Sharing visual-inertial data for collaborative decentralized simultaneous localization and mapping

Rodolphe Dubois, Alexandre Eudes, Vincent Frémont

Sharing Visual-Inertial Data for Collaborative Decentralized Simultaneous Localization And Mapping

Rodolphe Dubois[a,b], Alexandre Eudes[a], Vincent Frémont[b]

[a] *DTIS, ONERA, F-91123 Palaiseau, France*
[b] *Centrale Nantes, LS2N, UMR 6004, Nantes, France*

Abstract

Building on the maturity of single-robot SLAM algorithms, collaborative SLAM has brought significant gains in terms of efficiency and robustness, but has also raised new challenges to cope with like informational, network and resource constraints. Several multi-robot frameworks have been coined for visual SLAM, ranging from highly-integrated and fully-centralized architectures to fully distributed and decentralized methods. However, many proposed architectures compromise the autonomy of the robots in fusing the data processed by the other agents to enhance their own estimation accuracy. In this paper, we propose three methods to share visual-inertial information, based on rigid, condensed and pruned visual-inertial packets. We also propose a common collaborative SLAM architecture to organize the computation, exchange and integration of such packets. We evaluated those methods on the EuRoC [1] dataset and on our custom dataset AirMuseum [2]. Experiments showed that the proposed methods allow the agents to build, exchange and integrate consistent visual-inertial packets, and improve their trajectory estimation accuracy up to several centimeters.

Keywords: Robotics, Visual-Inertial Collaborative SLAM

1. Introduction

Collaborative robotics has gained much attention over the past years as it promises to make the traditional applications of mobile robotics more efficient and robust. Indeed, missions such as exploration, inspection and Search-And-Rescue (SAR) [3] operations may benefit from the parallelization and distribution of tasks, the inter-robot mutual information derived from the fusing of their complementary observations, and the resiliency of the fleet in facing the loss of one of its agents. Such applications depend on the ability of the robots to navigate through an environment without prior knowledge on it, as aimed by Simultaneous Localization And Mapping (SLAM).

While single-robot SLAM has been extensively studied [4], a current challenge is to extend it to collaborative frameworks. Collaborative SLAM (CSLAM) does not boil down to a mere extension of single-robot SLAM as it raises specific challenges to cope with. The first challenge is informational. The exchanged data should allow to spot inter-robot correspondences and consistently re-estimate the trajectories while faithfully reflecting the knowledge of the agents. Secondly, processing inter-robot interactions should not cut into the computational and memory resources required by the other processes running in parallel to the SLAM algorithm. Finally, robots should address communication issues stemming from the limited bandwidth and communication range. Despite those difficulties, CSLAM demonstrates an enhanced potential for various domestic, industrial and military applications. However, by relying on task integration in centralized architectures or task distribution in decentralized architectures, most proposed solutions undermine the data processing autonomy of the robots when it comes to fuse information from the other robots by exclusively relying on their own resources and data.

The goal of this paper is to design data sharing methods for collaborative decentralized visual-inertial SLAM, which enforce the agents' autonomy while coping with informational, communication and resource constraints. They aim at allowing each agent to process the information provided by its own sensors as well as the information received from the other agents independently, without the need for cloud computing resources. As contributions, we propose three ways of locally summarizing the visual-inertial information: condensed packets based on the computation of consistent virtual factors, pruned packets which select a subset of raw visual-inertial factors in a less conservative way (both packets being completed with raw 2D visual information), and rigid packets which convey more exhaustive visual-structural information. We also build a full collaborative SLAM architecture to organize their computation, exchange and integration. Dedicated to multi-robot navigation, those methods aim at enhancing the estimation accuracy of each robot. They

Email addresses: rodolphe.dubois@protonmail.com (Rodolphe Dubois), alexandre.eudes@onera.fr (Alexandre Eudes), vincent.fremont@ls2n.fr (Vincent Frémont)

were designed to be agnostic to the SLAM Front-End as long as it outputs a visual-inertial factor graph. We evaluate them on multi-robot scenarios built from the EuRoC dataset [1] and our custom AirMuseum dataset [2].

This paper extends our previous work [5] in three ways. First, it proposes multiple changes into the building of the communicated packets and their integration into the recipient robots' maps. Secondly, it brings a deeper focus into the underlying system architecture regarding the chosen communication policy and task allocation scheme. Finally, it provides a more exhaustive performance evaluation of the proposed methods in coping with the embeddability, communication and information constraints which are classically faced by multi-robot SLAM.

The paper is organized as follows. Section 2 first tackles the structure of multi-robot SLAM algorithms and reviews the main contributions which have been published in that field. Section 3 introduces the notations and mathematical notions which we will use throughout the paper. Sections 4, 5 and 6 detail the proposed common collaborative SLAM architecture regarding the retained task and data allocation scheme, the communication policy and the data association and fusion strategy. The next three sections 7, 8 and 9 propose three ways of building visual-inertial packets to be exchanged between robots, each one being introduced by a specific related work subsection. Finally, experiments are carried out and discussed in section 10.

2. Related works

2.1. Simultaneous Localization And Mapping

Simultaneous Localization And Mapping methods concurrently estimate a map of the observed environment and the trajectory the robot has run within it, and continuously refines them through local and global relocalization against the reconstructed map. Figure 1 depicts the general architecture of a SLAM algorithm. It first requires at least one exteroceptive sensor like a LiDAR [6] or a camera bench [7] for visual SLAM (VSLAM), those latter being cheaper, lower power consuming and easier to embed on drones. A local odometry and mapping module interprets, abstracts and fuses such measurements to initialize locally consistent map and trajectory estimates. However, it inevitably drifts, what compromises the global consistency of the map. Hence, a correspondence detection module builds on place recognition algorithms [8] to spot global loop closures when the robot observes previously mapped areas. Finally, a global inference module may globally and smoothly refine the estimates based on the spotted correspondences through Maximum A Posteriori (MAP) estimation. All those modules are flagged either as Front-End if they process incoming data at their reception rate, or Back-End processes otherwise.

The contributions of this article focus on Visual-Inertial (VI) SLAM, whose sensor bench enhances a monocular camera with an Inertial Measurement Unit (IMU) [9, 10]. Such coupling of low-cost sensors brings multiple advantages: not only does the IMU allow to scale the 3D model reconstructed from monocular images and align it with gravity vector, but it also improves the visual tracking, especially in the case of tight-fusion VI Odometry (VIO) [11]. However, it requires a steady time-synchronization between the IMU and the camera, and enlarges the estimation problem with velocity and inertial biases states.

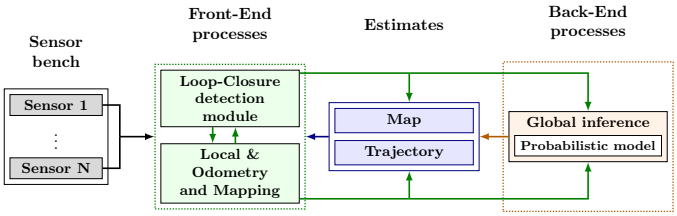


Figure 1: Common architecture of a SLAM algorithm

2.2. Collaborative SLAM

As emphasized in [4], single-robot SLAM has achieved high maturity and widened new research perspectives, one of which being CSLAM [12]. A CSLAM framework extends single-robot SLAM architectures along three axis: i) a task and data allocation scheme; ii) an adapted communication policy and iii) a matching and merging (or data association and fusion) strategy. The allocation scheme governs how each task and data is either centralized, decentralized or distributed. The communication policy supervises the topology, planning and content of inter-robot exchanges. Finally, the matching and merging strategy deals with the spotting of inter-robot correspondences and multi-robot inference.

2.3. Task and Data allocation scheme

A task (resp. piece of data) is centralized when a single agent is responsible for its completion (resp. storage). The latter aggregates and processes data collected by the others and eventually reports results and updates to them. On the contrary, the task is decentralized when it is performed by multiple agents. It is distributed if its computations are shared among several agents.

2.3.1. Centralized allocation schemes

Centralized allocation schemes rely on a central server which exhibits increased computational power and storage capacity. Though relieving the agents from complex Back-End tasks, such architectures might nonetheless undermine their data processing autonomy. Existing methods can thus be classified according to what extents the agents are integrated to the server. A first approach, coined by *CoSLAM* [13], reduces agents to mobile sensors and concentrates all mapping and localization tasks on the server. On the contrary, a second strategy grants more autonomy to the agents by providing them a full SLAM pipeline, while it delegates the detection of global inter-robot correspondences and global inference operations to

the server. Such approach has been adopted by *C^2TAM* [14], *MOARSLAM* [15] and *CORB-SLAM* [16], which all build on client-server architectures. Finally, a third and intermediary strategy splits individual SLAM pipelines between the agents and the server but leaves critical VIO modules to agents. For instance, *CSfM* [17] spawns one thread handler per agent on the server which builds it associated map while a common place recognition module spots intra and inter-robot correspondences. However, *CSfM* does not provide any feedback to the agents, contrary to *CCM-SLAM* [18] and *CVI-SLAM* [19] for monocular and visual-inertial CSLAM. In the same way, Qin et al. proposed *CoVins* [20] which extends *Vins-Mono* [10] into CSLAM, and Deustch et al. introduced the generic *TeamSLAM* [21] framework.

2.3.2. Decentralized allocation schemes

Decentralized allocation schemes seek increased flexibility for the fleet and enhanced autonomy for its agents. Each agent thus maintains its own map and is responsible for its full SLAM pipeline and the integration of the data received from the other agents (e.g. spotting of inter-robot correspondence and running global inference). Such approach has been implemented by several methods such as *DDF-SAM* [22], *AUV-CSLAM* [23] for underwater acoustic SLAM, Schuster et al. [24] for dense stereo-visual collaborative SLAM, Dubois et al. [5] and Sartipi et al. [25] for visual-inertial CSLAM. Decentralized allocation schemes are more conducive to the distribution of tasks and data which compensates for the absence of a central server. Such pooling of computational resources may target specific modules such as correspondence detection [26] and global correspondences consistency checks [27], distributed inference [27, 28] and map storage [29, 30].

2.4. Communication policies

2.4.1. Communication topology

The communication topology defines the set of robots each agent can communicate with. Though the adopted topology should match to the chosen allocation scheme, it still retains some degrees of freedom. In centralized topologies, some frameworks [13, 17] restrict to unidirectional exchanges from the agents to the server while a second family [18, 19] make the server provide feedback to the agents through bidirectional exchanges. On the opposite, decentralized topologies allow peer-to-peer exchanges. They certainly make multi-robot interactions more flexible, but also denser and unstructured. Special attention should be paid to handle communication losses and recoveries, and to prevent double-counting issues. Double-counting arises when a robot fuses received data while being unaware of its eventual correlations with its own estimates; this is likely to stem from cyclic data exchanges and result into inconsistent, overconfident and biased estimates.

2.4.2. Communicated packets

SLAM algorithms process three kinds of data: i) raw or pre-processed sensor outputs (like image keypoints and descriptors); ii) probabilistic models and iii) map and trajectory estimates. The communication policy dictates which of them are shared among the agents. Communication strategies can be classified according to the degree of processing involved in the exchanged data, and whether such data is communicated exhaustively or in a summarized way. In most centralized methods such as *CSfM* [17] and *CVI-SLAM* [19], agents communicate extensive information on each keyframe. It may include up to keypoints, descriptors, landmarks, inertial measurements, etc. Van Opdenbosch et al. [31] proposed to communicate a selection of compressed visual binary features to share visual information. Schuster et al. [24] make robots exchange submaps augmented with rigid stereo-triangulated dense point clouds, pose estimates and a factor graph. A second strategy relies on marginalization and sparsification techniques, as in *DDF-SAM* [22], Lazaro et al. [32] using condensed measurements and *AUV-CSLAM* [23].

2.4.3. Communication planning

The communication policy schedules the sending of the packets. Two strategies have been coined in the literature. The first one is robot-centric as packets are broadcast based on intrinsic criteria such as the run distance. It ensures regular data exchanges which may occur keyframe-per-keyframe as in *CSfM* [17] and *DOOR-SLAM* [27], or submap-per-submap as done by *AUV-CSLAM* [23]. The second strategy is *altruistic* as exchanged packets are generated based on the other agents' estimated knowledge. Such approach was notably investigated by Tian et al. [33]: preliminary exchanges of light metadata allow to identify potential inter-robot correspondences and subsequently compute the optimal exchange policy to minimize the communication cost and the induced labor division.

