}, \widehat{\mathbf{P}})$, which is a standard result from MAP approaches (Barfoot (2023)). The final individual estimates are obtained by marginalizing out the other robots' states, which is trivial to do in covariance form by simply extracting the corresponding blocks from $\widehat{\mathbf{x}}, \widehat{\mathbf{P}}$, yielding

$$\begin{aligned} p(\mathbf{x}_1 | \mathbf{y}_1, \mathbf{y}_2, \mathbf{y}_{12}) = \mathcal{N}\left(\widehat{\mathbf{x}}_1, (\mathbf{1} - \mathbf{K}_1)\tilde{\mathbf{P}}_1\right), \\ p(\mathbf{x}_2 | \mathbf{y}_1, \mathbf{y}_2, \mathbf{y}_{12}) = \mathcal{N}\left(\widehat{\mathbf{x}}_2, (\mathbf{1} - \mathbf{K}_2)\tilde{\mathbf{P}}_2\right), \end{aligned}$$

The equations (13) has a form similar to a situation where robots simply treated the other robot's state estimate as a "measurement" of their own state. This is a result that is specific to this simple toy problem, where robots have full state overlap.

Conditioning on the pseudomeasurement $\mathbf{y}_{12}$ has introduced cross-correlation terms in (13), which are feasible to keep track of for this two-robot scenario, but introduce substantial complexity for an arbitrary multi-robot scenario. Therefore, this paper simply employs the CI approximation as required, including for this toy problem for the sake of consistency. Specifically, before each state fusion using (13), inflate the covariance matrices with

$$\tilde{\mathbf{P}}_1 \leftarrow \frac{1}{w}\tilde{\mathbf{P}}_1, \ \tilde{\mathbf{P}}_2 \leftarrow \frac{1}{1-w}\tilde{\mathbf{P}}_2, \tag{14}$$

where $w = 0.99$ is used.

Figure 2 shows the estimation error of each robot as multiple pseudomeasurements are fused in succession. The two robots' estimates not only converge to zero error, but also to a common value, which is the main effect of the pseudomeasurement. A pseudomeasurement covariance of $\mathbf{\Psi} = 10 \cdot \mathbf{1}$ was chosen for this simulation to show its effect, but smaller values can be used. Since the expressions in (13) are in covariance form, it is even possible to use $\mathbf{\Psi} = \mathbf{0}$, in which case the two estimates will converge together after the first pseudomeasurement. For the prior distributions, arbitrary Gaussian distributions were chosen, with the initial true states drawn from these distributions.

In Figure 3, 100 Monte Carlo trials are performed on a simulation of this toy problem, but extended to four robots using the methods from the next section. The root-mean-squared error (RMSE) and normalized estimation error squared (NEES), calculated as per (Bar-Shalom et al., 2001, Ch. 5.4), are plotted through time. The lines marked "Proposed" fuse pseudomeasurements as described,

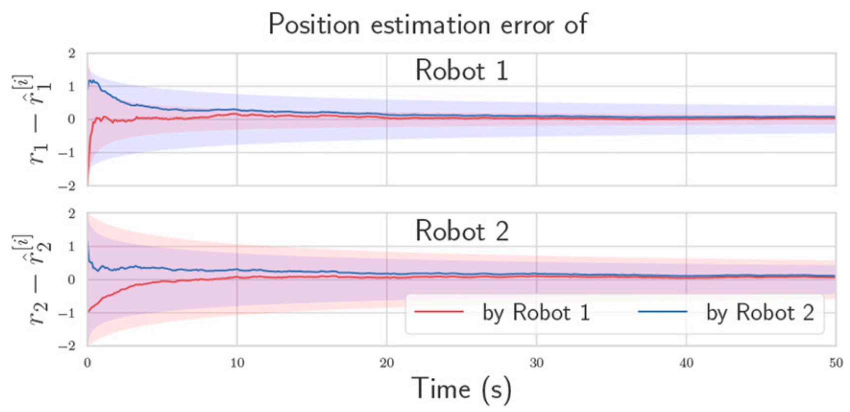

Figure 2. Estimation convergence for a single trial of the two-robot toy problem with $\mathbf{\Psi} = 10 \cdot \mathbf{1}$. Due to pseudomeasurements, the robot states successfully converge to a common value.

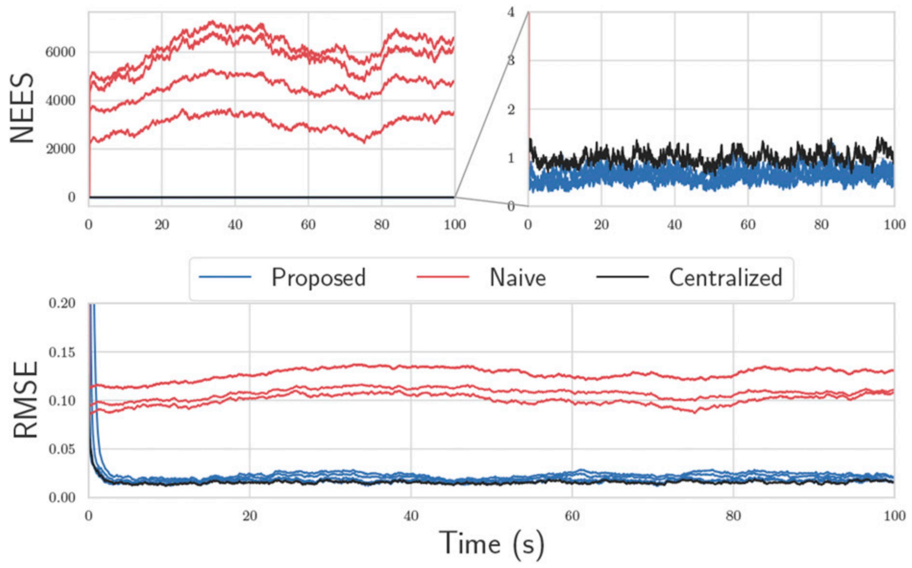

Figure 3. Results of 100 Monte Carlo trials for a four-robot version of the toy problem. The top two plots consist of a NEES plot, which is a measure of consistency. The bottom plot is the RMSE of the state. The proposed solution, which performs CI, remains statistically consistent and has reasonably low error in many cases.

and use CI before each state fusion. The naive solution is identical, but does not perform the covariance intersection step in (14) before state fusion, thus completely neglects cross-correlations. The centralized solution is simply a Kalman filter with state $\mathbf{x} = [r_1 \;\; r_2]^T$ fusing both the measurements y_1 and y_2 using the standard equations. Although the use of CI does introduce error compared to the centralized solution, it is still vastly better than the naive approach.

5. General problem

Consider now N robots, which communicate in correspondence with an arbitrary undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V} = \{1, \ldots, N\}$ is the set of nodes or robot IDs, and $\mathcal{E}$ is the set of edges. The robots have states $\mathcal{X}_i \in G_i, i \in \mathcal{V}$ belonging to possibly different groups. Pseudomeasurement models $\mathbf{c}_{ij} : G_i \times G_j \to \mathbb{R}^c$ are now defined in a generic way

$$\mathbf{y}_{ij_k} = \mathbf{c}_{ij}\left(\mathcal{X}_{i_k}, \mathcal{X}_{j_k}\right) + \mathbf{v}_{ij_k}, \mathbf{v}_{ij_k} \sim \mathcal{N}(\mathbf{0}, \mathbf{\Psi}),$$

for each pair $(i,j) \in \mathcal{E}$. These models are designed by the user, and should correspond with redundant, common, or over-parameterizations of states appearing on different robots, such as when two robots are estimating the same physical quantities. However, these quantities can also differ in the way they are represented from robot to robot, such as each robot resolving the same physical pose in their own frame, and hence, the pseudomeasurement function is left as a general nonlinear function. Examples shall be given in Section 7 and 8.

Using MAP, the general state fusion problem is now

$$\hat{\mathcal{X}}_{1:N} = \underset{\mathcal{X}_{1:N}}{\operatorname{argma} x}\; p\left(\mathcal{X}_{1:N} \middle| \mathbf{y}_{ij}\right), \text{for all } (i,j) \in \mathcal{E}.$$

It is easier to instead consider a single pseudomeasurement at a time, such as, without loss of generality, $\mathbf{y}_{12}$. The posterior distribution given only $\mathbf{y}_{12}$ is

$$\begin{aligned} p(\mathcal{X}_{1:N}|\mathbf{y}_{12}) &= \eta p(\mathbf{y}_{12}|\mathcal{X}_1, \mathcal{X}_2) p(\mathcal{X}_{1:N}) \\ &= \eta \; \mathcal{N}(\mathbf{c}_{12}(\mathcal{X}_1, \mathcal{X}_2), \mathbf{\Psi}) \; \prod_{i=1}^{N} \mathcal{N}_L\left(\tilde{\mathcal{X}}_i, \tilde{\mathbf{P}}_i\right) \end{aligned} \tag{15}$$

where robot state priors have again been assumed to be independent, as a result of using covariance intersection. Due to this independence, the variables $\mathcal{X}_{3:N}$ can be removed from the optimization problem since their optimal values are simply $\hat{\mathcal{X}}_{3:N} = \tilde{\mathcal{X}}_{3:N}$ and have no effect on $\mathcal{X}_1, \mathcal{X}_2$. Minimizing the negative logarithm of (15), omitting terms involving $\mathcal{X}_{3:N}$, leads to a least-squares problem with error vector given by

$$\mathbf{e}(\mathcal{X}_1, \mathcal{X}_2) = \begin{bmatrix} \mathcal{X}_1 \ominus \tilde{\mathcal{X}}_1 \\ \mathcal{X}_2 \ominus \tilde{\mathcal{X}}_2 \\ -\mathbf{c}_{12}(\mathcal{X}_1, \mathcal{X}_2) \end{bmatrix},$$

and weight $\mathbf{W} = \operatorname{diag}(\tilde{\mathbf{P}}_1^{-1}, \tilde{\mathbf{P}}_2^{-1}, \mathbf{\Psi}^{-1})$. Defining $\mathbf{J}_i$ as the group Jacobian associated with G_i, as well as $\mathbf{S}_i, \mathbf{S}_j$ being the Jacobians of $\mathbf{c}_{ij}$ with respect to $\mathcal{X}_i, \mathcal{X}_j$, respectively, the Jacobian of the error vector is

$$\mathbf{H} = \begin{bmatrix} \mathbf{J}_1^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{J}_2^{-1} \\ -\mathbf{S}_1 & -\mathbf{S}_2 \end{bmatrix},$$

which is written without arguments $(\mathcal{X}_1, \mathcal{X}_2)$ for brevity. The relevant terms of the Gauss–Newton system are

$$\mathbf{H}^{\mathrm{T}}\mathbf{W}\mathbf{H} = \begin{bmatrix} \mathbf{J}_1^{-\mathrm{T}}\mathbf{P}_1^{-1}\mathbf{J}_1^{-1} + \mathbf{S}_1^{\mathrm{T}}\mathbf{\Psi}^{-1}\mathbf{S}_1 & \mathbf{S}_1^{\mathrm{T}}\mathbf{\Psi}^{-1}\mathbf{S}_2 \\ \mathbf{S}_2^{\mathrm{T}}\mathbf{\Psi}^{-1}\mathbf{S}_1 & \mathbf{J}_2^{-\mathrm{T}}\mathbf{P}_2^{-1}\mathbf{J}_2^{-1} + \mathbf{S}_2^{\mathrm{T}}\mathbf{\Psi}^{-1}\mathbf{S}_2 \end{bmatrix},$$