2.5. Matching & Merging Strategies

2.5.1. Inter-robot correspondences and loop closures

When a robot receives data from another robot, its first task is to spot eventual correspondences with its own observations. This step is critical for the mutual registration of the reference frames attached to the robots, and for the multi-robot inference as it enforces inter-robot mutual information. Correspondences may first stem from direct inter-robot observations, through marker detection [34] or appearance tracking [35]. Additionally, indirect correspondences are spotted through visual-structural correspondences. Instantaneous indirect correspondences, which only compares the robots' current observations, are mostly used for collaborative localization [25, 36]. Retrospective indirect correspondences may be spotted between any inter-robot pair of keyframes upon the reception of data from other robots. Centralized architectures take advantage of the central server to spot such correspondences, while decentralized ones may distribute the task [26], make each

robot build one visual database per other robot [27], or leverage preliminary metadata exchanges to circumscribe the search [37].

2.5.2. Multi-robot inference

Though filtering-based multi-robot inference has first been proposed [38], most CSLAM algorithms have taken advantage of the flexibility of factor graphs. They naturally extend to multi-robot frameworks, for which ones Kim et al. [39] proposed to use *anchor nodes* to handle multiple trajectories. Furthermore, factor graphs are prone to support distributed optimizations as proposed by Choudhary et al. [28] and *DOOR-SLAM* [27]. Decentralized methods should prevent double-counting which may compromise the consistency of the estimates. For such purpose, Cunningham et al. [22] coined *anti-factors* to subtract information while Paull et al. [23] send consecutive independent local packets. Multi-robot inference is also responsible for estimating the transformations between the reference frames attached to the robots. To that aim, Indelman et al. [40] use Expectancy-Maximization (EM) optimization where latent variables are used to deactivate inconsistent inter-robot correspondences.

2.6. Autonomy in multi-robot frameworks

This literature review shows that the current trend for CSLAM to enhance efficiency and robustness is to develop either highly-integrated centralized architectures or highly-distributed decentralized frameworks. However, both approaches prevent the agents from independently fusing the information received from the other agents with the information provided by their own sensors by extensively resorting to cloud computing resources, and thus mitigate their autonomy. Indeed, they either depend on a central server whose failure may neutralize the whole fleet, or mutually depend on each other. To tackle this issue, we propose three data sharing methods for VI CSLAM which enforce the autonomy of the agents while addressing the information, network and resource constraints.

3. Notations

In the next sections, ${}^{\mathrm{A}}\boldsymbol{t}_{\mathrm{BC}}$ denotes a physical quantity $\boldsymbol{t}$ attached to C w.r.t. B expressed in A, where A, B and C are reference frames. $\mathbb{SE}_n$ and $\mathbb{SO}_n$ respectively denote the *Special Euclidean Group* and the *Special Orthogonal Group* of dimension n. We parameterize $\mathbb{SE}_3$ using the composite Lie group $\mathbb{SO}_3 \times \mathbb{R}^3$. The operator $\oplus$ denotes the $\mathbb{SE}_3$ pose product. We define the $\boxplus$ and $\boxminus$ operators:

$$\boldsymbol{\delta\tau} \triangleq T_2 \boxminus T_1 = \begin{bmatrix} \log_{\mathbb{SO}_3}(\boldsymbol{R}_2 \cdot \boldsymbol{R}_1^\top)^\vee \\ \boldsymbol{t}_2 - \boldsymbol{t}_1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{\delta\theta} \\ \boldsymbol{\delta p} \end{bmatrix} \quad (1)$$

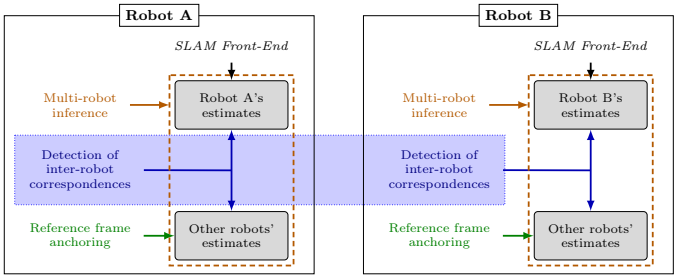
$$T_2 = \boldsymbol{\delta\tau} \boxplus T_1 = \begin{bmatrix} \exp_{\mathbb{SO}_3}(\boldsymbol{\delta\theta}^\wedge) \cdot \boldsymbol{R}_1 & \boldsymbol{\delta p} + \boldsymbol{t}_1 \\ \boldsymbol{0} & 1 \end{bmatrix} \quad (2)$$

where $\boldsymbol{R}_{(\cdot)} \in \mathbb{SO}_3$ and $\boldsymbol{t}_{(\cdot)} \in \mathbb{R}^3$ parameterize $T_{(\cdot)} \in \mathbb{SE}_3$, $\log_{\mathbb{SO}_3}$ maps from $\mathbb{SO}_3$ to its Lie algebra $\mathfrak{so}_3$, the *vee* operator maps from $\mathfrak{so}_3$ to $\mathbb{R}^3$, and the $\exp_{\mathbb{SO}_3}$ and *hat* operators respectively map reversely. Interested readers may refer to [41] for more details on operations on Lie groups. $\boldsymbol{J}_{\boldsymbol{\theta}}^{f(\boldsymbol{\theta})}(\hat{\boldsymbol{\theta}})$ denotes the Jacobian matrix of $f(\boldsymbol{\theta})$ w.r.t. $\boldsymbol{\theta}$ evaluated at $\hat{\boldsymbol{\theta}}$. Finally, the Mahalanobis norm of $\boldsymbol{x} \in \mathbb{R}^n$, weighted by a symmetric positive-definite matrix $\boldsymbol{A}$, is denoted $\|\boldsymbol{x}\|_{\boldsymbol{A}}$.

4. Overview of the proposed architecture

In this paper, we propose three decentralized collaborative visual-inertial SLAM algorithms dedicated to multi-robot navigation, respectively based on the exchange of condensed, pruned and rigid visual-inertial packets. We first assume that each robot estimates its own map and trajectory using a given VIO algorithm which outputs a visual-inertial factor graph. We make robots regularly exchange data packets which summarize successive, uncorrelated and local portions of their map, and append them with a selection of visual information. Such packets are used by the receiver to spot inter-robot correspondences, and estimate the trajectories with enhanced accuracy. A client-server architecture is also added: it allows the robots to optionally request additional visual information upon the spotting inter-robot correspondences in order to dynamically refine them.

Task and data allocation scheme. As illustrated in Figure 2, we use a fully decentralized and weakly distributed task and data allocation scheme: each robot is responsible for running its own SLAM algorithm and integrating the received packets into its own estimated map. Nonetheless, it yields an implicit labour division since each robot restricts to spotting correspondences against its own map.



Each robot handles its own Front-End modules to build its map and spot loop-closures within its own trajectory. It also handles multi-robot inference and reference frame anchoring. However, it only detects inter-robot correspondences against its own observations.

Figure 2: Task and Data allocation scheme

Communication policy. The proposed decentralized communication policy, described in section 5, splits into two communication regimes. First, a regular communication policy makes robots regularly build and exchange data packets from successive uncorrelated local submaps to avoid double-counting issues. Each packet decouples the data for place recognition and trajectory estimation.

It also handles the exchange of spotted correspondences and direct inter-robot observations. A client-server architecture allows each robot to request additional visual information after spotting inter-robot correspondences. The communication policy is completed with a regularization policy which operates when two robots recover contact.

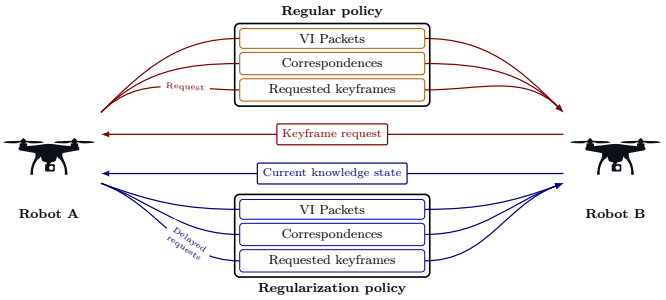


Figure 3: Communication policies

Matching and Merging strategy. Received packets are integrated independently from each other and connected to the previously received ones in a dedicated reference frame. Those latest get mutually anchored as soon as enough inter-robot correspondences between them are available. Upon reception, each new packet is used to spot inter-robot correspondences with the receiver's past keyframes, and regularly matched against the receiver's new keyframes. Newly spotted correspondences trigger global multi-robot inference which only uses the factors attached to sent and received packets (see section 6).

5. Common communication policy

5.1. Regular communication policy

The regular communication policy supervises the exchange of three kinds of data: i) packets; ii) spotted correspondences and iii) requested visual frame information, as described in Figure 3. The computation and broadcasting of a new packet is triggered based on kinematic criteria, as soon as the robot estimates it has traveled a distance $d \geq d_{\max}$ since it computed its last packet, where $d_{\max}$ can be set to several meters. The computed packet is then broadcast to all reachable neighbors. Each spotted correspondence is broadcast the same way. A correspondence message holds the IDs of the robots, the timestamp of their matched keyframes and the measurement itself. Sent packets and correspondences are associated with sending indexes, which are incremented for each newly broadcast packet or correspondence. Robots maintain and update a table which maps such indexes either to the correspondence's ID or a structure describing the spatial extend of the packet. Finally, when a robot spots a inter-robot correspondence, it can request additional visual information about the k neighboring frames to the matched one to spot more correspondences. As a response, the queried robot sends a selection of the keypoints and descriptors of the neighboring frames (see §5.3).

5.2. Regularization communication policy

In a decentralized framework with limited communication range, robots may occasionally lose contact between each other when covering large areas, and thus miss some valuable information broadcast in the meanwhile. When they recover contact, they should regularize their knowledge. For that purpose, we make each robot hold a reception bookkeeping. Robot i stores the index of the most recently received information from each other robot $j \neq i$. When two robots recover contact, they first exchange their current reception bookkeeping. They can then regularize their knowledge by sending to the other robot all the packets and all the correspondences with posterior indexes. For each robot, data must be sent in chronological order not to break the consistency of the reception history if the communication was to be lost again during the process.

5.3. Visual information selection

All presented methods resort to a keyframe and keypoint selection mechanisms. Given a sub-trajectory, the objective is to select a subset of the keyframes which covers the whole environment observed from that sub-trajectory, as described by Algorithm 1. As a second step, we select a subset of the keypoints on the selected keyframes. We use a simple heuristic which involves to retain the n_{kp} keypoints associated with the most observed landmarks, as they are likely to be associated to the most stable and informative descriptors. The track length is a good heuristic indicator of the salience of a keypoint, which depends on its distinctiveness, repeatability and detectability.

Algorithm 1– Keyframe selection on a sub-trajectory

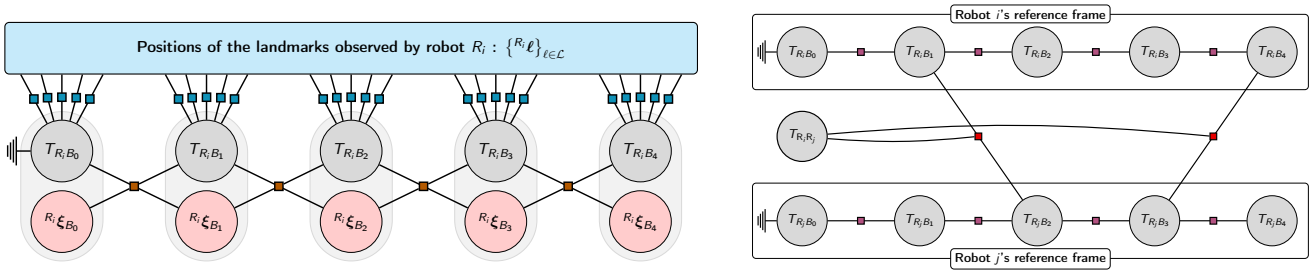
Parameters

$n_{\min}$: threshold on commonly observed landmarks;
$q_{\min}$: threshold quotient of commonly observed landmarks;
α : coefficient between 0 and 1;
$\mathcal{L}_i$: set of observed landmarks from keyframe with index i;

Input: $I_{1:N} = \{I_0, ..., I_N\}$ the set of successive keyframes within the given sub-trajectory.

Output: $\mathcal{S}$ the set of selected keyframes.

$\mathcal{S} \leftarrow \{I_0, I_N\}$;
$r \leftarrow 0$;
for $i \in \{1, ..., N-1\}$ **do**
  $n_{ri} \leftarrow |\mathcal{L}_r \cap \mathcal{L}_i|$;
  $q_{ri} \leftarrow \dfrac{|\mathcal{L}_r \cap \mathcal{L}_i|}{|\mathcal{L}_r|}$;
  if $n_{iN} \leq n_{\min}$ ***ou*** $q_{iN} \leq q_{\min}$ **then**
    $n_{iN} \leftarrow |\mathcal{L}_i \cap \mathcal{L}_N|$;
    $q_{iN} \leftarrow \dfrac{|\mathcal{L}_i \cap \mathcal{L}_N|}{|\mathcal{L}_N|}$;
    if $n_{ri} \leq \alpha \cdot n_{\min}$ ***ou*** $q_{ri} \leq \alpha \cdot q_{\min}$ **then**
      $\mathcal{S} \leftarrow \mathcal{S} \cup \{I_i\}$;
      $r \leftarrow i$;



Inertial ■ and visual ■ factors encode stochastic constraints between pose ○, inertial ○ and landmark ○ states.

(a) Structure of a visual-inertial factor graph

The trajectories of robots i and j are estimated in their own reference frame. The pose offset between them is modeled as an anchor node [39] which is involved in inter-robot factors.

(b) Multi-robot factor graph

Figure 4: Single and multi-robot factor graphs

6. Common Matching & Merging strategy

6.1. Correspondence detection module

To spot indirect correspondences, we use the pipeline developed by Lynen et al. [42] and implemented in the *Maplab* framework [43]. Each spotted correspondence is broadcast to all reachable robots. When a correspondence is received from another robot, it is added to the map if both corresponding keyframes are known. Otherwise, it is buffered until missing keyframes from upcoming packets are received. When a new packet is received, the robot looks for correspondences between the received keyframes and its own keyframes. It also checks whether some buffered correspondences may be added to the map. Finally, when it adds a new keyframe to its own trajectory, it looks for loop closures with its own keyframes, as well as inter-robot correspondences with the keyframes received from the other robots. All spotted correspondences are broadcast to the other robots.

6.2. Global inference module

The trajectory is modeled as a discrete set of consecutive keyframes, each one being described by its pose $T_{\mathrm{WB}_k} \in \mathbb{SE}_3$, its velocity ${}^{\mathrm{W}}\boldsymbol{v}_{\mathrm{WB}_k}$ and its inertial biases ${}^{\mathrm{B}_k}\boldsymbol{b}_k$, where W and B respectively denote the global and the body inertial frames. The environment is sparsely represented as a set of 3D landmarks ℓ_i whose positions ${}^{\mathrm{B}_i}\boldsymbol{l}_i$ are estimated w.r.t. their first observer keyframe B_i. We encompass all those variables in the set Θ and denote their associated space $\mathcal{H}_\Theta$. The knowledge carried by a set of measurements $\mathcal{Z}$ onto a set of variables Θ is modeled as a graph of factors [44]. This is a bipartite graphical probabilistic model which encodes the likelihood $p(\mathcal{Z}|\Theta)$ of the observations w.r.t. the estimated variables. Under the hypothesis of identically and independently distributed observations, it factors as:

$$p(\mathcal{Z}|\Theta) = \prod_{z\in\mathcal{Z}} p(z|\Theta) \tag{3}$$

Each observation $z \in \mathcal{Z}$ yields a factor $p(z|\Theta)$ which appears in such decomposition. In the Gaussian case:

$$p(z|\Theta) = \mathcal{N}(\boldsymbol{\xi}_z(\Theta); \mathbf{0}, \boldsymbol{\Sigma}_{\boldsymbol{\xi}_z}) \propto \exp\left(-\frac{1}{2}\|\boldsymbol{\xi}_z(\Theta)\|^2_{\boldsymbol{\Sigma}_{\boldsymbol{\xi}_z}}\right) \tag{4}$$

where $\mathcal{N}$ is the normal distribution, $\mathbf{0}$ is the null vector, $\boldsymbol{\xi}_z(\Theta)$ is the associated residual and $\boldsymbol{\Sigma}_{\boldsymbol{\xi}_z}$ its covariance matrix. Inference is performed by minimizing the resulting negative log-likelihood of the measurements:

$$\hat{\Theta} = \underset{\Theta\in\mathcal{H}_\Theta}{\arg\max}\, p(\mathcal{Z}|\Theta) = \underset{\Theta\in\mathcal{H}_\Theta}{\arg\min} \sum_{z\in\mathcal{Z}} \|\boldsymbol{\xi}_z(\Theta)\|^2_{\boldsymbol{\Sigma}_{\boldsymbol{\xi}_z}} \tag{5}$$

which yields a nonlinear least-square optimization problem over the residuals in the Gaussian case, classically solved using the Levenberg-Marquardt algorithm. In the visual-inertial case, the associated factor graph has the form depicted by Figure 4a with inertial and visual factors which are defined below. In our multi-robot framework, we use the architecture advocated by Kim et al. [39], where each robot's baseframe is associated to a dedicated *anchor* node as illustrated by Figure 4b. Multi-robot inferences are triggered as soon as a minimum number of new correspondences have been spotted or if a minimum time duration has elapsed since the spotting of the latest new and unprocessed correspondence. In the next paragraphs, we detail the residuals for inertial, visual and relative pose factors.

6.2.1. Inertial factors

Each inertial measurements u_k connecting keyframes $\mathcal{K}_\mathrm{k}$ and $\mathcal{K}_{\mathrm{k}+1}$ yields an inertial residual:

$$\boldsymbol{\xi}_{u_k}(\Theta) = \begin{bmatrix} T_{\mathrm{WB}_{\mathrm{k}+1}} \boxminus \hat{T}_{\mathrm{WB}_{\mathrm{k}+1|\mathrm{k}}} \\ {}^{\mathrm{W}}\boldsymbol{v}_{\mathrm{WB}_{\mathrm{k}+1}} - {}^{\mathrm{W}}\hat{\boldsymbol{v}}_{\mathrm{WB}_{\mathrm{k}+1|\mathrm{k}}} \\ {}^{\mathrm{B}_{\mathrm{k}+1}}\boldsymbol{b}_{\mathrm{k}+1} - {}^{\mathrm{B}_\mathrm{k}}\boldsymbol{b}_\mathrm{k} \end{bmatrix} \tag{6}$$

where $\hat{T}_{\mathrm{WB}_{\mathrm{k}+1|\mathrm{k}}}$, ${}^{\mathrm{W}}\hat{\boldsymbol{v}}_{\mathrm{WB}_{\mathrm{k}+1|\mathrm{k}}}$ are predicted using a RK4 integrated discrete-time dynamics of the robot from the pose and inertial states at time k. Inertial factors may also be derived from pre-integrated IMU factors [45].

6.2.2. Visual factors

The observation $\boldsymbol{z}_{\mathrm{ij}}$ of landmark l_j from keyframe $\mathcal{K}_i$ yields a visual residual:

$$\boldsymbol{\xi}_{z_{\mathrm{ij}}}(\Theta) = \boldsymbol{z}_{\mathrm{ij}} - \pi_{B_i}\left(T_{\mathrm{WB}_i}^{-1} \oplus T_{\mathrm{WB}_j} \cdot {}^{\mathrm{B}_j}\boldsymbol{l}_j\right) \tag{7}$$

where B_j denotes the first observer keyframe of landmark l_j. The function π_{B_i} is the pinhole camera projection which maps a 3D point expressed in B_i onto the camera image plane, and which depends on distortion, intrinsic and camera-IMU extrinsic parameters.

6.2.3. Relative pose factors

A relative pose factor z_{ij} between keyframes $\mathcal{K}_i$ and $\mathcal{K}_j$ of robots R_i and R_j yields the residual:

$$\boldsymbol{\xi}_{z_{\mathrm{ij}}}(\Theta) = \left(T_{\mathrm{R}_i\mathrm{B}_i}^{-1} \oplus T_{\mathrm{R}_i\mathrm{R}_j} \oplus T_{\mathrm{R}_j\mathrm{B}_j}\right) \boxminus \hat{T}_{\mathrm{B}_i\mathrm{B}_j} \tag{8}$$

where $\hat{T}_{\mathrm{B}_i\mathrm{B}_j}$ is the estimated relative pose, and $T_{\mathrm{R}_i\mathrm{R}_j}$ is the offset pose between the reference frames attached to robots R_i and R_j, associated to an *anchor* node [39].

7. Condensed Visual-Inertial Packets (CVIP)

7.1. Related works

A first strategy to communicate visual-inertial information is to summarize it into condensed packets. Marginalization techniques allow to bound the ever-increasing complexity of the SLAM inference problem by pruning variables without losing information. However, it yields a dense distribution as it induces cross-correlations between all the remaining variables. Sparsification is the process of approximating this dense distribution by a sparse one with less partial pairwise correlations yielded by a set of *virtual* factors and which result into the same MLE estimate. In the Gaussian case, it involves to compute the mean and covariance of the virtual factors of the new topology.

Ideally, the sparsification process should minimize the loss of information which is commonly quantified using the relative entropy i.e. Kullabck-Leibler divergence (KLD) w.r.t. the dense distribution. This first requires a judicious choice for the enforced topology. Invertible topologies such as spanning trees (e.g. Chow-Liu trees [46]) allow closed-form sparsification as used for *Generic Linear Constraints* [47] and *Nonlinear Factor Recovery* [48]. However, richer topologies may provide better approximations at the cost of requiring iterative minimization with the constraint of enforcing a positive-definite information matrix. For such case, Interior Point [49], quasi-Newton [48] and block-coordinate descent [50] methods can be used.

The sparsified distribution should ideally be consistent with the original one, meaning that it should not add artifactual information by artificially reducing the uncertainty on any subset of variables. In the Gaussian case, this yields a Loewner ordering constraint over the dense and sparsified covariance matrices, respectively denoted $\boldsymbol{\Sigma}_d$ and $\boldsymbol{\Sigma}_s$: $\boldsymbol{\Sigma}_d \leq \boldsymbol{\Sigma}_s$. Such constraint can be enforced with log-barriers or by truncating inconsistent eigenvalues in the spectrum of the sparsified covariance matrix resulting from the relaxed problem [23].

Marginalization and sparsification techniques are used in single-robot SLAM frameworks to simplify the inference problem, especially by pruning the oldest variables of optimization sliding windows [51]. In multi-robot frameworks, it provides a convenient means to communicate condensed packets by projecting the information over commonly observed landmarks [22, 23, 32]. Marginalization and sparsification steps may be coupled to minimize subsequent communication costs [52].

7.2. Packet computation

The proposed method is inspired from *AUV-CSLAM* [23] and relies on the exchange of consistently marginalized and sparsified packets, augmented with a selection of raw 2D visual information to spot inter-robot correspondences.