$$\mathbf{H}^{\mathrm{T}}\mathbf{W}\mathbf{e}(\mathcal{X}_1, \mathcal{X}_2) = \begin{bmatrix} \mathbf{J}_1^{-\mathrm{T}}\mathbf{P}_1^{-1}\left(\mathcal{X}_1 \ominus \tilde{\mathcal{X}}_1\right) - \mathbf{S}_1^{\mathrm{T}}\mathbf{\Psi}^{-1}\mathbf{c}_{12}(\mathcal{X}_1, \mathcal{X}_2) \\ \mathbf{J}_2^{-\mathrm{T}}\mathbf{P}_2^{-1}\left(\mathcal{X}_2 \ominus \tilde{\mathcal{X}}_2\right) - \mathbf{S}_2^{\mathrm{T}}\mathbf{\Psi}^{-1}\mathbf{c}_{12}(\mathcal{X}_1, \mathcal{X}_2) \end{bmatrix},$$

which, by substantial manipulation with the SMW identities, can be used to analytically compute $\delta\hat{\mathbf{x}} = (\mathbf{H}^{\mathrm{T}}\mathbf{W}\mathbf{H})^{-1}\mathbf{H}^{\mathrm{T}}\mathbf{W}\mathbf{e}(\hat{\mathcal{X}}_1, \hat{\mathcal{X}}_2)$, producing on-manifold iterated-EKF-like expressions. The result is

$$\begin{aligned} \delta\hat{\mathbf{x}}_1 &= -\mathbf{J}_1\left(\hat{\mathcal{X}}_1 \ominus \tilde{\mathcal{X}}_1\right) + \mathbf{K}_1\mathbf{z}, \\ \delta\hat{\mathbf{x}}_2 &= -\mathbf{J}_2\left(\hat{\mathcal{X}}_2 \ominus \tilde{\mathcal{X}}_2\right) + \mathbf{K}_2\mathbf{z}, \\ \mathbf{K}_1 &= \mathbf{J}_1\tilde{\mathbf{P}}_1\mathbf{J}_1^{\mathrm{T}}\mathbf{S}_1^{\mathrm{T}}\mathbf{V}^{-1}, \\ \mathbf{K}_2 &= \mathbf{J}_2\tilde{\mathbf{P}}_2\mathbf{J}_2^{\mathrm{T}}\mathbf{S}_2^{\mathrm{T}}\mathbf{V}^{-1}, \\ \mathbf{z} &= -\mathbf{c}_{12}\left(\tilde{\mathcal{X}}_1, \tilde{\mathcal{X}}_2\right) + \mathbf{S}_1\mathbf{J}_1\left(\hat{\mathcal{X}}_1 \ominus \tilde{\mathcal{X}}_1\right) \\ &\quad + \mathbf{S}_2\mathbf{J}_2\left(\hat{\mathcal{X}}_2 \ominus \tilde{\mathcal{X}}_2\right), \\ \mathbf{V} &= \mathbf{\Psi} + \mathbf{S}_1\mathbf{J}_1\tilde{\mathbf{P}}_1\mathbf{J}_1^{\mathrm{T}}\mathbf{S}_1^{\mathrm{T}} + \mathbf{S}_2\mathbf{J}_2\tilde{\mathbf{P}}_2\mathbf{J}_2^{\mathrm{T}}\mathbf{S}_2^{\mathrm{T}}, \end{aligned} \tag{16}$$

where iteration is done with $\hat{\mathcal{X}}_i \leftarrow \hat{\mathcal{X}}_i \oplus \delta\hat{\mathbf{x}}_i$, after initialization with $\hat{\mathcal{X}}_i \leftarrow \tilde{\mathcal{X}}_i$, until a convergence condition is met, such as $\delta\hat{\mathbf{x}}_i$ being sufficiently small. The marginal posterior covariances of Robots 1 and 2 are obtained from the corresponding diagonal blocks of $(\mathbf{H}^{\mathrm{T}}\mathbf{W}\mathbf{H})^{-1}$, and can be shown to be

$$\begin{aligned} \hat{\mathbf{P}}_1 &= (\mathbf{1} - \mathbf{K}_1\mathbf{S}_1)\mathbf{J}_1\tilde{\mathbf{P}}_1\mathbf{J}_1^{\mathrm{T}}, \\ \hat{\mathbf{P}}_2 &= (\mathbf{1} - \mathbf{K}_2\mathbf{S}_2)\mathbf{J}_2\tilde{\mathbf{P}}_2\mathbf{J}_2^{\mathrm{T}}. \end{aligned} \tag{17}$$

The above fusion step introduces cross-correlations between the state estimates of $\mathcal{X}_1$ and $\mathcal{X}_2$, and hence require a covariance intersection step

$$\hat{\mathbf{P}}_1 \leftarrow \frac{1}{w}\hat{\mathbf{P}}_1, \hat{\mathbf{P}}_2 \leftarrow \frac{1}{1-w}\hat{\mathbf{P}}_2, w \in (0,1).$$

The next step is to make the approximation that $p(\mathcal{X}_{1:N}|\mathbf{y}_{12}) \approx \prod_{i=1}^{N}\mathcal{N}_L(\hat{\mathcal{X}}_i, \hat{\mathbf{P}}_i)$, and proceed with the fusion of a second pseudomeasurement, using $p(\mathcal{X}_{1:N}|\mathbf{y}_{12}, \mathbf{y}_{13}) = \eta p(\mathbf{y}_{13}|\mathcal{X}_1, \mathcal{X}_3)p(\mathcal{X}_{1:N}|\mathbf{y}_{12})$. This new posterior can again be approximated as Gaussian using the expressions in (16) and (17), and the process is repeated until all pseudomeasurements are incorporated. The order in which the pseudomeasurements are fused is arbitrary and will correspond to the times at which information is shared between robots.

If only one iteration is performed, equations (16) and (17) simplify to the on-manifold EKF equations since $\mathbf{J}_i(\tilde{\mathcal{X}}_i \ominus \tilde{\mathcal{X}}_i) = \mathbf{J}_i(\mathbf{0}) = \mathbf{1}$. Much like the EKF is often sufficient compared to the iterated EKF, performing a single iteration in this multi-robot case is also sufficient for some problems. Algorithm 1 summarizes the general-purpose decentralized estimation algorithm from the point of view of an arbitrary robot. The algorithm is presented in a callback format, describing how the current state estimate is updated when various events occur.

Algorithm 1 Decentralized estimation with no input sharing.

For Robot i with process and measurement models given by (1), the following points break down how to compute the estimate when various events occur. Robot i's estimate is initialized with $\check{\mathcal{X}}_{i_0}, \check{\mathbf{P}}_{i_0}$. The matrices $\mathbf{F}_{i_k}, \mathbf{L}_{i_k}$ are the Jacobians of $\mathbf{f}$ with respect to $\mathcal{X}_{i_k}$ and $\mathbf{w}_{i_k}$, respectively. The matrix $\mathbf{G}_{i_k}$ is the Jacobian of $\mathbf{g}$, and $\mathbf{S}_{i_k}, \mathbf{S}_{j_k}$ are the Jacobians of $\mathbf{c}_{ij}$ with respect to $\mathcal{X}_{i_k}, \mathcal{X}_{j_k}$, respectively.

- On the reception of an input measurement $\mathbf{u}_{i_{k-1}}$:
$$\check{\mathcal{X}}_{i_k} = \mathbf{f}(\hat{\mathcal{X}}_{i_{k-1}}, \mathbf{u}_{i_{k-1}}, \mathbf{0}),$$
$$\check{\mathbf{P}}_{i_k} = \mathbf{F}_{i_{k-1}}\hat{\mathbf{P}}_{i_{k-1}}\mathbf{F}_{i_{k-1}}^{\mathsf{T}} + \mathbf{L}_{i_{k-1}}\mathbf{Q}_{k-1}\mathbf{L}_{i_{k-1}}^{\mathsf{T}}.$$
- On the reception of a local measurement $\mathbf{y}_{i_k}$:
$$\begin{aligned} \mathbf{K} &= \check{\mathbf{P}}_{i_k}\mathbf{G}_{i_k}^{\mathsf{T}}(\mathbf{G}_{i_k}\check{\mathbf{P}}_{i_k}\mathbf{G}_{i_k}^{\mathsf{T}} + \mathbf{R}_{i_k})^{-1}, \\ \delta\tilde{\mathbf{x}} &= \mathbf{K}(\mathbf{y}_{i_k} - \mathbf{g}(\check{\mathcal{X}}_{i_k})), \\ \tilde{\mathcal{X}}_{i_k} &= \check{\mathcal{X}}_{i_k} \oplus \delta\tilde{\mathbf{x}}, \\ \tilde{\mathbf{P}}_{i_k} &= (\mathbf{1} - \mathbf{K}\mathbf{G}_{i_k})\check{\mathbf{P}}_{i_k}. \end{aligned}$$
- On the reception of a neighbors' state estimate $\tilde{\mathcal{X}}_{j_k}, \tilde{\mathbf{P}}_{j_k}$:
$$\tilde{\mathbf{P}}_{i_k} \leftarrow \frac{1}{w}\tilde{\mathbf{P}}_{i_k}, \qquad \tilde{\mathbf{P}}_{j_k} \leftarrow \frac{1}{1-w}\tilde{\mathbf{P}}_{j_k},$$
$$\begin{aligned} \mathbf{K} &= \tilde{\mathbf{P}}_{i_k}\mathbf{S}_{i_k}^{\mathsf{T}}(\boldsymbol{\Psi} + \mathbf{S}_{i_k}\tilde{\mathbf{P}}_{i_k}^{\mathsf{T}}\mathbf{S}_{i_k}^{\mathsf{T}} + \mathbf{S}_{j_k}\tilde{\mathbf{P}}_{j_k}^{\mathsf{T}}\mathbf{S}_{j_k}^{\mathsf{T}})^{-1}, \\ \delta\hat{\mathbf{x}} &= -\mathbf{K}(\mathbf{c}_{ij}(\tilde{\mathcal{X}}_{i_k}, \tilde{\mathcal{X}}_{j_k})), \\ \hat{\mathcal{X}}_{i_k} &= \tilde{\mathcal{X}}_{i_k} \oplus \delta\hat{\mathbf{x}}, \\ \hat{\mathbf{P}}_{i_k} &= (\mathbf{1} - \mathbf{K}\mathbf{G}_{i_k})\tilde{\mathbf{P}}_{i_k}, \end{aligned}$$
and optionally further iterate *both* robot estimates with (16), (17).
- At any time, send the current state estimate $\tilde{\mathcal{X}}_{j_k}, \tilde{\mathbf{P}}_{j_k}$ to neighbors.

A feature of Algorithm 1 is that the robots can share their state information at anytime, with performance improving the more often sharing occurs, at the cost of increased communication bandwidth. After states are shared and pseudomeasurements are fused, the common states between robots will naturally drift due to sensor noise, until the next pseudomeasurement re-synchronizes the common states. This therefore becomes a tunable trade-off between estimation accuracy and communication bandwidth, which needs to be evaluated for a specific problem. In this paper's experiments, sharing is done at a set frequency of 10 Hz. Regarding computational complexity, Algorithm 1 will share a nearly identical runtime to a Kalman filter.

5.1. An observability test

In a decentralized state estimation context, observability refers to the ability for *each* robot to uniquely determine their state trajectory, given the inputs and measurements obtained by *all* robots. Determining observability for a decentralized estimator is non-trivial due to the need to capture the communication topology within the test itself. To illustrate this, consider again the linear toy problem with two robots from Section 4. A naive approach to determining observability would be to construct one "total state" $\mathbf{x} = [\mathbf{x}_1 \;\; \mathbf{x}_2]^{\mathsf{T}}$, and to perform a standard observability test on this augmented system using the collected sensor measurements from both robots y_1, y_2. However, such a test would falsely conclude that the system is unobservable, whereas the results from Section 4 clearly show successful estimation. The pseudomeasurement must additionally be incorporated into the test to give the correct result, as the robots are reliant on communication to attain observability of their individual states. To the best of the authors' knowledge, existing observability tests do not take into the full problem scope that this paper is concerned with. A decentralized observability test is presented in Pilloni et al. (2013) for linear-time-invariant systems with unknown input, but assumes each node is observable. The observability analysis in Huang et al. (2011) has a similar approach to this paper, but is specific to their system where robots have identical state definitions.