7.2.1. Local marginalization

We first perform a local visual-inertial bundle adjustment in a local gravity-aligned frame of reference L. The pose of the first keyframe is kept constant for gauge fixing. The optimized variables Θ include all the keyframe poses $\mathcal{T}$ and inertial states $\mathcal{E}$, and the positions $\mathcal{L}$ of the observed landmarks. $\mathcal{Z}_{\mathrm{VI}}$ denotes the set of visual and inertial factors. The posterior distribution $p(\Theta|\mathcal{Z}_{\mathrm{VI}})$ is locally approximated as a Gaussian distribution in the neighbourhood of the resulting MLE estimate $\hat{\Theta}$:

$$p(\Theta|\mathcal{Z}_{\mathrm{VI}}) = \mathcal{N}\left(\boldsymbol{\xi}_{\hat{\Theta}}(\Theta); \mathbf{0}, \left[\boldsymbol{\mathcal{I}}_{\mathcal{Z}_{\mathrm{VI}}}^{\Theta}(\hat{\Theta})\right]^{-1}\right) \tag{9}$$

where $\boldsymbol{\xi}_{\hat{\Theta}}$ is the stacked residual w.r.t. the MLE estimates. $\boldsymbol{\mathcal{I}}_{\mathcal{Z}_{\mathrm{VI}}}^{\Theta}(\hat{\Theta})$ is the observed Fisher Information Matrix (FIM):

$$\boldsymbol{\mathcal{I}}_{\mathcal{Z}_{\mathrm{VI}}}^{\Theta}(\hat{\Theta}) = \sum_{z \in \mathcal{Z}_{\mathrm{VI}}} \left[\boldsymbol{J}_{\Theta}^{\boldsymbol{\xi}_z}(\hat{\Theta})\right]^{\top} \cdot \boldsymbol{\Sigma}_{\boldsymbol{\xi}_z}^{-1} \cdot \left[\boldsymbol{J}_{\Theta}^{\boldsymbol{\xi}_z}(\hat{\Theta})\right] \tag{10}$$

where $\boldsymbol{\xi}_z$, $\boldsymbol{\Sigma}_{\boldsymbol{\xi}_z}$ and $\boldsymbol{J}_{\Theta}^{\boldsymbol{\xi}_z}(\hat{\Theta})$ respectively denote the measurement residual, its covariance matrix, and its Jacobian matrix w.r.t. Θ evaluated at $\hat{\Theta}$. We marginalize $p(\Theta|\mathcal{Z}_{\mathrm{VI}})$ to yield the following distribution on the pose variables $\mathcal{T}$:

$$p\left(\mathcal{T}|\mathcal{Z}_{\mathrm{VI}}\right) = \mathcal{N}\left(\boldsymbol{\xi}_{\hat{\mathcal{T}}}(\mathcal{T}); \mathbf{0}, \left[\boldsymbol{\mathcal{I}}_{\mathcal{Z}_{\mathrm{VI}}}^{\mathcal{T}}(\hat{\mathcal{T}})\right]^{-1}\right) \tag{11}$$

where $\boldsymbol{\xi}_{\hat{\mathcal{T}}}(\mathcal{T})$ is the stacked absolute pose residual. The marginalized information matrix $\boldsymbol{\mathcal{I}}_{\mathcal{Z}_{\mathrm{VI}}}^{\mathcal{T}}(\hat{\mathcal{T}})$ is computed as the Schur's complement of the discarded variables:

$$\begin{aligned}\boldsymbol{\mathcal{I}}_{\mathcal{Z}_{\mathrm{VI}}}^{\mathcal{T}}(\hat{\mathcal{T}}) = &\left[\boldsymbol{\mathcal{I}}_{\mathcal{Z}_{\mathrm{VI}}}^{\Theta}(\hat{\Theta})\right]_{\mathrm{RR}} - \\ &\left[\boldsymbol{\mathcal{I}}_{\mathcal{Z}_{\mathrm{VI}}}^{\Theta}(\hat{\Theta})\right]_{\mathrm{RM}} \left[\boldsymbol{\mathcal{I}}_{\mathcal{Z}_{\mathrm{VI}}}^{\Theta}(\hat{\Theta})\right]_{\mathrm{MM}}^{-1} \left[\boldsymbol{\mathcal{I}}_{\mathcal{Z}_{\mathrm{VI}}}^{\Theta}(\hat{\Theta})\right]_{\mathrm{MR}}\end{aligned} \tag{12}$$

where the subscripts R and M respectively denote the sets of rows or columns associated to the retained and

marginalized variables. The resulting distribution is equivalent to a single dense $|\mathcal{T}|$-ary stacked absolute pose factor $z_{\hat{\mathcal{T}}}$ such that $p\left(z_{\hat{\mathcal{T}}}|\mathcal{T}\right) \triangleq p\left(\mathcal{T}|\mathcal{Z}_{\text{VI}}\right)$. The marginalization process is depicted by Figure 5a, compared to Figure 4a.

7.2.2. Consistent sparsification

The resulting marginalized factor $p(z_{\hat{\mathcal{T}}}|\mathcal{T})$ is far too complex to be communicated as such. As a second step, we thus take advantage on sparsification techniques to consistently approximate it using a set $\mathcal{Z}_S$ of uncorrelated Gaussian relative pose factors (as defined in equation (8) and shown in Figure 5b) between successive keyframes. Such *virtual* factors yield a distribution:

$$p(\mathcal{Z}_S|\mathcal{T}) = \prod_{z\in\mathcal{Z}_S} p\left(z|\mathcal{T}\right) = \mathcal{N}\left(\boldsymbol{\xi}_{\mathcal{Z}_S}(\mathcal{T}); \mathbf{0}, \boldsymbol{\Sigma}_{\boldsymbol{\xi}_{\mathcal{Z}_S}|\mathcal{Z}_S}\right) \quad (13)$$

where $\boldsymbol{\xi}_{\mathcal{Z}_S}(\mathcal{T})$ is the stacked relative pose residual, and $\boldsymbol{\Sigma}_{\boldsymbol{\xi}_{\mathcal{Z}_S}|\mathcal{Z}_S}$ is its block diagonal covariance matrix. The virtual factors must yield the same MLE estimate:

$$\hat{\mathcal{T}} = \underset{\mathcal{T}\in\mathcal{H}_\mathcal{T}}{\arg\max}\, p(\mathcal{Z}_{\text{VI}}|\mathcal{T}) = \underset{\mathcal{T}\in\mathcal{H}_\mathcal{T}}{\arg\max}\, p(\mathcal{Z}_S|\mathcal{T}) \quad (14)$$

whose locally associated Gaussian distribution is:

$$p(\mathcal{T}|\mathcal{Z}_S) = \mathcal{N}\left(\boldsymbol{\xi}_{\hat{\mathcal{T}}}(\mathcal{T}); \mathbf{0}, \left[\boldsymbol{\mathcal{I}}^{\mathcal{T}}_{\mathcal{Z}_S}(\hat{\mathcal{T}})\right]^{-1}\right) \quad (15)$$

with the following observed Fisher Information matrix:

$$\boldsymbol{\mathcal{I}}^{\mathcal{T}}_{\mathcal{Z}_S}(\hat{\mathcal{T}}) = \left[\boldsymbol{J}^{\boldsymbol{\xi}_{\mathcal{Z}_S}}_{\mathcal{T}}(\hat{\mathcal{T}})\right]^\top \cdot \boldsymbol{\Sigma}^{-1}_{\boldsymbol{\xi}_{\mathcal{Z}_S}} \cdot \left[\boldsymbol{J}^{\boldsymbol{\xi}_{\mathcal{Z}_S}}_{\mathcal{T}}(\hat{\mathcal{T}})\right] \quad (16)$$

The covariance matrices of the virtual factors are found as the solution of an optimization problem which minimizes the information loss under a consistency constraint:

$$\boldsymbol{\Sigma}^*_{\boldsymbol{\xi}_{\mathcal{Z}_S}} = \underset{\boldsymbol{\Sigma}_{\boldsymbol{\xi}_{\mathcal{Z}_S}}\in\mathcal{D}_+}{\arg\min}\ \mathcal{D}_{\text{KL}}\left(p(\mathcal{T}|\mathcal{Z}_{\text{VI}}), p(\mathcal{T}|\mathcal{Z}_S)\right) \quad (17a)$$

$$\text{subjected to } \boldsymbol{\mathcal{I}}^{\mathcal{T}}_{\mathcal{Z}_S}(\hat{\mathcal{T}}) \leq \boldsymbol{\mathcal{I}}^{\mathcal{T}}_{\mathcal{Z}_{\text{VI}}}(\hat{\mathcal{T}}) \quad (17b)$$

where $\mathcal{D}_{\text{KL}}$ denotes the Kullback-Leibler divergence, $\mathcal{D}_+$ is the set of block diagonal positive definite matrices which matches the desired sparsified topology. Mazuran et al. [48] showed that this problem, relaxed from the consistency constraint, has a closed-form solution as the Jacobian matrix $\boldsymbol{J}^{\boldsymbol{\xi}_{\mathcal{Z}_S}}_{\mathcal{T}}(\hat{\mathcal{T}})$ of the sparsified topology is invertible. The dense covariance matrix is first projected onto the space of the new relative pose factors $\mathcal{Z}_S$:

$$\boldsymbol{\Sigma}_{\boldsymbol{\xi}_{\mathcal{Z}_S}|\mathcal{Z}_{\text{VI}}} = \left[\boldsymbol{J}^{\boldsymbol{\xi}_{\mathcal{Z}_S}}_{\mathcal{T}}(\hat{\mathcal{T}})\right]\left[\boldsymbol{\mathcal{I}}^{\mathcal{T}}_{\mathcal{Z}_{\text{VI}}}(\hat{\mathcal{T}})\right]^{-1}\left[\boldsymbol{J}^{\boldsymbol{\xi}_{\mathcal{Z}_S}}_{\mathcal{T}}(\hat{\mathcal{T}})\right]^\top \quad (18)$$

The projected covariance matrix $\boldsymbol{\Sigma}_{\boldsymbol{\xi}_{\mathcal{Z}_S}|\mathcal{Z}_{\text{VI}}}$ is then marginalized onto each new virtual factor $z \in \mathcal{Z}_S$ to get its covariance matrix $\boldsymbol{\Sigma}_{\boldsymbol{\xi}_z|\mathcal{Z}_{\text{VI}}}$. The transformations applied to the covariance matrices are illustrated by Figure 6. However, such factors are still correlated as each one summarizes the whole information of the original distribution.

We finally need to impose the consistency constraint (17b) from the solution of the relaxed problem. Classical methods [53] based on the solving of Semi-Definite Programming problems by interior point methods and which model the consistency constraint with log-det barriers are intractable in real-time. We therefore use a non-optimal method to enforce this constraint with less complexity. The consistency constraint can be reformulated with covariances:

$$\boldsymbol{\mathcal{I}}^{\mathcal{T}}_{\mathcal{Z}_S}(\hat{\mathcal{T}}) \leq \boldsymbol{\mathcal{I}}^{\mathcal{T}}_{\mathcal{Z}_{\text{VI}}}(\hat{\mathcal{T}}) \Leftrightarrow \boldsymbol{\Sigma}_{\boldsymbol{\xi}_{\mathcal{Z}_S}|\mathcal{Z}_S} \geq \boldsymbol{\Sigma}_{\boldsymbol{\xi}_{\mathcal{Z}_S}|\mathcal{Z}_{\text{VI}}} \quad (19)$$

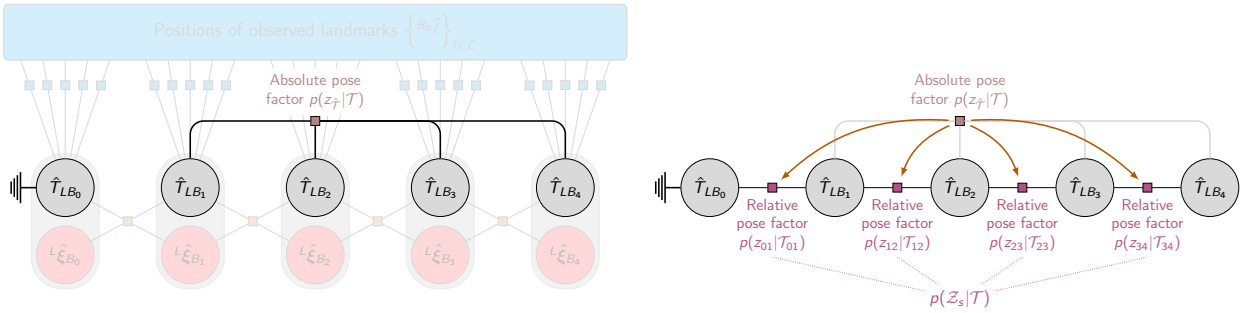
We compute the largest eigenvalue $\lambda_{\max}$ of $\boldsymbol{\Sigma}_{\boldsymbol{\xi}_{\mathcal{Z}_S}|\mathcal{Z}_{\text{VI}}}$. This can be efficiently achieved using for instance the Lanczos algorithm [54]. Then, we iterate over each factor $z \in \mathcal{Z}_S$ and perform an eigen-decomposition of its marginalized covariance matrix $\boldsymbol{\Sigma}_{\boldsymbol{\xi}_z|\mathcal{Z}_{\text{VI}}} = \boldsymbol{Q}\cdot\text{diag}(\boldsymbol{\lambda}_z)\cdot\boldsymbol{Q}^\top$ with $\boldsymbol{\lambda}_z$ the vector of its eigenvalues and $\boldsymbol{Q} \in \mathbb{SO}_6$. Two alternative truncation techniques are possible. The first one replaces by $\lambda_{\max}$ each eigenvalue of $\boldsymbol{\lambda}_z$ which is lower while the second expands $\boldsymbol{\lambda}_z$ by a factor $\alpha \geq 1$ such that $\min(\alpha\cdot\boldsymbol{\lambda}_z) = \lambda_{\max}$. This preserves the relative uncertainties between the residuals. To account for the fact that each eigenvalue will indifferently contribute to the orientation and translation residuals, which exhibit different magnitudes and units, the dilatation is carried out in a *normalized* space, by working on the correlation matrix.