An advantage of the proposed approach is that the effects of communication of observability can be accurately captured by incorporating the pseudomeasurements themselves into a standard observability test. Concretely, for nonlinear systems, a *local* observability test can be formed by considering the MAP problem on an entire trajectory simultaneously, but without prior information on the initial state (Psiaki (2013)). Applying this to the multi-robot system, let the bolded $\boldsymbol{\mathcal{X}}_k = (\mathcal{X}_{1_k}, \ldots, \mathcal{X}_{N_k})$ denote the "total state" of all robots at time step k, and $\mathbf{y}_k = [\mathbf{y}_{1_k}^{\mathsf{T}} \;\; \ldots \;\; \mathbf{y}_{N_k}^{\mathsf{T}}]^{\mathsf{T}}$ denote a stacked vector containing all the robots' local measurements at time step k. Let $\boldsymbol{\psi}_k = [\;\ldots\;\; \mathbf{y}_{ij}^{\mathsf{T}} \;\;\ldots\;]^{\mathsf{T}}, (i,j) \in \mathcal{E}$ denote the stacked pseudomeasurements between all robots at time step k. The MAP problem is

$$\hat{\mathcal{X}}_{0:K} = \underset{\mathcal{X}_{0:K}}{\operatorname{argmax}}\, p(\mathcal{X}_{0:K}|\mathbf{y}_{0:K}, \boldsymbol{\psi}_{0:K})$$

with

$$p(\mathcal{X}_{0:K}|\mathbf{y}_{0:K}, \boldsymbol{\psi}_{0:K}) = \eta \prod_{k=0}^{K} p(\mathbf{y}_0|\mathcal{X}_0) p(\boldsymbol{\psi}_0|\mathcal{X}_0) \prod_{k=1}^{K} p(\mathcal{X}_k|\mathcal{X}_{k-1}),$$

leading to a nonlinear least-squares problem with weight $\mathbf{W}$ and error vector

$$\begin{aligned}
\mathbf{e}(\mathcal{X}_{0:K}) &= \left[\ \dots\ \mathbf{e}_{u,k}\ \dots\ \mathbf{e}_{y,k}\ \dots\ \mathbf{e}_{\psi,k}\ \dots\right]^{\mathrm{T}},\\
\mathbf{e}_{u,k} &= \left[\dots(\mathcal{X}_{i_k} \ominus \mathbf{f}(\mathcal{X}_{i_{k-1}}, \mathbf{u}_{i_{k-1}}))^{\mathrm{T}}\dots\right],\\
\mathbf{e}_{y,k} &= \left[\ \dots\ (\mathbf{y}_{i_k} - \mathbf{g}(\mathcal{X}_{i_k}))^{\mathrm{T}}\ \dots\ \right], i = 1, \dots, N,\\
\mathbf{e}_{\psi,k} &= \left[\ \dots\ -\mathbf{c}_{ij}(\mathcal{X}_{i_k}, \mathcal{X}_{j_k})^{\mathrm{T}}\ \dots\ \right], (i,j) \in \mathcal{E}.
\end{aligned}$$

The error Jacobian is

$$\mathbf{H} = \begin{bmatrix}
-\mathbf{F}_0 & \mathbf{1} & & \\
 & \ddots & \ddots & \\
 & & -\mathbf{F}_{K-1} & \mathbf{1}\\
-\mathbf{G}_0 & & & \\
 & -\mathbf{G}_1 & & \\
 & & \ddots & \\
 & & & -\mathbf{G}_K\\
-\boldsymbol{\Phi}_0 & & & \\
 & -\boldsymbol{\Phi}_1 & & \\
 & & \ddots & \\
 & & & -\boldsymbol{\Phi}_K
\end{bmatrix},$$

$$\begin{aligned}
\mathbf{F}_k &= \operatorname{diag}\left(\dots, \frac{D\mathbf{f}(\mathcal{X}_{i_k}, \mathbf{u}_{i_k})}{D\mathcal{X}_{i_k}}, \dots\right),\\
\mathbf{G}_k &= \operatorname{diag}\left(\dots, \frac{D\mathbf{g}(\mathcal{X}_{i_k})}{D\mathcal{X}_{i_k}}, \dots\right), i = 1, \dots, N,\\
\boldsymbol{\Phi}_k &= \frac{D\mathbf{e}_{\psi,k}(\mathcal{X}_k)}{D\mathcal{X}_k},
\end{aligned}$$

with all undisplayed entries in $\mathbf{H}$ equal to zero. For the solution to the MAP problem to be unique, then to first order, $(\mathbf{H}^{\mathrm{T}}\mathbf{W}\mathbf{H})$ must be invertible, and thus full rank. Fortunately, $\mathbf{W}$ is always positive definite regardless of any cross-correlations that would add off-diagonal entries. Hence,

$$\operatorname{rank}(\mathbf{H}^{\mathrm{T}}\mathbf{W}\mathbf{H}) = \operatorname{rank}(\mathbf{H}),$$

and it is thus required that $\mathbf{H}$ be full column rank. This implies that the proposed observability test is unaffected by the approximation induced by CI, or any cross-correlation terms that may or may not be successfully tracked. In a similar way to (Barfoot 2023, Ch 3.1.4), it can be shown that by performing a variety of elementary row/column operations, the rank of $\mathbf{H}$ is equivalent to the rank of

$$\mathcal{O} = \begin{bmatrix} \mathbf{M}_0 \\ \mathbf{M}_1\mathbf{F}_0 \\ \vdots \\ \mathbf{M}_K\mathbf{F}_{K-1}\dots\mathbf{F}_0 \end{bmatrix}, \mathbf{M}_k = \begin{bmatrix} \mathbf{G}_k \\ \boldsymbol{\Phi}_k \end{bmatrix}.$$

Hence, if $\mathcal{O}$ has maximum rank, the solution to the MAP problem is locally unique, and the system is said to be observable. Note that this test easily allows for time-varying graphs, which would yield a different $\boldsymbol{\Phi}_k$ for each time step k.

6. Efficient odometry sharing using preintegration

Many problems, especially those where robots estimate their neighbors' positions, will require robots to have access to their neighbors' process model input values $\mathbf{u}$. Previously in this paper, it has been assumed that all robots have unrestricted access to each other's inputs. In robot state estimation applications, the input is often the odometry measurements, such as wheel encoder or IMU measurements. These can occur at frequencies of 100–1000 Hz, and can therefore be infeasible to share in real time, especially if multiple robots are to simultaneously share measurements at high frequency. This could quickly reach a bandwidth limit on the common communication channel, such as ultra-wideband radio. While Xu et al. (2020) solve this problem by directly sharing pose changes between two points in time, this violates statistical independence assumptions and leads to inconsistent estimates.

The proposed solution to this problem is to use *preintegration*. That is, robots will instead share preintegrated input measurements over an arbitrary duration of time instead of individual input measurements. Specifically, consider the following generic process model

$$\mathcal{X}_k = \mathbf{f}(\mathcal{X}_{k-1}, \mathbf{u}_{k-1}, \mathbf{w}_{k-1}).$$

The action of preintegration is to directly iterate this process model by repeated compositions in order to, after algebraic manipulation, generate a new *preintegrated process model* $\mathbf{f}_{pq}$ that relates two states at arbitrary time steps $k = p$ and $k = q$. That is,

$$\begin{aligned}
\mathcal{X}_{p+1} &= \mathbf{f}(\mathcal{X}_p, \mathbf{u}_p, \mathbf{w}_p),\\
\mathcal{X}_{p+2} &= \mathbf{f}(\mathbf{f}(\mathcal{X}_p, \mathbf{u}_p, \mathbf{w}_p), \mathbf{u}_{p+1}, \mathbf{w}_{p+1}),\\
&\ \ \vdots\\
\mathcal{X}_q &= \mathbf{f}(\mathbf{f}(\dots\mathbf{f}(\mathcal{X}_p, \mathbf{u}_p, \mathbf{w}_p)\dots), \mathbf{u}_{q-1}, \mathbf{w}_{q-1})\\
&\triangleq \mathbf{f}_{pq}(\mathcal{X}_i, \Delta\mathcal{X}_{pq}) \oplus \mathbf{w}_{pq},
\end{aligned} \tag{18}$$

where $\Delta\mathcal{X}_{pq}$ is the *relative motion increment* (RMI), which in general may also belong to a Lie group, and $\mathbf{w}_{pq} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_{pq})$ is the *preintegrated noise*. The advantages of preintegration will stem from the careful choice of RMI definition, which is ideally done such that the RMI has the following properties.

1. The RMI is determined from the input measurements exclusively, and is independent of the state estimate:
$$\Delta\mathcal{X}_{pq} = \Delta\mathcal{X}_{pq}\big(\mathbf{u}_{p:q-1}\big).$$
2. Far fewer numbers are required to represent the RMI than the $(q-p)$ raw measurements that occurred during the preintegration interval:
$$\dim\big(\Delta\mathcal{X}_{pq}\big) \ll (q-p)\dim(\mathbf{u}_k).$$

If the above points are true, communicating $\Delta\mathcal{X}_{pq}, \mathbf{Q}_{pq}$ instead of $\mathbf{u}_{p:q-1}$ will not only reduce the communication cost, but will also result in a fixed message size and ability to directly predict the state forward over a long duration of time, instead of sequentially processing the measurements. Note that it is not always possible to define an RMI such that (18) holds exactly. However, it turns out that many common process models in robotics are amenable to preintegration (Barrau and Bonnabel (2019); Eckenhoff et al. (2019)), and furthermore are typically extremely fast to preintegrate incrementally as input measurements are obtained (Forster et al. (2017)). In other words, there exists a function $\textsc{Increment}(\cdot)$ defined such that it satisfies
$$\Delta\mathcal{X}_{pq}, \mathbf{Q}_{pq} = \textsc{Increment}\ (\Delta\mathcal{X}_{p:q-1}, \mathbf{Q}_{p:q-1}, \mathbf{u}_{q-1}).$$

A few examples now follow, which describe concrete implementations of $\Delta\mathcal{X}_{pq}$, $\mathbf{f}_{pq}(\cdot)$, and $\textsc{Increment}(\cdot)$.

Example 1. Linear preintegration. *The linear process model*
$$\mathbf{x}_k = \mathbf{F}_{k-1}\mathbf{x}_{k-1} + \mathbf{L}_{k-1}\mathbf{u}_{k-1}$$
can be directly iterated to yield
$$\begin{aligned}\mathbf{x}_q &= \left(\prod_{k=p}^{q-1}\mathbf{F}_k\right)\mathbf{x}_p + \sum_{k=p}^{q-1}\left(\prod_{\ell=k+1}^{q-1}\mathbf{F}_\ell\right)\mathbf{L}_k\mathbf{u}_k \\ &\triangleq \mathbf{F}_{pq}\mathbf{x}_p + \Delta\mathbf{x}_{pq},\end{aligned} \tag{19}$$
where (19) *defines* $\mathbf{f}(\cdot)$. *Assuming that noise enters the model additively through the input* $\mathbf{u}_k = \bar{\mathbf{u}}_k + \mathbf{w}_k$, *where* $\mathbf{w}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_k)$, (19) *becomes* $\mathbf{x}_j = \mathbf{F}_{pq}\mathbf{x}_i + \Delta\bar{\mathbf{x}}_{pq} + \mathbf{w}_{pq}$ *where*
$$\begin{aligned}\mathbf{w}_{pq} &\triangleq \sum_{k=p}^{q-1}\left(\prod_{\ell=k+1}^{q-1}\mathbf{F}_\ell\right)\mathbf{L}_k\mathbf{w}_k, \\ &= \mathbf{F}_{q-1}\mathbf{w}_{pq-1} + \mathbf{L}_{q-1}\mathbf{w}_{q-1}.\end{aligned}$$
The RMI $\Delta\mathbf{x}_{pq}$ *and corresponding covariance are therefore built incrementally with*
$$\begin{aligned}\Delta\mathbf{x}_{pq} &= \mathbf{F}_{q-1}\Delta\mathbf{x}_{pq-1} + \mathbf{L}_{q-1}\mathbf{u}_{q-1}, \\ \mathbf{Q}_{pq} &= \mathbf{F}_{q-1}\mathbf{Q}_{pq-1}\mathbf{F}_{q-1}^{\mathrm{T}} + \mathbf{L}_{q-1}\mathbf{Q}_{q-1}\mathbf{L}_{q-1,}^{\mathrm{T}}\end{aligned}$$
which together define the $\textsc{Increment}(\cdot)$ *function.*

Example 2. Wheel odometry preintegration on $SE(2)$. *Given a robot pose* $\mathbf{T} \in SE(2)$, *the wheel odometry process model is given by*
$$\mathbf{T}_k = \mathbf{T}_{k-1}\,\mathrm{Exp}(\Delta t\mathbf{u}_{k-1}),$$
where $\mathbf{u} = [\omega\ v\ 0]^{\mathrm{T}}$, ω *is the robot's heading rate-of-change, and* v *is its forward velocity in its own body frame. Direct iteration yields the preintegrated process model* $\mathbf{f}_{pq}(\cdot)$ *given by*
$$\mathbf{T}_q = \mathbf{T}_p\underbrace{\prod_{k=p}^{q-1}\mathrm{Exp}(\Delta t\mathbf{u}_k)}_{\triangleq\Delta\mathbf{T}_{pq}}.$$
Noise $\mathbf{w}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_k)$ *is again assumed to enter additively through the input, and a series of first-order approximations lead to*
$$\begin{aligned}\Delta\mathbf{T}_{pq} &= \prod_{k=p}^{q-1}\mathrm{Exp}(\Delta t(\bar{\mathbf{u}}_k + \mathbf{w}_k)) \\ &\approx \prod_{k=p}^{q-1}\mathrm{Exp}(\Delta t\bar{\mathbf{u}}_k)\mathrm{Exp}(\Delta t\mathbf{J}_k\mathbf{w}_k) \\ &\approx \Delta\bar{\mathbf{T}}_{pq}\underbrace{\prod_{k=p}^{q-1}\mathrm{Exp}\Big(\Delta t\mathbf{Ad}\Big(\Delta\mathbf{T}_{k+1j}^{-1}\Big)\mathbf{J}_k\mathbf{w}_k\Big)}_{\mathrm{Exp}\left(\mathbf{w}_{pq}\right)},\end{aligned}$$
where $\mathbf{J}_k \triangleq \mathbf{J}(\Delta t\bar{\mathbf{u}}_k)$ *is the right Jacobian of* $SE(2)$. *Having identified an expression for* $\mathrm{Exp}(\mathbf{w}_{pq})$, *under the assumption that* $\mathbf{w}_{pq}$ *is small,*
$$\begin{aligned}\mathbf{w}_{pq} &\approx \sum_{k=p}^{q-1}\Delta t\mathbf{Ad}\Big(\Delta\mathbf{T}_{k+1q}^{-1}\Big)\mathbf{J}_k\mathbf{w}_k \\ &= \sum_{k=p}^{q-2}\Delta t\mathbf{Ad}\Big(\Delta\mathbf{T}_{k+1q}^{-1}\Big)\mathbf{J}_k\mathbf{w}_k \\ &\quad + \Delta t\underbrace{\mathbf{Ad}\Big(\Delta\mathbf{T}_{qq}^{-1}\Big)}_{\mathbf{1}}\mathbf{J}_{q-1}\mathbf{w}_{q-1} \\ &= \underbrace{\mathbf{Ad}\Big(\Delta\mathbf{T}_{q-1q}^{-1}\Big)}_{\triangleq\mathbf{F}_{q-1}}\mathbf{w}_{pq-1} + \underbrace{\Delta t\mathbf{J}_{q-1}}_{\triangleq\mathbf{L}_{q-1}}\mathbf{w}_{q-1},\end{aligned}$$
and the defining operations of the $\textsc{Increment}(\cdot)$ *function follow,*
$$\begin{aligned}\Delta\mathbf{T}_{pq} &= \Delta\mathbf{T}_{pq-1}\,\mathrm{Exp}(\Delta t\mathbf{u}_k), \\ \mathbf{Q}_{pq} &= \mathbf{F}_{q-1}\mathbf{Q}_{pq-1}\mathbf{F}_{q-1}^{\mathrm{T}} + \mathbf{L}_{q-1}\mathbf{Q}_{q-1}\mathbf{L}_{q-1}^{\mathrm{T}}.\end{aligned}$$

Example 3. IMU preintegration. *Being the most well-known usage of preintegration, a complete reference for IMU preintegration on the* $SO(3)\times\mathbb{R}^3\times\mathbb{R}^3$ *manifold can be obtained from* Forster et al. (2017), *and alternatively for the* $SE_2(3)$ *group from* Brossard et al. (2022); Barfoot (2023). *Either approach can be used with the framework in this paper. However in Section 8 of this* *paper,* $\mathbf{T}_{wi} \in SE_2(3)$ *matrices are used to represent the extended pose of Robot i relative to an inertial world frame w. Following* Shalaby et al. (2023) and Brossard et al. (2022)*, it can be shown that the discrete-time IMU kinematic equations can be written in the form*

$$\mathbf{T}_{wi_k} = \mathbf{G}_{k-1}\mathbf{T}_{wi_{k-1}}\mathbf{U}_{k-1}, \tag{20}$$

where

$$\begin{aligned}
\mathbf{T}_{wi_k} &= \begin{bmatrix} \mathbf{C} & \mathbf{v} & \mathbf{r} \\ \mathbf{0} & 1 & 0 \\ \mathbf{0} & 0 & 1 \end{bmatrix} \\
\mathbf{G}_{k-1} &= \begin{bmatrix} \mathbf{1} & \Delta t\mathbf{g} & -\dfrac{\Delta t^2}{2}\mathbf{g} \\ 0 & 1 & -\Delta t \\ 0 & 0 & 1 \end{bmatrix} \\
\mathbf{U}_{k-1} &= \begin{bmatrix} \exp(\Delta t\boldsymbol{\omega}^\wedge) & \Delta t\mathbf{J}(\Delta t\boldsymbol{\omega})\mathbf{a} & \dfrac{\Delta t^2}{2}\mathbf{N}(\Delta t\boldsymbol{\omega})\mathbf{a} \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{bmatrix} \\
\mathbf{N}(\boldsymbol{\phi}) &= \mathbf{z}\mathbf{z}^\mathsf{T} + 2\left(\frac{1}{\phi} - \frac{\sin\phi}{\phi^2}\right)\mathbf{z}^\wedge + 2\frac{\cos\phi - 1}{\phi^2}\mathbf{z}^\wedge\mathbf{z}^\wedge \\
\phi &= \|\boldsymbol{\phi}\|, \ \mathbf{z} = \boldsymbol{\phi}/\phi
\end{aligned}$$

and $\mathbf{C} \in SO(3)$, $\mathbf{v}$, $\mathbf{r}$ *respectively represent attitude, velocity, position relative to the world frame w,* $\boldsymbol{\omega}$ *is the IMU's unbiased gyro measurement,* $\mathbf{a}$ *is the IMU's unbiased accelerometer measurement,* $\mathbf{g}$ *is the gravity vector resolved in frame w, and* $\mathbf{J}(\boldsymbol{\phi})$ *is the left Jacobian of* $SO(3)$*. Preintegration of these kinematics is easily achieved by direct iteration with*

$$\begin{aligned}
\mathbf{T}_{wi_q} &= \left(\prod_{k=p}^{q-1}\mathbf{G}_{k-1}\right)\mathbf{T}_{wi_p}\left(\prod_{k=p}^{q-1}\mathbf{U}_{k-1}\right) \\
&\triangleq \Delta\mathbf{G}_{pq}\mathbf{T}_{wi_p}\Delta\mathbf{U}_{pq}
\end{aligned}$$

where $\Delta\mathbf{U}_{pq}$ *is the RMI, and the noise statistics can be propagated through this preintegration process by a standard linearization procedure.* Shalaby et al. (2023) *present the details of the noise propagation, as well as how to adapt this formulation for relative poses.*

6.1. Multi-robot preintegration

In the context of multi-robot estimation problems, an individual robot's process model may involve the input values of many neighboring robots. To reflect this, rewrite the process model for Robot i as

$$\mathcal{X}_{i_k} = \mathbf{f}\left(\mathcal{X}_{i_{k-1}}, \mathbf{u}_{i_{k-1}}, \mathbf{u}_{j_{k-1}}\right), \ j \in \mathcal{N}_i, \tag{21}$$

where $\mathbf{u}_{i_k}$ denotes an input measured by Robot i and $\mathcal{N}_i$ denotes the set of neighbor IDs of Robot i. The preintegrated process model would now be written as

$$\mathcal{X}_{i_q} = \mathbf{f}_{pq}\left(\mathcal{X}_{i_p}, \Delta\mathcal{X}_{i_{pq}}, \Delta\mathcal{X}_{j_{pq}}\right), \ j \in \mathcal{N}_i \tag{22}$$

where $\Delta\mathcal{X}_{i_{pq}}$ denotes an RMI calculated from the input measurements of Robot i.

A complication is that the RMIs from neighboring Robots $\Delta\mathcal{X}_{j_{pq}}$ are only available asynchronously, meaning it is not always possible to evaluate (22) directly. To deal with this, assume that the state can be temporarily partially propagated with a value of $\mathbf{0}$ for the neighbor's input, and then fully propagated once the RMI is shared. That is, assume the preintegrated process model $\mathbf{f}_{pq}$ is compatible with

$$\begin{aligned}
\mathfrak{X}_{i_{k-1}} &= \mathbf{f}(\mathcal{X}_{i_{k-2}}, \mathbf{u}_{i_{k-2}}, \mathbf{0}), \\
\mathfrak{X}_{i_k} &= \mathbf{f}(\mathfrak{X}_{i_{k-1}}, \mathbf{u}_{i_{k-1}}, \mathbf{0}), \\
\mathcal{X}_{i_k} &= \mathbf{f}_{pq}\left(\mathfrak{X}_{i_k}, \mathcal{I}, \Delta\mathcal{X}_{j_{pq}}\right),
\end{aligned}$$

where $\mathcal{I}$ is the "identity" or "zero" RMI constructed from input values of zero, and $\mathbf{f}$ is defined as per (21) with $\mathbf{u}_j$ substituted for $\mathbf{0}$. The variable $\mathfrak{X}_{i_k}$ represents an intermediate, non-physical state that is propagated using the process model without input information from neighboring robots. Sometime later, at arbitrary time step $k = q$, the RMI from a neighboring robot $\Delta\mathcal{X}_{j_{pq}}$ is received and this intermediate state $\mathfrak{X}_{i_q}$ is propagated back into a physically meaningful quantity $\mathcal{X}_{i_q}$. A concrete example of this asynchronous intermediate state updating is shown in Section 7, and a summary is shown in Algorithm 2.

Algorithm 2 Decentralized estimation with preintegration.

The setup is identical to Algorithm 1, except that the process model requires input information from neighboring robots as in (21), (22).