7.2.3. Finalizing the packet

The packet is completed by adding a selection of raw visual information following the procedure described in §5.3 and Algorithm 1. The final packet consists of the computed relative pose virtual factors and the communicated pruned visual frames. The camera model is appended to the first packet as it is required to estimate relative poses from spotted inter-robot correspondences. The computed relative pose factors are recorded in the map, and will be used for global inference along with the received factors.

7.3. Packet integration

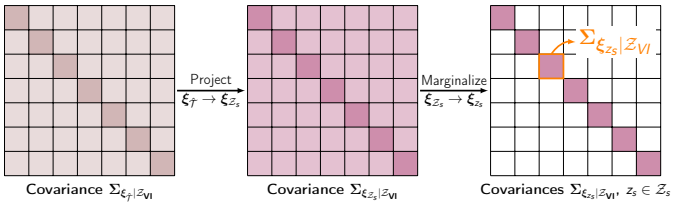
Once the trajectory is initialized using the relative pose factors, we search for inter-robot correspondences. Inter-robot correspondences are spotted by querying the received visual frames into the receiver's visual database as exposed in section §6.1. For instantaneous correspondences, the receiver's new keyframes are temporally stored into a local visual database of limited capacity, which we query with all the received visual frames when it gets full. The local visual database is then re-initialized. When a robot spots an inter-robot correspondence using a received visual frame, it can request additional visual information over the neighboring visual frames to the corresponding robot. It then receives the keypoints coordinates and descriptors associated to the highly covisible keyframes. To mitigate the communication cost of such requests, only a subset of the keypoints is communicated, using the same selection process as described in section §5.3.



(a) Marginalized graph: marginalization yields an equivalent n-ary factor $z_{\hat{\mathcal{T}}}$ over the remaining pose variables $\mathcal{T}$ encoding the likelihood $p(\mathcal{T}|\mathcal{Z}_{\text{VI}})$.

(b) Sparsified graph: sparsification approximates the dense distribution $p(\mathcal{T}|\mathcal{Z}_{\text{VI}})$ as $p(\mathcal{T}|\mathcal{Z}_S) \propto \prod_{z_s \in \mathcal{Z}_S} p(z_s|\mathcal{T})$.

Figure 5: Marginalization and Sparsification processes



The dense covariance matrix is first projected onto the virtual factors' space, an then marginalized onto each virtual factor's subspace.

Figure 6: Solving the relaxed problem

8. Pruned Visual-Inertial Packets (PVIP)

8.1. Related works

A second strategy to synthetize visual-inertial information is to summarize the localization information through factor pruning. It seeks to approximate the original visual-inertial distribution as compactly as possible without excessively compromising the reliability of the resulting MLE estimate in terms of accuracy and uncertainty. It does not face the consistency issue met by condensation techniques, and boils down to a NP-hard resource-constrained optimal subset selection problem :

$$\mathcal{Z}_S^* = \underset{\mathcal{Z}_S \subseteq \mathcal{Z}}{\arg\max}\, u(\Theta, \mathcal{Z}_S) \text{ s.t. } c(\Theta, \mathcal{Z}_S) \geq 0 \qquad (20)$$

where u quantifies the utility of the set of selected factors $\mathcal{Z}_S$ w.r.t. some interest variables Θ, and c encodes a selection constraint. The utility function should ideally promote the synergistic selection of informative factors while penalizing the joint selection of mutually redundant ones.

The information theory provides adequate tools to measure the redundancy between variables, such as mutual information [55] and relative entropy [56]. However, they result into complex metrics, which can be approached by simpler correlated heuristic, geometric [7] or statistical metrics [18]. Building on graph theory, Khosoussi et al. [57] related the reliability of the MLE estimate to the number of spanning trees in the pose graph and designed a subsequent graph pruning strategy.

The second purpose of the utility function is to quantify how a subset $\mathcal{Z}_S$ of factors contributes to the estimation of interest variables Θ like trajectory variables. Joint utility functions may once again benefit from information-theoretic tools to derive landmark selection schemes which bound the localization uncertainty along the trajectory [58]. Zhao et al. [59] coined a max-LogDet criteria to select landmarks which best condition the least square pose estimation problem. Finally, Dymczyk et al. [60] proposed an optimization framework for landmark sparsification under limited budget constraints in the context integer linear and quadratic programming.

In practice, it may be approximated as a sum of individual contributions. Landmarks are then ranked according to their information gain to spot the most informative landmarks for localization [61] or collision-avoidance [62]. The utility of each landmark can be learnt using statistical regression models [63], based on simple criteria including tracking, reprojection and appearance. Dymczyk et al. [64] used an iterative reduction algorithm alternating between sampling and scoring, based on the number of observer keyframes. In the context of Structure from Motion, Cao et al. [65] proposed a probabilistic Min-K-Cover (MKC) algorithm to ensure a probabilistic k-cover while maximizing the associated descriptor variance.

8.2. Packet computation

The proposed method resembles the previous one but relies on landmark pruning formulated as a set cover problem. It makes robots regularly exchange pruned packets. Each packet consists of raw inertial factors and selected landmark tracklines to convey the localization information on the trajectory poses, and of raw 2D visual information on selected keyframes.

8.2.1. Subgraph extraction

Contrary to CVIP, we consider a larger subgraph which includes some keyframes communicated in the previous packet but which have a covisibility ratio $r \geq r_{\text{common}}$ with the first new keyframe, as illustrated in Figure 7. The objective is to mitigate border effects during the selection of the landmarks and enforce trackline overlapping between successive packets. Furthermore, no local optimization nor covariance extraction is needed.

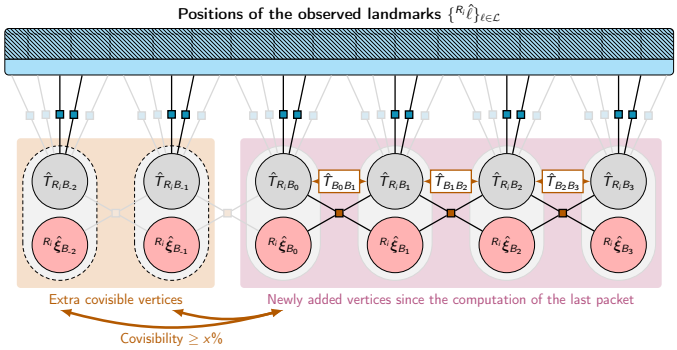


Figure 7: Structure of the underlying visual-inertial factor graph of a PVIP packet, to be compared with Figure 4a

8.2.2. Appending trajectory and inertial measurements

The first step is to append the trajectory and inertial information to the packet. For each keyframe $\mathcal{K}_i$, we append its current relative pose estimate $T_{\mathcal{K}_{\text{i-1}}\mathcal{K}}$ with its previous keyframe $\mathcal{K}_{\text{i-1}}$. Only inertial factors posterior to the previous packet are actually appended despite of the subgraph's overlapping.

8.2.3. Landmark selection

Then, we select a subset of the landmarks which ideally maximizes the information on the pose of the keyframes. The first method models the selection as a set k-cover problem. It consists of computing a minimal subset of well-constrained landmarks such that each keyframe is covered with at least k landmark observations. This is a NP-complete problem whose solution can be approximated using a greedy sampling and scoring algorithm. At each iteration, we select the landmark ℓ_i which covers the most partially covered keyframes. In the case of multiple optimal landmarks, we select the one with the most observations. Each landmark is selected with its full trackline, which may result in supernumerary covers on some keyframes. Then, the selected landmark is deleted from the pool of candidates and the covering scores for keyframes and landmarks are updated. We also delete all the landmarks ℓ_j which show redundancies to ℓ_i which we quantify using the quantity $\mathcal{R}_{\text{ij}}$, inspired from [60]:

$$\mathcal{R}_{\text{ij}} = \frac{1}{|\mathcal{K}_{\text{ij}}|} \sum_{k\in\mathcal{K}_{\text{ij}}} \frac{\min\left(d_{\min}, d(\boldsymbol{p}_{k,i}, \boldsymbol{p}_{k,j})\right)}{d_{\min}} \leq \mathcal{R}_{\min} \qquad (21)$$

where $\mathcal{K}_{\text{ij}}$ denotes the set of the keyframes onto which ones they both project, and $\boldsymbol{p}_{k,\bullet}$ stands for their projection on the k$^{\text{th}}$ keyframe; $d_{\min}$ is a threshold and normalization distance and $\mathcal{R}_{\min} \in [0,1]$ is a user-defined threshold to characterize redundancy. The algorithm stops as soon as all keyframes are covered with at least k landmark observations or no remaining landmark covers a partially covered keyframe. We call this method PVIP-MKC.

The iterative sampling and scoring method proposed by Dymczyk et al. [64] provides an alternative suitable solution for solving the above set cover problem as an integer-based optimization problem [60]:

$$\mathbb{I}^*_{\mathcal{L}_S} = \underset{\mathbb{I}_{\mathcal{L}_S}\in\{0,1\}^N}{\arg\min} \quad \boldsymbol{q}^\top \cdot \mathbb{I}_{\mathcal{L}_S} \text{ s.t. } \boldsymbol{A}\cdot\mathbb{I}_{\mathcal{L}_S} \geq \boldsymbol{k} \qquad (22)$$

where $\mathbb{I}_{\mathcal{L}_S}$ is a binary indicator vector whose i$^{\text{th}}$ variable indicates whether the i$^{\text{th}}$ landmark is selected into $\mathcal{L}_S$ or not. The i$^{\text{th}}$ coefficient of $\boldsymbol{q}$ is the inverse observer count for the i$^{\text{th}}$ landmark. The cover constraint is encoded by a visibility matrix $\boldsymbol{A}$ defined such that $A_{\text{ij}} = 1$ if keyframe i observes landmark j and $A_{\text{ij}} = 0$ otherwise. $\boldsymbol{k}$ is the target cover vector. As the optimization is already performed on the bounded extracted subgraph, no preliminary partitioning is needed. We refer to this approach as PVIP-ILP.

8.2.4. Finalizing the packet

The visual information is appended the same way as it was done for CVIP. The final packet includes the consecutive inertial factors, the selected landmark tracklines, as well the selected keypoints and descriptors. We might need to fuse landmarks and keypoints on the already received extra vertices. The current relative pose estimates between the consecutive keyframes are also appended to the packet. In the first packet, we also include the IMU and camera model coefficients which are needed for inference and the detection of inter-robot correspondences.

8.3. Packet integration

Upon reception, the trajectory is reconstructed from the relative pose estimates, which we then use to initialize the inertial states and triangulate the communicated landmarks. Inter-robot correspondences are spotted following the same process as CVIP. Global multi-robot inference is performed as a reduced visual-inertial bundle-adjustment over the selected landmarks. However, such inference is run as a Back-End process. To quickly impact new correspondences, we formulate a pose-graph relaxation problem which we build over the new correspondences and by enforcing priors on the current relative pose estimates along the trajectories and the previously included correspondences, and we only consider unit-covariance matrices.

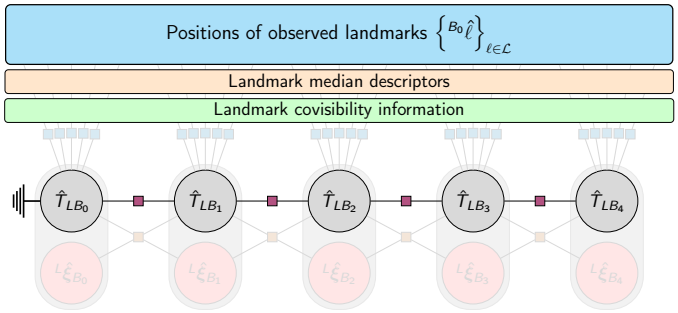
9. Submap Visual-Inertial Packets (SVIP)

9.1. Related works

When sharing data with other robots, a third and more classical solution involves transmitting rigid submaps, holding extensive visual-structural information on the observed environment. Global consistency is then approached by mutually registering the locally consistent submaps. The sub-mapping approach has been used in decentralized collaborative SLAM: Schuster et al. [24] make robots exchange dense stereo-visual point clouds registered through ICP, Sartipi et al. [25] exchange and update current visual-inertial submaps for collaborative localization and [66] exchange submaps associated to Truncated Signed Distance Functions for dense mapping.