- On the reception of a local input measurement $\mathbf{u}_{i_{k-1}}$:

$$\begin{aligned}
\mathfrak{X}_{i_k} &= \mathbf{f}(\hat{\mathcal{X}}_{i_{k-1}}, \mathbf{u}_{i_{k-1}}, \mathbf{0}) \\
\mathfrak{P}_{i_k} &= \mathbf{F}_{i_{k-1}}\hat{\mathbf{P}}_{i_{k-1}}\mathbf{F}_{i_{k-1}}^\mathsf{T} + \mathbf{L}_{i_{k-1}}\mathbf{Q}_{k-1}\mathbf{L}_{i_{k-1}}^\mathsf{T},
\end{aligned}$$

  and increment Robot i's own RMI,

$$\Delta\mathcal{X}_{i_{pk}}, \mathbf{Q}_{i_{pk}} = \textsc{Increment}(\Delta\mathcal{X}_{i_{pk-1}}, \mathbf{Q}_{i_{pk-1}}, \mathbf{u}_{i_{k-1}}).$$

- Whenever required, send $\Delta\mathcal{X}_{i_{pq}}, \mathbf{Q}_{i_{pq}}$ to neighbors.
- On the reception of a neighboring robot's RMI and covariance $\Delta\mathcal{X}_{j_{pq}}, \mathbf{Q}_{j_{pq}}$,

$$\begin{aligned}
\mathcal{X}_{i_q} &= \mathbf{f}_{pq}(\mathfrak{X}_{i_q}, \mathcal{I}, \Delta\mathcal{X}_{j_{pq}}), \\
\mathbf{F}_{pq} &= \left.\frac{D\mathbf{f}_{pq}(\mathfrak{X}, \mathcal{I}, \Delta\mathcal{X}_{j_{pq}})}{D\mathfrak{X}}\right|_{\mathfrak{X}_{i_q}}, \\
\check{\mathbf{P}}_{i_q} &= \mathbf{F}_{pq}\mathfrak{P}_{i_{k-1}}\mathbf{F}_{pq}^\mathsf{T} + \mathbf{Q}_{j_{pq}}.
\end{aligned}$$

- On the reception of a local measurement $\mathbf{y}_{i_k}$, proceed as per Algorithm 1.
- On the reception of a neighbors' state estimate $\tilde{\mathcal{X}}_j, \tilde{\mathbf{P}}_j$, proceed as per Algorithm 1.
- At any time, send the current state estimate $\tilde{\mathcal{X}}_{j_k}, \tilde{\mathbf{P}}_{j_k}$ to neighbors.

6.2 Estimating input biases

For some problems, it may be desired to estimate an input bias $\mathbf{b}$ as part of the overall state $\mathcal{X}$, (a) setup commonly occurring in inertial navigation where accelerometer and rate gyro biases are estimated. The difficulty lies in the frequent inability to express RMIs independently of the bias values, thus leaving RMIs in the form of

$$\Delta\mathcal{X}_{pq}\left(\mathbf{u}_{p:q-1}, \mathbf{b}_{p:q-1}\right)$$

In the context of the multi-robot estimation scheme shown in Algorithm 2, computing RMIs this way causes inconsistency in the filter, since the RMIs are now correlated with the robot states. Accounting for this would require maintaining the cross-correlation between a robot's state and their neighbors' biases.

A simpler alternate solution is to have robots estimate their neighbors' input biases in addition to their own. This requires to exploit the fact that biases are usually modeled to follow a random walk, and therefore have a constant mean in the absence of any correcting information. This motivates the approximation $\mathbf{b}_p \approx \mathbf{b}_{p+1} \approx \ldots \approx \mathbf{b}_q$ and hence

$$\Delta\mathcal{X}_{pq}\left(\mathbf{u}_{p:q-1}, \mathbf{b}_{p:q-1}\right) \approx \Delta\mathcal{X}_{pq}\left(\mathbf{u}_{p:q-1}, \mathbf{b}_q\right).$$

When robots receive input measurements, they increment their RMIs with raw (biased) inputs to produce biased RMIs $\Delta\mathcal{X}_{pq}(\mathbf{u}_{p:q-1}, \mathbf{0})$. At an appropriate time, they share their current biased RMIs, which is corrected for bias by the receiving robot using the first-order approximation

$$\begin{aligned} \Delta\mathcal{X}_{pq}\left(\mathbf{u}_{p:q-1}, \mathbf{b}_q\right) &\approx \Delta\mathcal{X}_{pq}\left(\mathbf{u}_{p:q-1}, \mathbf{0}\right) \oplus \mathbf{B}_{pq}\mathbf{b}_q, \\ \mathbf{B}_{pq} &\triangleq \frac{D}{D\mathbf{b}_q}\left(\Delta\mathcal{X}_{pq}\left(\mathbf{u}_{p:q-1}, \mathbf{b}_q\right)\right), \end{aligned} \tag{23}$$

where $\mathbf{B}_{pq}$ is defined as the bias Jacobian. Equation (23) is an approximation that contributes unmodeled errors to the estimation problem, relying on an assumption that $\mathbf{b}_q$ is small. This can be enabled by a proper offline calibration procedure that removes any large bias, and only small deviations from this are estimated online. Such a procedure is used for the quadcopter problem in Section 8, where good, consistent estimation results are still obtained despite the approximation in (23).

6.3. Autoencoding covariance matrices

As shown in Algorithm 2, predicting state estimates that are a function of neighbor inputs requires the RMI $\Delta\mathcal{X}_{pq}$ along with a corresponding covariance $\mathbf{Q}_{pq}$. As is, these two quantities must be shared between robots. This section proposes an optional method that further reduces communication costs by eliminating the requirement to share the preintegrated covariance $\mathbf{Q}_{pq}$.

The key insight is that $\Delta\mathcal{X}_{pq}(\mathbf{u}_{p:q-1})$ and $\mathbf{Q}_{pq}(\mathbf{u}_{p:q-1})$ are both calculated from the same input values $\mathbf{u}_{p:q-1}$. Hence, if an alternate mapping $\mathbf{Q}_{pq} = \mathbf{h}(\Delta\mathcal{X}_{pq})$ existed, then it would be sufficient to share $\Delta\mathcal{X}_{pq}$ only, and the receiving robot could infer $\mathbf{Q}_{pq}$ directly from the RMI. In the absence of analytic expressions for $\mathbf{h}(\cdot)$, this paper approximates the function with a neural network, trained on purely synthetic RMI covariance pairs. An additional complication is that such a function $\mathbf{h}(\cdot)$ may not always exist since an RMI can correspond to many possible covariances depending on the input values. In this case $\mathbf{h}(\cdot)$ is not a true function since it is one-to-many, and its definition is modified to also accept a low-dimensional *encoding* $\boldsymbol{e}(\mathbf{Q}_{pq})$, leading to $\mathbf{Q}_{pq} = \mathbf{h}(\Delta\mathcal{X}_{pq}, \boldsymbol{e}(\mathbf{Q}_{pq}))$. This leads to an architecture here referred to as *mean-assisted autoencoding*, depicted in Figure 4.

The flattened lower-triangular half of $\mathbf{Q}_{pq}$ is given to a simple fully connected *encoder* network with GELU activation functions and a single hidden layer with 256 neurons. The output of this network is the encoding, which can be as small as one or two numbers. This encoding is then concatenated with a parameterization of the RMI $\Delta\mathcal{X}_{pq}$ and fed to a similar *decoder* network, again with a single 256-neuron hidden layer. The decoder network outputs the flattened lower-triangular half of a Cholesky decomposition $\mathbf{L}$, which is used to reconstruct the matrix using $\widehat{\mathbf{Q}}_{pq} = \mathbf{L}\mathbf{L}^{\mathsf{T}}$. The covariance matrix is guaranteed to be positive definite as long as the diagonal elements of $\mathbf{L}$ are non-zero, which is extremely unlikely to occur in practice. For training, the loss function simply uses the Frobenius norm,

$$\mathcal{L}\left(\mathbf{Q}_{pq}, \widehat{\mathbf{Q}}_{pq}\right) = \left\|\mathbf{Q}_{pq} - \widehat{\mathbf{Q}}_{pq}\right\|_{\mathrm{F}}.$$

Figure 5 shows the training convergence history for various encoding sizes, applied to IMU preintegration. The Adam optimizer is chosen with an initial learning rate of 10^{-3} that

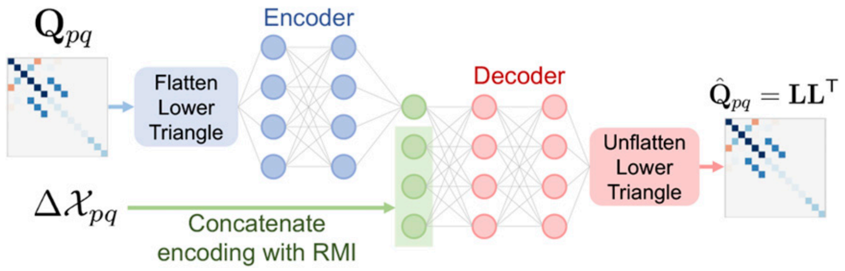

Figure 4. Concept diagram of mean-assisted autoencoder.

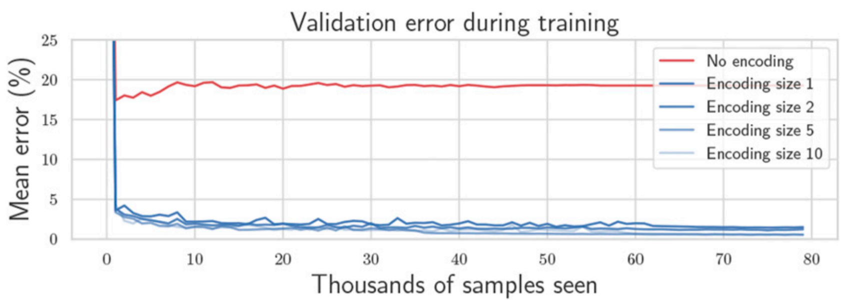

Figure 5. Mean percentage reconstruction error throughout training for various encoding sizes including no encoding. A single encoding number is sufficient to achieve less than 1% reconstruction error on average.

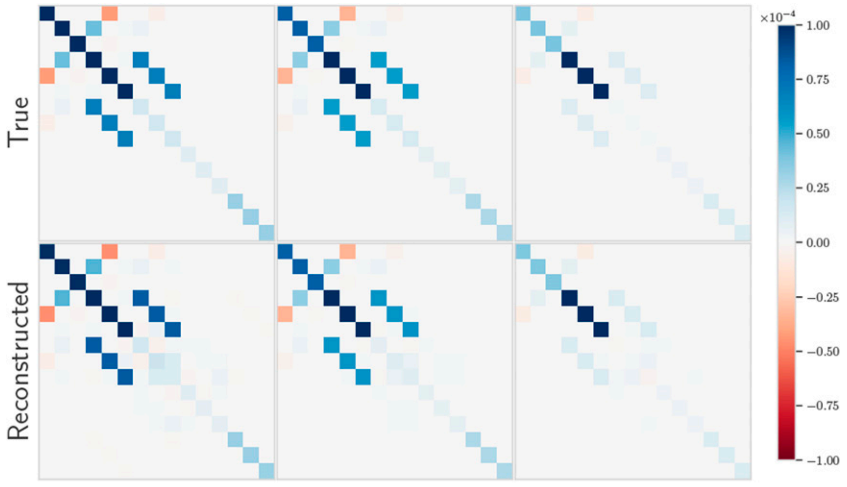

Figure 6. Visualization of preintegrated IMU noise covariance matrices along with reconstruction using mean-assisted autoencoding.

is scheduled to decrease once the loss plateaus. The training process takes just a few minutes on a laptop CPU to achieve less than 1% average reconstruction error on a validation dataset from experimental data. The "No encoding" baseline represents an architecture where the decoder network attempts to infer the covariance matrix directly from the RMI itself, without using the output of the encoder network, and hence eliminating the need to communicate RMI covariance information. However, doing this yields substantially higher error than when an encoding is provided. A visualization of the reconstructed covariance matrices can be seen in Figure 6, using an encoding size of only one number. Section 8 will employ this method "in the loop" for a real quadcopter problem. It will be shown that the reconstruction error is so small that the impact on the estimation results are negligible.