9.2. Packet computation

The proposed method relies on the exchange of visual-inertial submaps. In each packet, the localization information is summarized into a condensed factor graph using local marginalization and sparsification techniques as presented in §7. However, no consistency constraint was enforced on those packets. The particularity of SVIP lies in the way it communicates visual-structural information. We take advantage on the rigid point cloud associated to the submap to encode visual and structural information required for place recognition and correspondence characterization. The positions of well-constrained landmarks are expressed relatively to the body frame of the first keyframe of the submap. We associate each landmark with its most consensual (median) descriptor. Landmarks with high descriptor variance are ignored as they may be unstable or result from incorrect data association.



A SVIP packet includes virtual relative pose factors, and a rigid point cloud of landmarks resulting from local optimization, along with descriptor and a landmark covisibility table as required to spot and filter correspondences.

Figure 8: Content of a SVIP packet, compared with Figure 4a

Finally, inspired by the structure of the summary maps described by Lynen et al. [42], we also append a landmark covisibility table: each landmark is associated with a list of the indexes of all its observer keyframes within the submap, but the coordinates of the associated 2D keypoints are not communicated. Such information is necessary to filter outlier candidate keyframes for correspondences. The final packet is represented in Figure 8. Once the submap is completed, we replace the portion of the original map with the computed submap but we keep the original frames whose 2D visual information is needed to spot correspondences.

9.3. Packet integration

The received submap is reconstructed and connected to the previously received submaps. Indirect retrospective inter-robot correspondences are spotted by querying a subset of the recipient robot's keyframes against the visual database associated to the received submap. Those query keyframes are selected along the recipient robot's trajectory using Algorithm 1. If a correspondence is found, then neighboring keyframes are also queried. Finally, the submap visual database is appended to the global visual database attached to the sender robot. When adding a new keyframe on the receiver's trajectory, it is queried against the visual databases of the other robots.

10. Performance evaluation

The proposed methods aim at enforcing the autonomy of the robots while meeting the requirements derived from informational, communication and embeddability constraints. While the autonomy regarding the detection of inter-robot correspondences and multi-robot inference is achieved by construction, the suitability of the proposed architectures and data exchange methods w.r.t. those constraints must be assessed.

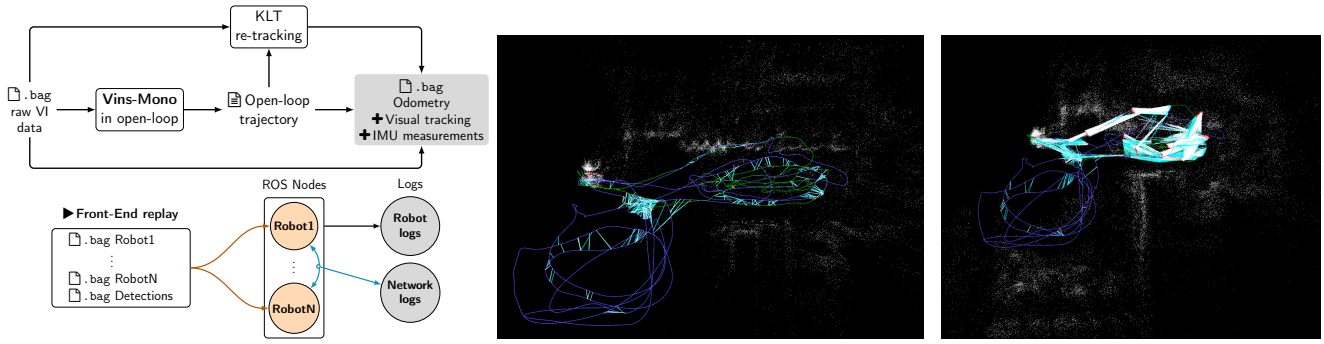
10.1. Test scenarios

We evaluated the proposed methods on multiple multi-robot scenarios. We first used the EuRoC dataset [1] to build 4 multi-robot scenarios by synchronizing individual *Machine Hall* sequences, as detailed in table 1a. However, the EuRoC sequences were primarily designed for single-robot SLAM, and consequently, trajectories are already self-sufficient in terms of loop closures. To properly stage the specific challenges of online collaborative SLAM, we designed the AirMuseum dataset [2]. It consists in five heterogeneous multi-robot scenarios with aerial and terrestrial sequences whose properties are displayed in Table 1b. Their trajectories were jointly designed such that they accumulate a significant odometric drift which would be correctable online based on the temporal and spatial distribution of informative inter-robot direct and indirect correspondences. The additional objective was to evaluate the proposed methods in a heterogeneous context including terrestrial robots and test the limits of the proposed methods on more complicated scenarios than the one built from the EuRoC sequences (e.g. less correspondences, lower signal-to-noise ratio, IMU excitations and visual richness on the terrestrial sequences, lower IMU quality, etc.).

10.2. Implementation and simulation details

Simulations were carried out using the *ROS* middleware on an Inter® Xeon(R) W-123 CPU 3.60 GHz × 8 processor. We simulated the outputs of an elementary SLAM Front-End by using the open loop trajectory estimates of the Vins-Mono visual-inertial estimator. We then performed a KLT retracking [67] over the trajectory and the input images. The output was then dumped into a `.bag` file with custom keyframe messages including visual frame information, landmark tracklines, inertial states and relative pose estimates between consecutive keyframes.

The simulation were carried out as a turn-by-turn process based on the chronological iteration of all the keyframes and direct observation ROS messages extracted from the provided `.bag` files as described by Algorithm 2 and Figure 9a. We implemented each robot as a ROS node with its own topic publishers and subscribers to simulate data sending and reception. Robot manage their own map using



(a) Multi-robot simulation architecture (b) CVIP simulated on MH123 (c) SVIP simulated on MH123

Figure 9: Simulation architecture and visuals

Seq.	Distance [m]	Duration [s]	Scenarios 12	13	45	123
MH1	80.6	182	✖	✖		✖
MH2	73.5	150	✖			✖
MH3	130.9	132		✖		✖
MH4	91.7	99			✖	
MH5	97.6	111			✖	

Multi-robot scenarios built from the EuRoC dataset are given names of the form MH{XYZ} according to the involved sequences. For instance, MH12 denotes the scenario which consists of sequences MH1 and MH2.

(a) Multi-robot scenarios built from EuRoC [1]

Scenarios		**#1**	**#2**	**#3**	**#4**	**#5**
Duration	[s]	445	387	304	316	304
Trajectory length						
Robot A	[m]	229	116	147	182	141
Robot B	[m]	225	141	143	120	178
Robot C	[m]	191	162	202	222	131
Drone	[m]	—	—	203	215	204

Robots A, B and C are terrestrial Wifibots^{TM} and the drone is a DJI-M100^{TM}. All robots carry a stereo camera bench and an IMU.

(b) AirMuseum multi-robot scenarios

Table 1: Properties of the multi-robot scenarios

the Maplab framework [43], and handle dedicated modules to add new keyframes, perform detect intra and inter-robot correspondences, perform multi-robot inference, build, exchange and integrate packets and anchor reference frames. Figures 9b and 9c give some example visuals of the respective simulations of the CVIP and the SVIP methods on the MH123 scenario.

The presented methods share some common parameters. The first one is the minimum traveled length $\ell_{\min}$ between two consecutive packets, which tunes the amount of information contained in the packets versus their computation complexity. It especially affects CVIP and SVIP since the processing requirements of the marginalization step drastically increase with the size of the packets. In the following simulation, we empirically set $\ell_{\min} = 5m$ as a good compromise. The second family of parameters include the covisibility ratio $q_{\min}$ and threshold $n_{\min}$ for

Algorithm 2– Multi-robot turn-by-turn simulation

```
foreach msg in sorted_messages do
    update_clock(msg);
    robot ← robots[msg→robot_id]
    if msg.type = keyframe then
        Process subscribers' callbacks queues
        robot.add_received_packets();
        robot.add_received_correspondences();
        robot.process_frame_requests_and_responses();
        Add the new keyframe
        robot.add_new_keyframe_to_the_map(msg);
        robot.compute_new_packet_if_required();
        robot.detect_intra_robot_loop_closures();
        robot.detect_inter_robot_correspondences();
        robot.anchor_reference_frames_if_possible();
        robot.perform_inference_if_required();
    else if msg.type = direct_correspondence then
        robot.direct_correspondence_topic.publish(msg);
    Run the turn of the robots which have stayed inactive
      more than a given time period;
```

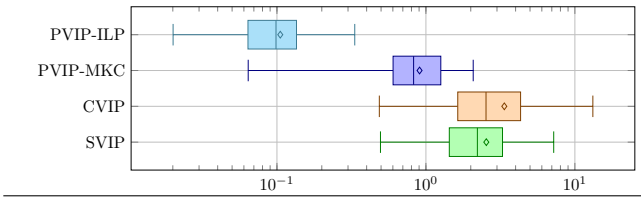
the selection of the keyframes in CVIP and PVIP. The third one is the number n_{kp} of keypoints and descriptors to retain on each selected keyframe, and which is used in the three methods. Since visual information accounts for most of the weights of the exchanged packets, those parameters should be tuned so that they allow to spot inter-robot correspondences without degrading the communication cost too much. We empirically set $q_{\min} = 20\%$, $n_{\min} = 20$ and $n_{kp} = 100$. The last family of parameters is specific to PVIP and includes the landmark target cover k and the maximum allowed redundancy threshold $\mathcal{R}_{\min}$. We set $\mathcal{R}_{\min} = 0.5$. We experiment different values for k, which should be chosen large enough such that the keyframe poses are sufficiently constrained for BA. If not explicitly specified, $k = 20$.

10.3. Assessing embeddability constraints

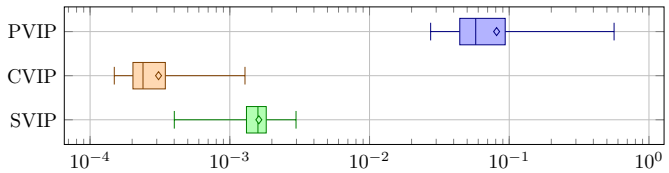
The distribution of packet computation times on the EuRoC scenarios are shown in Figure 10a. We note that they range from a few tenth of seconds to a few seconds, which suits to real-time requirements. We note that SVIP and CVIP methods require more processing time than

PVIP. This results from the local marginalization and sparsification process which require local inference and Fisher information matrix calculus and inversion. On the contrary, PVIP packets are exempted from any optimization step. We note that the ILP selection procedure outruns the MKC selection, what may be mostly explained by the need to first build a redundancy table of the $\mathcal{R}_{ij}$ indices between pairwise covisible landmarks.

(a) Packet computation times (s)



(b) Packet integration times (s)

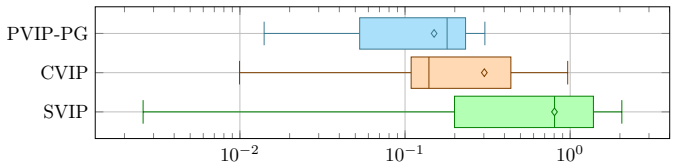


Results are similar for PVIP-ILP and PVIP-MKC without significant differences w.r.t. the target cover.

Figure 10: Distribution of packet computation and integration times

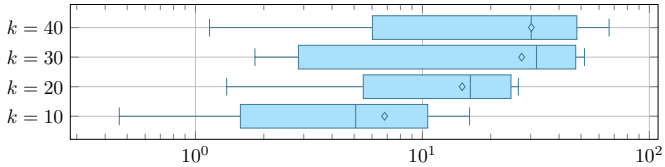
Packet integration times are displayed in Figure 10b. CVIP packets are the fastest packets to be integrated, in a few tenth of milliseconds, as they just require to extend the trajectory with the exchanged relative pose factors and rebuild the visual frames. SVIP requires additional time to rebuild the landmark point-cloud. PVIP packets require more time to be integrated – but still less than a tenth of second – as they need to re-initialize the inertial states and re-triangulated the landmarks based on their tracklines.

(a) Multi-robot pose-graph optimization times (s)



PVIP-PG denotes the pose-graph relaxation introduced in §8.

(b) Multi-robot reduced BA optimization times (s)



Processing times are displayed for different target cover k.

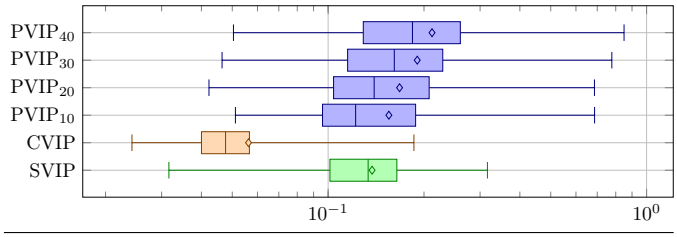
Figure 11: Distribution of processing times

Multi-robot pose-graph inference durations are shown in Figure 11a. Once again, we note that such processes are compatible with real-time requirements as their processing times range from a tenth of second to a second. Pose-graph optimizations for CVIP and SVIP, and pose-graph relaxation for PVIP, allow to quickly impact new intra and inter-robot correspondences than CVIP and PVIP. SVIP generally requires more time for pose-graph inference as it spots more correspondences. In the case of PVIP, pose-graph relaxations are completed with reduced visual-inertial bundle-adjustements, which last from a few seconds to a few dozen of seconds, depending on the target cover, as shown in Figure 11b.

10.4. Assessing communication constraints

The second aspect to evaluate is the induced network load. The distribution of the weights of the computed packets are displayed in Figure 12a while Figure 12b decomposes the content of the packets. CVIP outputs the lightest packets, with a median weight around 60 kBytes, since they only include the computed relative factors, augmented with local selections of keypoints and descriptors on a subset of the keyframes and which explain 80% of the weight of the packets. SVIP and PVIP packets are much heavier, with median weights between 0.1 and 0.2 MBytes. Most of the weight of a SVIP packet is explained by its visual-structural information. Regarding PVIP, the weight of the packet split between its inertial factors and its visual frame information (which also include the tracklines of the selected landmarks).

(a) Packet sizes (MBytes)



(b) Packet decomposition (MBytes)

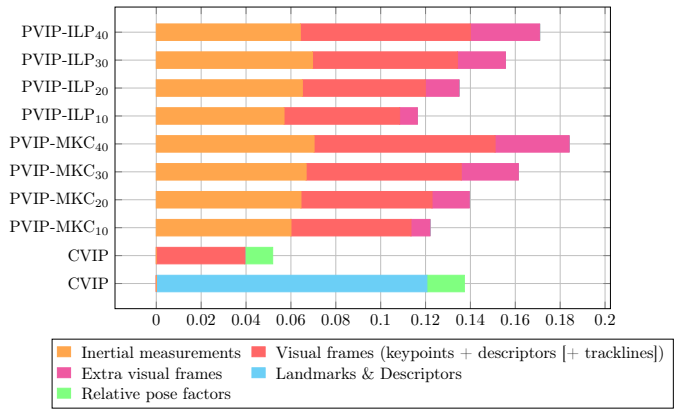


Figure 12: Packet sizes and decomposition

Packets are not the only items to be exchanged between the robots. Figure 14 shows the accumulation of ex-

changed data over time in the three-robot scenario MH123 between the three robots. We notice that most of the induced communication load is explained by the broadcasting of the computed packets, whose bandwidth requirements scale linearly with the number of robots, apart from regularization processes. Figure 13 shows the averaged bandwidth consumption associated to each method. In all methods, the average consumed bandwidth barely exceeds 100 kB/s. We note that the broadcasting of the spotted correspondences explains a significant portion of the consumed bandwidth in SVIP, as sharing visual-structural information helps spotting more correspondences.

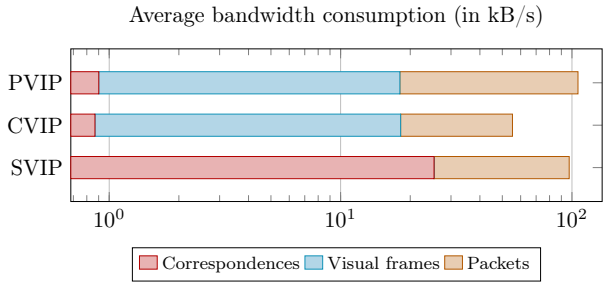


Figure 13: Average bandwidth consumption in MH123.

10.5. Assessing informational constraints

Finally, we need to evaluate how the proposed methods deal with the above-mentioned informational constraints. We first assess how well the computed packets manage to preserve the original visual-inertial information. For such purpose, we compute the Kullback-Leibler divergence between the distribution induced on the absolute pose estimates by the original visual-inertial factors $\mathcal{Z}_{\text{VI}}$ and the one derived from the compute packet factors $\mathcal{Z}_{\text{packet}}$. Such results are reported in Figure 15.It shows that both PVIP MKC and ILP packets are the ones which best minimize the loss of information, with one order of magnitude of difference with CVIP and SVIP. The influence of the target cover appears explicitly.

Additionally, Figure 16 displays the distribution of the eigenvalues of the difference of the covariance matrices of the original and the packet distributions. The resulting packets are consistent if and only if all those eigenvalues are positive, and all the less conservative as they are low. Thus, even though SVIP, which implements a version of CVIP relieved from the consistency constraint, seems to better preserve the original information, it clearly appears that it double counts the original information in the resulting distribution, and thus adds artifactual information spread between correlated factors. Figure 16 also clearly shows the impact of the proposed normalization step in the sparsification process, which results into a "lower" covariance matrix difference. Finally, it unsurprisingly confirms that PVIP performs better than SVIP and CVIP not only in minimizing the information loss, but also in ensuring the consistency of the final packets. In particular, it is less conservative than CVIP.

Even though individual packets are consistent and independent from each other, inconsistencies may still occur in the global map since the correlations between the inter-robot correspondences whose underlying PnP problems involve common landmarks are ignored.

Finally, we evaluate the multi-robot estimation accuracies. Figure 17 reports the translation estimation errors for each robot in each scenario on its own trajectory in the single-robot case (red), and in the multi-robot cases for each method. Each robot also estimates the trajectories of the other robots, with very similar accuracies of their associated self-estimation. [1]

10.5.1. Evaluation of the EuRoC scenarios

Results on the EuRoC scenario are reported in Figure 17a. As previously mentioned, EuRoC trajectories were intentionally made self-sufficient in terms in loop closures, which results into good single-robot accuracies, except on MH3 whose trajectory shows a much ampler kinematics.

First, we note that SVIP performs poorly on those scenarios, and significantly degrades the estimation accuracies with wider error distributions. This is particularly blatant in scenario MH45. Two factors may explain such poor performances. First, this may be explained by a poorer quality of the inter-robot correspondences spotted using the exchanged rigid visual-structural information, and whose relative pose factors are then expressed w.r.t. to the first vertex of the packet. This may especially happen when the matched section of the packet's point cloud relates to the packet's last keyframes. Furthermore, SVIP detects at once bunches of inter-robot correspondences and yields a resulting topology which enforces strong and sudden constraints solely on the first vertex of the spotted packet. This seems to compromise the online integration of such correspondences during the subsequent pose-graph optimization. It was notably the case on the MH45 scenario, what amplified an estimation error which could not be recovered afterwards.

Contrary to SVIP, CVIP shows encouraging results on all the scenarios as it generally allows to tighten the error distribution and reduce the average and median estimation errors. This is especially the case in scenario MH13 for Robot 3, which benefits from the integration of Robot 1's packets to reduce its median estimation error of nearly 14cm. However, Robot 1 fails to properly absorb Robot 3's estimation errors conveyed in its packets, which slightly degrades its estimation accuracy. This suggests that some

[1] Note that, as described in section §10.2, the reported performances for single-robot SLAM are those of the SLAM algorithm whose Front-End builds on the the open-loop trajectory estimated by Vins-Mono, and whose Back-End builds on the loop-closure and optimization tools provided by Maplab [43]. We advocate it allows a fairer comparison between the mono and multi-robot scenarios which thus share the very same underlying SLAM algorithm. The performances of the actual Vins-Mono algorithms on the EuRoC and AirMuseum sequences are respectively reported in [10] and [2].

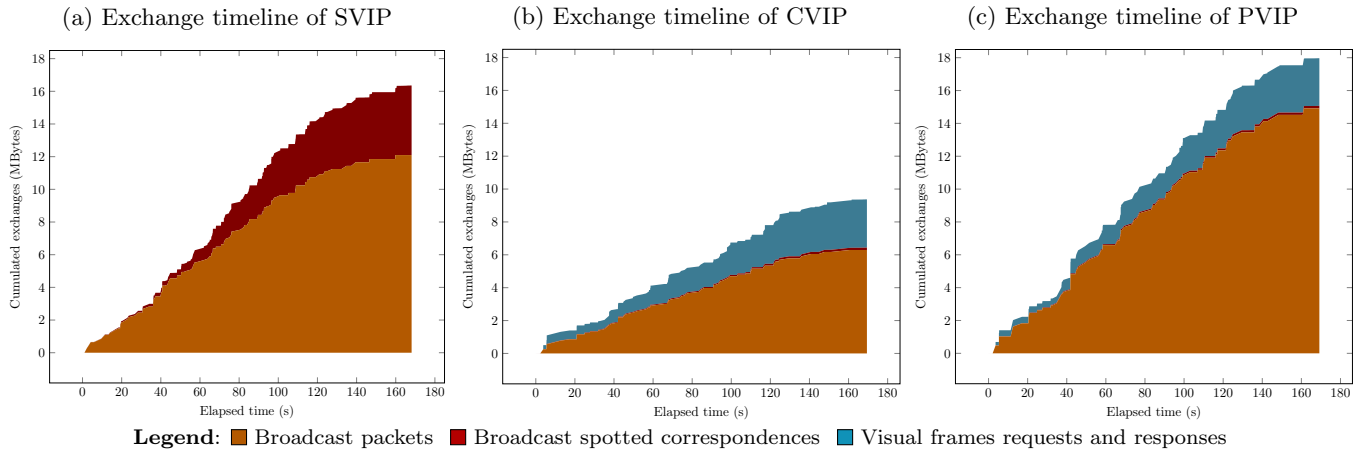


Figure 14: Cumulated exchanges (MBytes) timelines (s) in scenario MH123

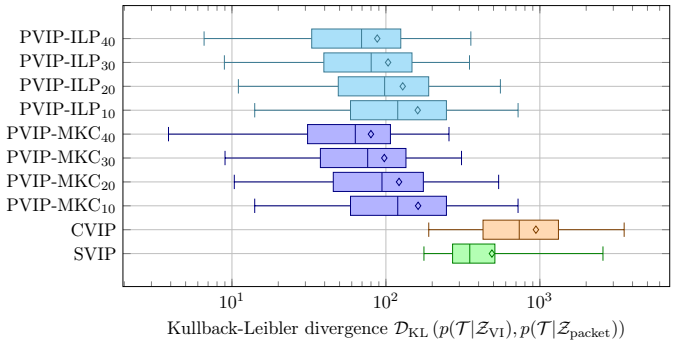


Figure 15: Evaluation of the packets' relative entropy

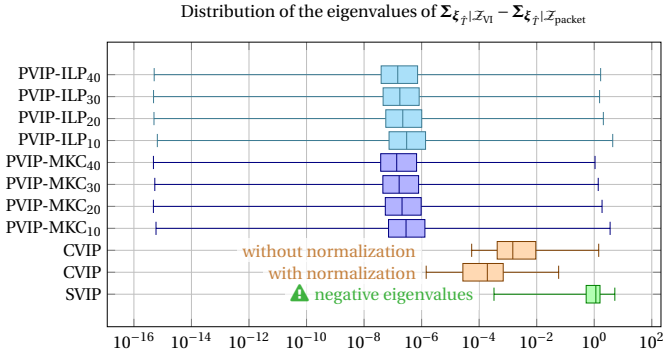


Figure 16: Evaluation of the packets' consistency

additional regularization mechanisms should frame the integration of the received packets to specifically deal with prior disparities in estimation accuracy.

Finally, PVIP performs better than CVIP and SVIP, as it improves the estimation accuracy for all the robots in all the EuRoC scenarios. It is notably the only methods which achieves to improve the accuracies for Robot 1 while its trajectory already benefits from several informative loop closures. Interestingly, we observe MKC and ILP versions perform similarly, and that the marginal utility of additional landmark cover is low as we achieve comparable performances with the different covers. PVIP clearly benefits from its two-level inference process with pose-graph relaxation and reduced visual-inertial Bundle Adjustement. Furthermore, the EuRoC scenarios provide an ideal framework for PVIP as it is a richly textured environment and IMU sensors are excited enough to make its biases observable during the packet integration.