The training data is purely synthetic, where RMIs are constructed from a random amount of random IMU measurements, with values covering the realistic range of real IMU measurements. Since the length of the dataset is infinite, the risk of overfitting is completely eliminated, as long as the real IMU measurements lie within the range of randomly generated values. In fact, the networks immediately generalize to any physical sensor of the same type. Physical characteristics such as biases, scale factors, and axis-misalignments are irrelevant since the result after these effects is still a list of values representing the sensor measurements. As long as those values remain within the randomly generated training domain, the network will perform well. In Section 8, the same autoencoder is used on three different quadcopters each with different physical IMUs.

Concretely, using IMU preintegration as an example, the RMI itself must be communicated, which requires 10 floating-point numbers. However, the covariance matrix is 15×15, which would require communication of an additional 120 floating-point numbers to represent one of its triangular halves. With the proposed autoencoder, these 120 numbers are replaced with an encoding consisting of *one* number, thus dramatically reducing the communication cost. Figure 7 shows how the required communication rate varies with the duration between two successive communications between two arbitrary robots. The naive solution without preintegration requires sharing all input measurements that have occurred during that period, whereas preintegration yields a constant message size. The proposed method can be applied to all problems discussed in the paper, where the networks must be trained for each problem.

7. Simulation with ground robots

The proposed algorithm is tested in a simulation with ground robots, shown in the center of Figure 1. Each robot estimates their own pose and their neighbors' poses relative to a world frame. Denoting the pose of Robot i relative to the world frame w as $\mathbf{T}_{wi} \in SE(2)$, the state of an arbitrary robot is given by

$$\mathcal{X}_{i_k} = \left(\mathbf{T}^{[i]}_{wi_k}, \mathbf{T}^{[i]}_{wj_k}, \dots\right), j \in \mathcal{N}_i,$$

where, again, the $(\cdot)^{[i]}$ superscript indicates Robot i's estimate or "instance" of that physical quantity. Each robot collects wheel odometry at 100 Hz, providing $\mathbf{u}_{i_k} = [\omega_{i_k} \;\; v_{i_k} \;\; 0]^{\mathrm{T}}$ as input measurements, where ω_{i_k} is Robot i's angular velocity and v_{i_k} is its forward velocity in its own body frame. The pose kinematics for any single robot along with its preintegration are shown in Example 2. When Robot i receives an input measurement, it updates the part of its state corresponding to its own pose to create

$$\mathfrak{X}_{i_k} = \left(\mathbf{T}^{[i]}_{iw_{k-1}} \operatorname{Exp}(\Delta t \mathbf{u}_{i_{k-1}}), \mathbf{T}^{[i]}_{wj_{k-1}}, \dots\right).$$

The neighbor poses $\mathbf{T}_{wj_{k-1}}$ are now out of date, as neighboring odometry information is not yet accessible to Robot i, and this partially out-of-date state is non-physical and given the symbol $\mathfrak{X}_{i_k}$. Every robot computes their own RMIs from wheel odometry using the equations from Example 2. When a neighbor's RMI $\Delta\mathbf{T}_{j_{pq}}$ is received at some later time step $k = q$, the state is updated with

$$\mathcal{X}_{i_q} = \left(\mathbf{T}^{[i]}_{iw_q}, \mathbf{T}^{[i]}_{wj_p}\Delta\mathbf{T}_{j_{pq}}, \dots\right)$$

where p represents the time step index of the last time a neighbor RMI was received.

Each robot also collects range measurements to its neighbors at 10 Hz, with the connectivity graph shown in Figure 1 (middle). Only two robots collect relative position measurements to known landmarks at 10 Hz. At an arbitrary separate frequency, each robot sends its current state and covariance to its neighbors, allowing the neighbors to compute pseudomeasurements of the form

$$\mathbf{c}_{ij}(\mathcal{X}_i, \mathcal{X}_j) = \begin{bmatrix} \operatorname{Log}\left(\mathbf{T}^{[i]^{-1}}_{wi}\mathbf{T}^{[j]}_{wi}\right) \\ \operatorname{Log}\left(\mathbf{T}^{[i]^{-1}}_{wj}\mathbf{T}^{[j]}_{wj}\right) \\ \operatorname{Log}\left(\mathbf{T}^{[i]^{-1}}_{w\ell}\mathbf{T}^{[j]}_{w\ell}\right) \\ \vdots \end{bmatrix}, \ell \in \mathcal{N}_i \cap \mathcal{N}_j.$$

A simulation is performed with four robots each executing Algorithm 2, with root-mean-squared error (RMSE) shown in Figure 8. The initial states are initialized to ground truth with some random error with covariance $\check{\mathbf{P}}_{i_0} = 0.1^2 \cdot \mathbf{1}$. The position and range measurements have Gaussian noise with 0.3 m and 0.1 m of standard deviation, respectively, and the

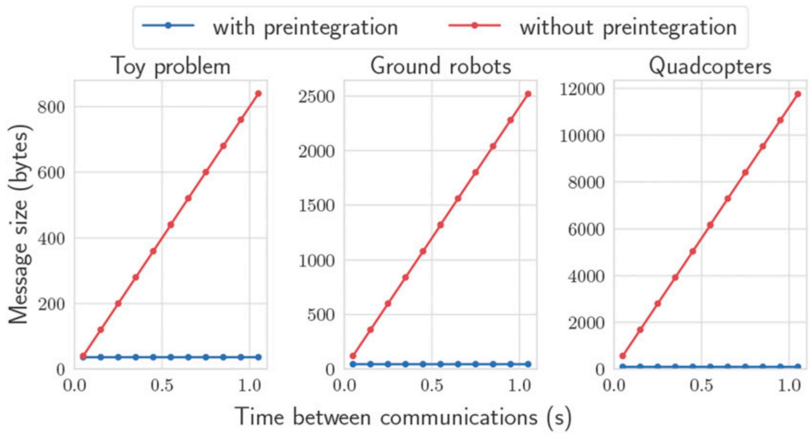

Figure 7. RMSE for the ground robot simulation. There are four blue lines for the four robots running the proposed algorithm, and four visibly coincident red lines for the naive algorithm. **Left:** State fusion occurring at 10 Hz. **Right:** State fusion occurring at 1 Hz.

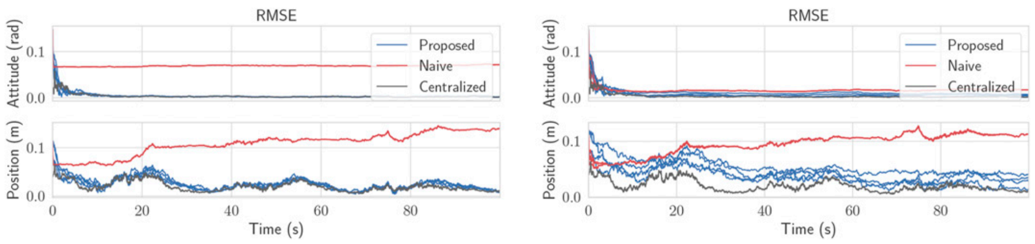

Figure 8. 50-trial NEES plot for the ground robot simulation for the proposed versus centralized solution, with the multiple blue lines each representing a robot. The naive solution without CI is far outside the plot. The red line represents the expected NEES value.

pseudomeasurement covariance is $\mathbf{\Psi} = \mathbf{0}$. The results show that all four robots' estimation errors successfully stabilize and remain low, despite only two robots having sufficient sensors to make their states observable. Similar to the Toy Problem, the naive solution is implemented which is identical to the proposed solution, but does not do a CI step. A 50-trial Monte Carlo simulation was also performed with the resulting average NEES plotted in Figure 9. The naive solution has significantly higher error than both the centralized or proposed algorithms and is so overconfident that it cannot be plotted within reasonable axis limits in Figure 9. Although, in theory, the proposed algorithm should have lower NEES values than the centralized solution, Figure 9 shows comparable values. This is suspected to be due to the high degree of nonlinearity of the problem.

As seen in Figure 8, if state fusion is done at a sufficiently high-frequency, performance is even comparable to the centralized estimator, but this will incur a larger communication and computation requirement as discussed in Section 5.

8. Simulation and experiments with quadcopters

To demonstrate the flexibility of the proposed framework, consider a new problem involving quadcopters. The kinematic state of each quadcopter is modeled using extended pose matrices $\mathbf{T} \in SE_2(3)$ (Brossard et al. (2022)). Each robot estimates both their absolute pose relative to the world frame $\mathbf{T}_{wi} \in SE_2(3)$, their own IMU bias $\mathbf{b}_i$, as well as the *relative* poses of their neighbors $\mathbf{T}_{ij} \in SE_2(3)$ and their IMU bias $\mathbf{b}_j$. The full state of Robot i is then given by

$$\mathcal{X}_i = \left(\mathbf{T}_{wi}, \mathbf{b}_i^{[i]}, \mathbf{T}_{ij}, \mathbf{b}_j^{[i]}, \ldots\right), j \in \mathcal{N}_i.$$

The pose of Robot j relative to Robot i $\mathbf{T}_{ij}$ has kinematics involving the IMU measurements of both robots, and are given in discrete time by

$$\mathbf{T}_{ij_k} = \mathbf{U}_{i_{k-1}}^{-1} \mathbf{T}_{ij_{k-1}} \mathbf{U}_{j_{k-1}},$$

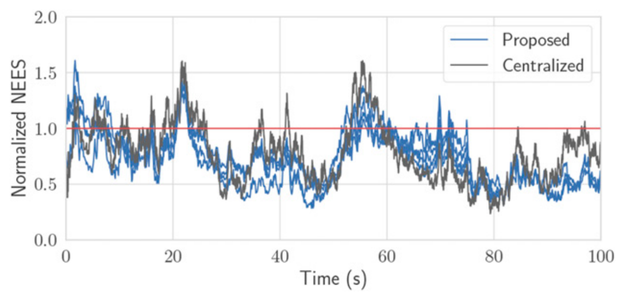

Figure 9. Message size in bytes required to share odometry, as a function of the time period between communications between two robots. The red line "without preintegration" naively transmits all input measurements that occurred within the time period. Preintegration maintains a constant message size while providing identical information.

where $\mathbf{U}_{j_{k-1}}$ has an identical definition as in (20), but computed from Robot j's IMU measurements. When Robot i receives input measurements from its own IMU $\mathbf{u}_k$, it predicts the part of its own state corresponding to its own pose, and additionally performs a partial prediction on the relative poses with

$$\mathcal{X}_{i_k} = \left(\mathbf{G}_{k-1}\mathbf{T}_{wi_{k-1}}\mathbf{U}_{i_{k-1}}, \mathbf{b}_{i_{k-1}}^{[i]}, \mathbf{U}_{i_{k-1}}^{-1}\mathbf{T}_{ij_{k-1}}, \mathbf{b}_{j_{k-1}}^{[i]}, \ldots\right).$$

The terms $\mathfrak{T}_{ij_{k-1}} \triangleq \mathbf{U}_{i_{k-1}}^{-1}\mathbf{T}_{ij_{k-1}}$, which are the partially predicted neighbor poses, are a strange, non-physical intermediate state. Only when the neighbor's RMI $\Delta\mathbf{U}_{j_{pq}}$ is received do the neighbor poses regain meaning with $\mathbf{T}_{ij_q} = \mathfrak{T}_{ij_p}\Delta\mathbf{U}_{j_{pq}}$. However, since biases are also being estimated in this problem, Robot i must first correct the neighbor's raw RMIs $\Delta\mathbf{U}_{j_{pq}}(\mathbf{u}_{j_{p:q-1}}, \mathbf{0})$ using its estimate of the neighbor's IMU bias, as described in Section 6.2. That is,

$$\Delta\mathbf{U}_{j_{pq}} \approx \Delta\mathbf{U}_{j_{pq}}\left(\mathbf{u}_{j_{p:q-1}}, \mathbf{0}\right) \oplus \mathbf{B}_{j_{pq}}\mathbf{b}_{j_q}^{[i]},$$

leading to the full state update given by

$$\mathcal{X}_{i_q} = (\mathbf{T}_{wi_q}, \mathbf{b}_q^{[i]}, \mathfrak{T}_{ij_p}\Delta\mathbf{U}_{j_{pq}}, \mathbf{b}_{j_q}^{[i]} \ldots)$$

Finally, the pseudomeasurements chosen for this problem are

$$\mathbf{c}_{ij}(\mathcal{X}_i, \mathcal{X}_j) = \begin{bmatrix} \mathrm{Log}\left(\mathbf{T}_{wi}\mathbf{T}_{ij}\mathbf{T}_{wj}^{-1}\right) \\ \mathrm{Log}\left(\mathbf{T}_{ij}\mathbf{T}_{ji}\right) \\ \mathbf{b}_i^{[i]} - \mathbf{b}_i^{[j]} \\ \mathbf{b}_j^{[i]} - \mathbf{b}_j^{[j]} \\ \mathrm{Log}\left(\mathbf{T}_{ij}\mathbf{T}_{j\ell}\mathbf{T}_{i\ell}^{-1}\right) \\ \vdots \end{bmatrix}, \ell \in \mathcal{N}_i \cap \mathcal{N}_j.$$

with corresponding covariance $\mathbf{\Psi} = \mathbf{0}$.

8.1. Hardware setup

The hardware setup in these experiments can be seen in Figure 10. Three Uvify IFO-S quadcopters are used that each possess an IMU at 200 Hz, a 1D LIDAR height sensor at 30 Hz, and magnetometers at 30 Hz. Additionally, two ultra-wideband (UWB) transceivers are installed on the quadcopter legs, producing inter-robot distance measurements at 90 Hz for each robot. As shown by Shalaby et al. (2021a), installing multiple UWB tags per robot results in relative position observability. The UWB transceivers are custom-printed modules that use the DW1000 UWB transceiver. The firmware for these modules has been written in C, implementing a double-sided two-way-ranging protocol with details described by Shalaby et al. (2022). Shalaby et al. (2022) also describe the power-based bias calibration and noise characterization procedure used in these experiments. Since all transceivers operate on the same frequency in these experiments, only one can transmit at a time to avoid interference. A decentralized scheduler is therefore implemented that continuously cycles through all transceiver pairs one at a time, obtaining range measurements and potentially transmitting other useful data. In these experiments, the communication graph is complete with all quadcopters capable of communicating with each other. Preintegrated RMIs are shared whenever a UWB measurement occurs, and state sharing occurs at a separate frequency of 10 Hz.

A Vicon motion capture system is used to collect ground truth, from which synthesized absolute position measurements with a standard deviation of 0.3 m are generated for Robots 1 and 2 only. Robot 3 does not receive absolutely position measurements, nor any magnetometer measurements, and therefore has no absolute pose information available without communication with the other two robots. Example trajectories for some of the experimental trials are shown in Figure 11.

8.2. Simulation results

The algorithm is first tested with simulated versions of the described quadcopters, and the estimation results for Robot 3's absolute pose and bias are shown in Figure 12. Although there are many other states associated with the simulation, these states are the most interesting as they are the ones that are unobservable without incorporation of the pseudo-measurements. Figure 12 shows that Robot 3 is capable of estimating its own absolute pose and bias, using information from sensors located on Robots 1 and 2. Furthermore, the errors remain within the 3-sigma confidence bounds, even

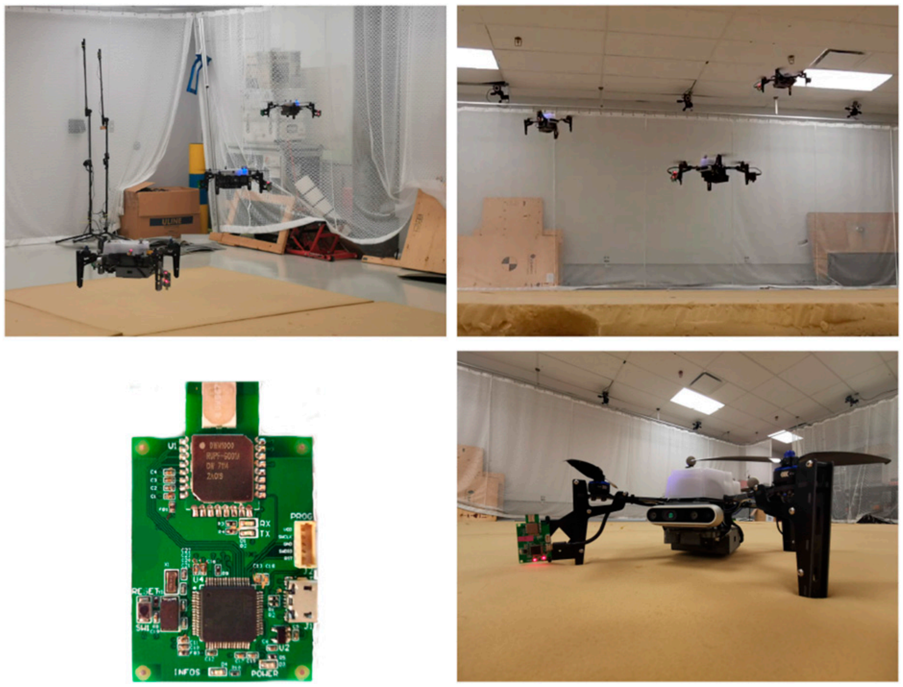

Figure 10. Simulated estimation error of Robot 3's estimate of its own kinematic state and IMU biases. The estimate and corresponding bounds with the proposed algorithm are shown in blue, with the centralized estimate overlayed in dark gray. For attitude, the $x-y-z$ components represent roll-pitch-yaw errors, respectively. Note that Robot 3 does not have position measurements, and therefore cannot observe the states shown in this plot without information sharing. The naive solution has rapidly diverging error and is not plotted.

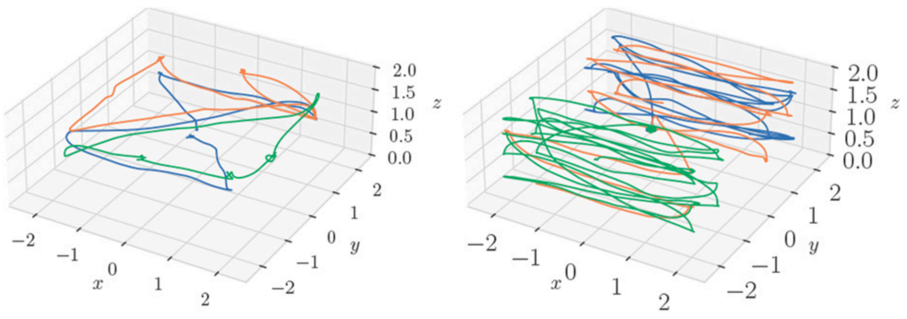

Figure 11. **Top:** Three quadcopters in flight under a motion capture system. **Bottom left:** custom UWB module. **Bottom right:** a close up of the Uvify IFO-S quadcopter, fitted with a UWB module seen on the left leg, as well as on the opposite leg.

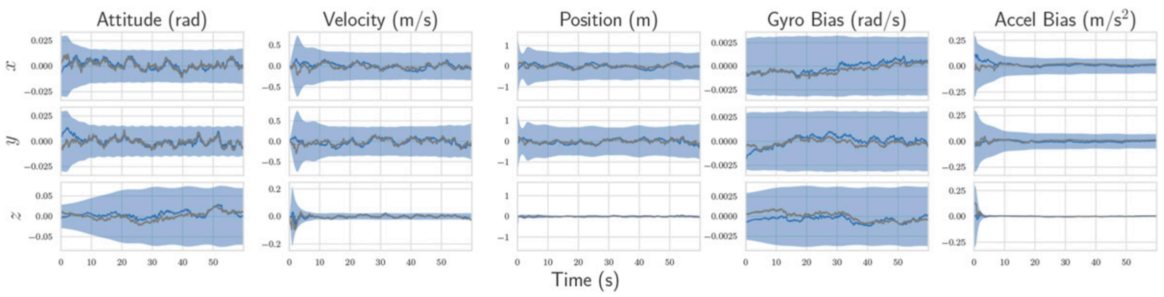

Figure 12. Examples of the various trajectories flown in the experimental trials, where each color represents a different quadcopter.

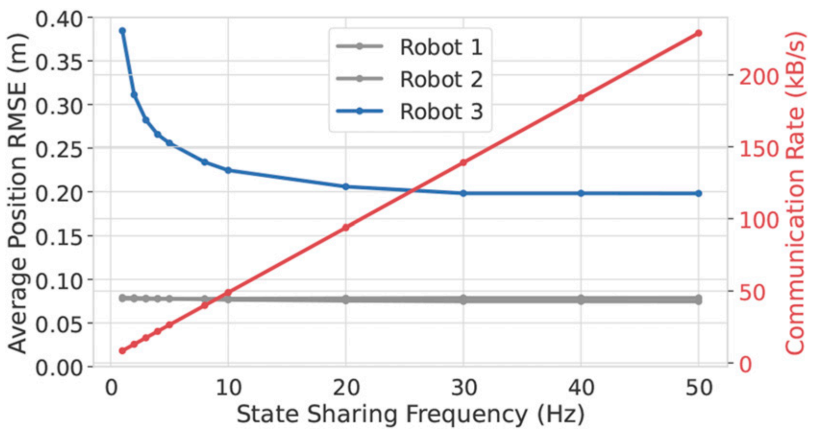

Figure 13. Average self-positioning RMSE with varying communication rate for the simulated version of the quadcopter problem. Robot 3 does not receive position measurements, and hence is reliant on the other robots to have an observable state.

Table 1. Self-Positioning RMSE (m) From Experimental Trials.

	Centralized			Proposed			Error Reduction		
Trial #	Robot 1	Robot 2	Robot 3	Robot 1	Robot 2	Robot 3	Robot 1	Robot 2	Robot 3
1	0.43	0.49	0.55	0.22	0.22	0.61	−48%	−54%	10%
2	0.18	0.26	0.34	0.16	0.18	0.40	−13%	−28%	16%
3	0.17	0.24	0.68	0.16	0.17	0.45	−10%	−28%	−33%
4	0.20	0.25	0.31	0.26	0.28	0.48	32%	−13%	55%

with the first-order RMI bias correction, indicating statistical consistency.

Figure 13 shows the positioning RMSE for varying frequency at which state information between robots is shared. At lower frequencies, Robot 3's estimate has more time to drift between communications, and hence, there is higher error. For this problem, roughly the same estimation performance is achieved for state sharing of 20 Hz and above, with 10 Hz being a compromising value providing a trade-off between accuracy and communication cost.

8.3. Experimental results

Multiple experimental runs are performed on different days, with the absolute positioning results for each robot viewable in Table 1. In some cases, the proposed algorithm even outperforms the centralized solution, which is theoretically optimal. However, the real world contains many unmodelled sources of error, such as frame misalignments, timestamping errors, vibrations, and UWB ranging outliers. These effects may break the assumptions that the optimality of the centralized estimator relies on. Even after tuning covariances to obtain the best centralized performance, it appears that the results benefit from the covariance inflation resulting from CI. For both estimators, the IMU is calibrated to compensate for large biases and scaling factors. The normalized-innovation-squared test (Bar-Shalom et al. (2001)) is also used to reject UWB outliers in both estimators,

A plot of RMSE versus time for Robot 3's absolute states can be seen in Figure 14, which are states that are unobservable from Robot 3's own measurements. Again,

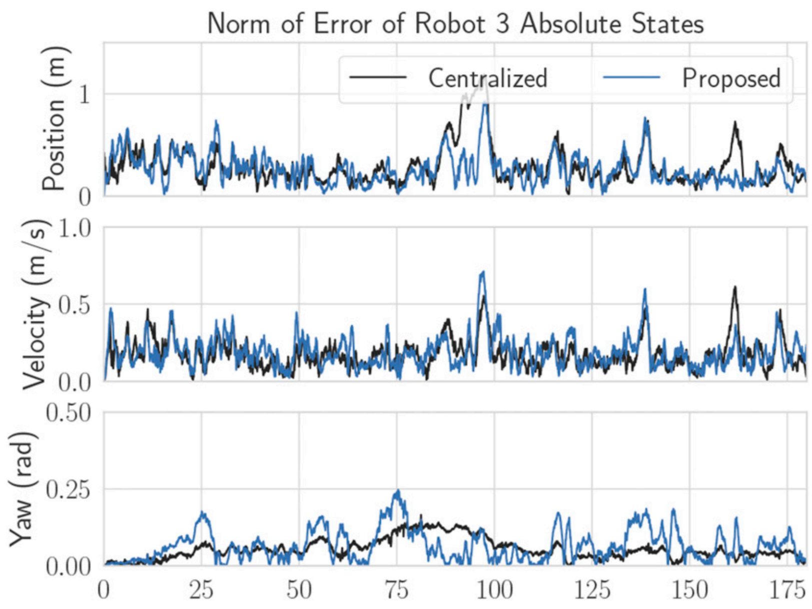

Figure 14. Position, velocity, and yaw RMSE for Robot three from one of the experimental trials. Since Robot 3 has no position measurements, these quantities are unobservable without the fusion of pseudomeasurements.

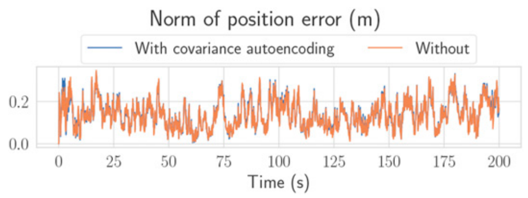

Figure 15. The effect of preintegrated covariance autoencoding, as described in Section 6.3, on the position estimate of Robot 1. The two lines are almost identical, showing that the proposed autoencoder induces minimal error on the estimate. All other states have similar plots.

Figure 14 shows that error magnitudes lie in similar ranges for both the centralized and decentralized estimators. Figure 15 compares two decentralized estimator runs, with one using the mean-assisted autoencoder from Section 6.3. As desired, the lines are identical, and the plot shows that the estimate is unaffected by the autoencoding. This means that the autoencoder is highly effective at compressing the covariance matrix with minimal reconstruction error.

9. Conclusion

This paper presents a general-purpose algorithm for decentralized state estimation in robotics. The algorithm is the result of a new way to formulate the decentralized state estimation problem, specifically with the assistance of pseudomeasurements that allow the definition of arbitrary nonlinear relationships between robot states. For problems involving relative measurements, a communication-efficient approach is proposed for preintegratable process models, as defined by (18), where state-change information is shared in the form of relative motion increments. The algorithm is tested on three different problems, each involving a variety of state definitions, process models, and measurements. In all of the presented problems, robots only need to share their states, RMIs, and corresponding covariances, which ultimately results in average transmission rates per robot of 0.2 kB/s for the toy problem, 4.5 kB/s for the ground robots, and 53.2 kB/s for the quadcopters.

Thanks to covariance intersection, the algorithm is appropriate for arbitrary graphs, and does not require any bookkeeping, growing memory, buffering of measurements, or reprocessing of data. At the same time, the approximation made by covariance intersection makes the proposed method suboptimal, as it is well-known to be overly conservative. Nevertheless, in the specific problems shown in this paper, the results using CI have been satisfactory provided that the fusion frequency is high enough, and the communication graph is not too sparse. It is also worth mentioning that the proposed algorithm still assumes that process model inputs, whether in raw or preintegrated form, have noise that is uncorrelated with the robot states, just like the sensor measurement noise. These assumptions must hold for a consistent estimator. In this paper, it is only correlations between different robots' states that are mitigated by covariance intersection.

One limitation of this proposed approach is that the communication cost grows quadratically with the state size, since the state covariance is also shared. While this is not an issue for small state sizes, such as those representing 3D poses, it could become a problem for states involving multiple time steps or a very large number of robots. Furthermore since this paper allows for variable state definitions between robots, the state definition itself for each robot may need to be communicated, or established a priori.

The initial value of the state is also assumed to be known through an arbitrary initialization procedure.

Future work can consider improving the approximation made by CI and compressing the covariance matrix associated with the state. Also, using the proposed MAP approach with pseudomeasurements, it should be possible to derive decentralized batch and sliding-window estimators, often termed *smoothers*, since these algorithms also originate from the MAP problem.

Acknowledgments

The authors would like to thank Justin Cano and Jérôme Le Ny for their assistance in the development of custom UWB modules.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This work was supported by the NSERC Alliance Grant program, the NSERC Discovery Grant Program, the CFI JELF program, and by the FRQNT.

ORCID iDs

Charles Champagne Cossette https://orcid.org/0000-0002-9380-8361
James Richard Forbes https://orcid.org/0000-0002-1987-9268

References

Allak E, Jung R and Weiss S (2019) Covariance pre-integration for delayed measurements in multi-sensor fusion. In: *International Conference on Intelligent Robots and Systems*. Macau: IEEE, 6642–6649.

Allak E, Barrau A, Jung R, et al. (2022) Centralized-equivalent pairwise estimation with asynchronous communication constraints for two robots. In: *International Conference on Intelligent Robots and Systems*. Kyoto: IEEE, 8544–8551.

Arambel PO, Rago C and Mehra RK (2001) Covariance intersection algorithm for distributed spacecraft state estimation. *Proceedings - American Control Conference* 6: 4398–4403.

Bar-Shalom Y, Li XR and Kirubarajan T (2001) *Estimation with Applications to Tracking and Navigation*. New York: John Wiley & Sons, Inc.

Barfoot TD (2023) *State Estimation for Robotics*. 2nd edition. Cambridge University Press.

Barfoot TD and Furgale PT (2014) Associating uncertainty with three-dimensional poses for use in estimation problems. *IEEE Transactions on Robotics* 30(3): 679–693.

Barrau A and Bonnabel S (2019) Linear observed systems on groups. *Systems and Control Letters* 129: 36–42.

Battistelli G, Chisci L, Mugnai G, et al. (2015) Consensus-based linear and nonlinear filtering. *IEEE Transactions on Automatic Control* 60(5): 1410–1415.

Bourmaud G, Mégret R, Giremus A, et al. (2016) From intrinsic optimization to iterated extended kalman filtering on lie groups. *Journal of Mathematical Imaging and Vision* 55(3): 284–303.

Brossard M, Barrau A, Chauchat P, et al. (2022) Associating uncertainty to extended poses for on lie group IMU preintegration with rotating earth. *IEEE Transactions on Robotics* 38(2): 998–1015.

Carrillo-Arce LC, Nerurkar ED, Gordillo JL, et al. (2013) Decentralized multi-robot cooperative localization using covariance intersection. In: *International Conference on Intelligent Robots and Systems*. Tokyo: IEEE, 1412–1417.

Eckenhoff K, Geneva P and Huang G (2019) Closed-form preintegration methods for graph-based visual–inertial navigation. *The International Journal of Robotics Research* 38(5): 563–586.

Forster C, Carlone L, Dellaert F, et al. (2017) On-manifold preintegration for real-time visual-inertial odometry. *IEEE Transactions on Robotics* 33(1): 1–21.

Grime S and Durrant-Whyte HF (1994) Data fusion in decentralized sensor networks. *Control Engineering Practice* 2(5): 849–863.

Huang GP, Trawny N, Mourikis AI, et al. (2011) Observability-based consistent EKF estimators for multi-robot cooperative localization. *Autonomous Robots* 30(1): 99–122.

Julier S (2001) General decentralized data fusion with covariance intersection (CI). In: *Multisensor Data Fusion*. Norwood: Artech House, 269–294. chapter 12.

Julier SJ and Uhlmann JK (1997) A non-divergent estimation algorithm in the presence of unknown correlations. In: *American Control Conference*. Albuquerque: IEEE, Vol 4, 2369–2373.

Jung R, Brommer C and Weiss S (2020) Decentralized collaborative state estimation for aided inertial navigation. In: *International Conference on Robotics and Automation*. Paris: IEEE, 4673–4679.

Lajoie PY and Beltrame G (2023) Swarm-SLAM: sparse decentralized collaborative simultaneous localization and mapping framework for multi-robot systems. URL. https://arxiv.org/abs/2301.06230

Leung KY, Barfoot TD and Liu HH (2010) Decentralized localization of sparsely-communicating robot networks: a centralized-equivalent approach. *IEEE Transactions on Robotics* 26(1): 62–77.

Leung KY, Barfoot TD and Liu HH (2012) Decentralized cooperative SLAM for sparsely-communicating robot networks: a centralized-equivalent approach. *Journal of Intelligent and Robotic Systems* 66(3): 321–342.

Li H and Nashashibi F (2013) Cooperative multi-vehicle localization using split covariance intersection filter. *IEEE Intelligent Transportation Systems Magazine* 5(2): 33–44.

Li L and Yang M (2021) Joint localization based on split covariance intersection on the lie group. *IEEE Transactions on Robotics* 37(5): 1508–1524.

Luft L, Schubert T, Roumeliotis SI, et al. (2018) Recursive decentralized localization for multi-robot systems with asynchronous pairwise communication. *The International Journal of Robotics Research* 37(10): 1152–1167.

Lupton T and Sukkarieh S (2012) Visual-inertial-aided navigation for high-dynamic motion in built environments without initial conditions. *IEEE Transactions on Robotics* 28(1): 61–76.

Olfati-Saber R (2005) Distributed Kalman filter with embedded consensus filters. In: *IEEE Conference on Decision and Control and European Control Conference*. Orlando, FL, USA: IEEE, 8179–8184.

Olfati-Saber R and Murray RM (2004) Consensus problems in networks of agents with switching topology and time-delays. *IEEE Transactions on Automatic Control* 49(9): 1520–1533.

Pilloni A, Pisano A, Usai E, et al. (2013) Decentralized state estimation in connected systems. *IFAC Proceedings Volumes* 46(2): 421–426.

Psiaki ML (2013) The blind tricyclist problem and a comparative study of nonlinear filters: a challenging benchmark for evaluating nonlinear estimation methods. *IEEE Control Systems Magazine* 33(3): 40–54.

Roumeliotis SI and Bekey GA (2002) Distributed multirobot localization. *IEEE Transactions on Robotics and Automation* 18(5): 781–795.

Shalaby M, Cossette CC, Forbes JR, et al. (2021a) Relative position estimation in multi-agent systems using attitude-coupled range measurements. *IEEE Robotics and Automation Letters* 6(3): 4955–4961.

Shalaby M, Cossette CC, Le Ny J, et al. (2021b) Cascaded filtering using the sigma point transformation. *IEEE Robotics and Automation Letters* 6(3): 4758–4765.

Shalaby MA, Cossette CC, Forbes JR, et al. (2022) Calibration and uncertainty characterization for ultra-wideband two-way-ranging measurements (preprint). URL https://arxiv.org/abs/2210.05888v2.

Shalaby M, Cossette CC, Forbes JR, et al. (2023) Multi-robot relative pose estimation and IMU preintegration using passive UWB transceivers. (preprint) URL https://arxiv.org/abs/2203.11004.

Solà J, Deray J and Atchuthan D (2018) A micro lie theory for state estimation in robotics. URL https://arxiv.org/abs/1812.01537

Tian Y, Chang Y, Herrera Arias F, et al. (2022) Kimera-multi: robust, distributed, dense metric-semantic SLAM for multi-robot systems. *IEEE Transactions on Robotics* 38(4): 2022–2038.

Xu H, Wang L, Zhang Y, et al. (2020) Decentralized visual-inertial-UWB fusion for relative state estimation of aerial Swarm. In: *IEEE International Conference on Robotics and Automation*. Paris: IEEE, 8776–8782.

Zhu J and Kia SS (2019) Cooperative localization under limited connectivity. *IEEE Transactions on Robotics* 35(6): 1523–1530.