10.5.2. Evaluation on the AirMuseum scenarios

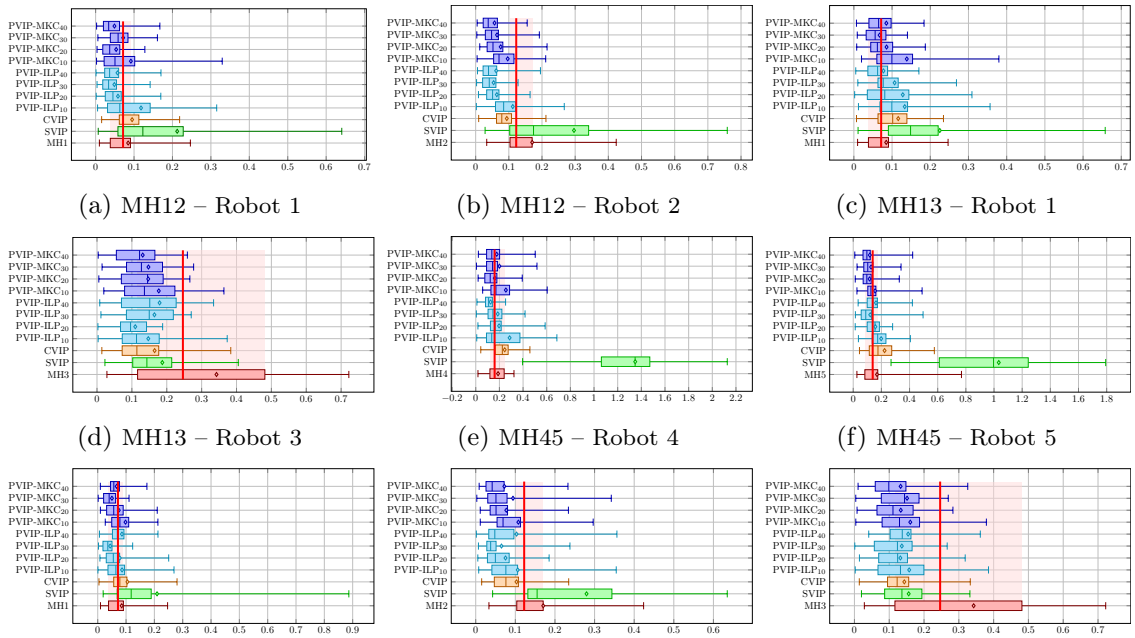
Results on the AirMuseum scenarios are reported in Figure 17b. Such scenarios allow to evaluate the proposed methods in an heterogeneous context, and confront them with the specific challenges of terrestrial VI SLAM (i.e. more agressive kinematics with jerk movements, higher noise to vibration ratio, poorlier textured ground observations, less intra-robot loop closures, etc.). The reported single-robot estimation accuracies suggest the difficulty of those sequences to VI-SLAM.

A first observation is that SVIP performs better in improving the estimation accuracies than it did on the EuRoC scenarios. One reason may be that fewer inter-robot correspondences are spotted, what enforces a more favorable constraint pattern than in the EuRoC scenarios. Furthermore, the inconsistency of the packet allows to yield stronger relative pose factors, which result into less flexible trajectory estimates.

The CVIP method is the one which achieves the best performances on the terrestrial sequences, and generally allows significant gains in accuracy as well as a manifest tightening of the error distribution. However, its results on drone sequences are very contrastive. As in the EuRoC scenarios, the method shows difficulties in absorbing the estimation errors conveyed with the received packets, hence the need for additional regularization mechanisms during packet integration. Another observation from the terrestrial sequences is the trajectory estimation suffers from the loss of the absolute gravity information, which is entirely projected onto the 6DoF relative pose factors.

PVIP shows reasonable performances on most of the scenarios. However, it also exhibits significant failures, as in scenario 3 for robot B, and also on the drone sequences. In terrestrial sequences, this may be explained by the fact that some local portions of the trajectories may be visu-

(a) EuRoC scenarios

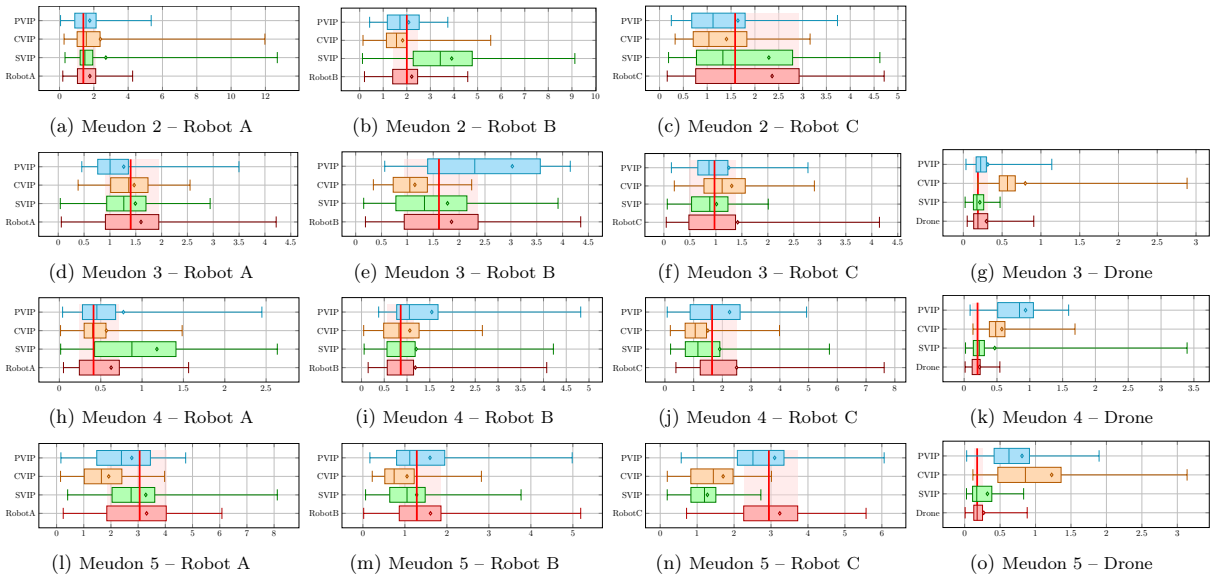


(a) MH12 – Robot 1 (b) MH12 – Robot 2 (c) MH13 – Robot 1

(d) MH13 – Robot 3 (e) MH45 – Robot 4 (f) MH45 – Robot 5

(g) MH123 – Robot 1 (h) MH123 – Robot 2 (i) MH123 – Robot 3

(b) AirMuseum scenarios



(a) Meudon 2 – Robot A (b) Meudon 2 – Robot B (c) Meudon 2 – Robot C

(d) Meudon 3 – Robot A (e) Meudon 3 – Robot B (f) Meudon 3 – Robot C (g) Meudon 3 – Drone

(h) Meudon 4 – Robot A (i) Meudon 4 – Robot B (j) Meudon 4 – Robot C (k) Meudon 4 – Drone

(l) Meudon 5 – Robot A (m) Meudon 5 – Robot B (n) Meudon 5 – Robot C (o) Meudon 5 – Drone

Figure 17: Distribution of the translation errors [m] on the estimated trajectories

ally under-constrained, and that the inertial biases may not always be observable when re-initializing the inertial states on the reconstructed received packets. Results on the drone trajectory once again highlights the difficulty to properly absorb the estimation errors associated to the trajectories which are received and integrated within the map, without compromising the estimation accuracy on its own trajectory in the case of such disparities. PVIP has difficulties in areas with low visual quality.

11. Conclusion and perspectives

In this article, we aimed at providing data exchange solutions for collaborative visual-inertial SLAM which enhances the autonomy of the robots in fusing the data received from the other agents while meeting the requirements for the informational, communication and resource constraints. We proposed CVIP, PVIP and SVIP, respectively based on the exchange of condensed, pruned and rigid visual-inertial packets, and designed a common multi-robot framework based on a decentralized and weakly dis-

tributed task allocation scheme, a dual communication policy based on a regular and a regularization regime, and a matching and merging strategy which builds on the exchanged packets for the spotting of inter-robot correspondences and multi-robot inference.

The proposed methods were evaluated on the multi-robot scenarios built from the EuRoC dataset and provided by the custom AirMuseum dataset. We showed that they allow the computation and the integration of consistent packets in real-time with a low induced bandwidth consumption in a decentralized architecture. The evaluation on the EuRoC scenarios showed that CVIP and PVIP allowed to enhance the estimation accuracy on the robot trajectories in most of the scenarios, while the SVIP method showed some limitations. The evaluation on the AirMuseum scenarios showed that they were able to positively impact the estimation accuracy on most of the terrestrial sequences, but also highlighted some limitations which draw perspectives for improvement. CVIP allows to share light packets which are easily fused within the receiver's map, but their computation is complex and conservative. PVIP provides a faster and more consistent way to broadcast the visual-inertial information which allows a more accurate two-level multi-robot inference, even though it requires more bandwidth. Finally, SVIP communicates exhaustive visual-structural information which helps to spot many inter-robot correspondences.

A perspective would be to make the integration of received packets more robust not to compromise the estimation accuracy on the other trajectories. This could be achieved prior to the multi-robot inference by adding some regularization mechanisms, for instance through a more extensive and systematic use of robust cost functions not to compromise the receiver's trajectory estimate to easily. Spotted inter-robot correspondences could also undergo a preliminary cycle analysis, as advocated by [68] and done in DOOR-SLAM [27]. Finally, the tuning of the parameters could be made automatic, especially regarding the target cover of PVIP, which may adapt to the local topology of the summarized map.

12. Acknowledgements

This work was supported by the *Direction Générale de l'Armement* (DGA).

References

[1] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, R. Siegwart, The EuRoC micro aerial vehicle datasets, The International Journal of Robotics Research 35 (10) (2016) 1157–1163.

[2] R. Dubois, A. Eudes, V. Frémont, AirMuseum: a heterogeneous multi-robot dataset for stereo-visual and inertial Simultaneous Localization And Mapping, in: 2020 IEEE International Conference on Multisensor Fusion and Integration (MFI), 2020.

[3] J. P. Queralta, J. Taipalmaa, B. C. Pullinen, V. K. Sarker, T. N. Gia, H. Tenhunen, M. Gabbouj, J. Raitoharju, T. Westerlund, Collaborative multi-robot search and rescue: Planning, coordination, perception, and active vision, IEEE Access 8 (2020) 191617–191643.

[4] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, J. J. Leonard, Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age, IEEE Transactions on robotics 32 (6) (2016) 1309–1332.

[5] R. Dubois, A. Eudes, V. Frémont, On data sharing strategy for decentralized collaborative visual-inertial simultaneous localization and mapping, in: 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 2019, pp. 2123–2130.

[6] J.-E. Deschaud, IMLS-SLAM: scan-to-model matching based on 3d data, in: 2018 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2018, pp. 2480–2485.

[7] R. Mur-Artal, J. M. M. Montiel, J. D. Tardos, ORB-SLAM: a versatile and accurate monocular SLAM system, IEEE transactions on robotics 31 (5) (2015) 1147–1163.

[8] S. Lowry, N. Sünderhauf, P. Newman, J. J. Leonard, D. Cox, P. Corke, M. J. Milford, Visual place recognition: A survey, IEEE Transactions on Robotics 32 (1) (2015) 1–19.

[9] A. I. Mourikis, S. I. Roumeliotis, A multi-state constraint Kalman filter for vision-aided inertial navigation, in: Proceedings 2007 IEEE International Conference on Robotics and Automation, IEEE, 2007, pp. 3565–3572.

[10] T. Qin, P. Li, S. Shen, Vins-mono: A robust and versatile monocular visual-inertial state estimator, IEEE Transactions on Robotics 34 (4) (2018) 1004–1020.

[11] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, P. Furgale, Keyframe-based visual–inertial odometry using nonlinear optimization, The International Journal of Robotics Research 34 (3) (2015) 314–334.

[12] S. Saeedi, M. Trentini, M. Seto, H. Li, Multiple-robot simultaneous localization and mapping: A review, Journal of Field Robotics 33 (1) (2016) 3–46.

[13] D. Zou, P. Tan, CoSLAM: Collaborative visual slam in dynamic environments, IEEE transactions on pattern analysis and machine intelligence 35 (2) (2012) 354–366.

[14] L. Riazuelo, J. Civera, J. M. Montiel, C2tam: A cloud framework for cooperative tracking and mapping, Robotics and Autonomous Systems 62 (4) (2014) 401–413.

[15] J. G. Morrison, D. Gálvez-López, G. Sibley, MoarSLAM: Multiple operator augmented rslam, in: Distributed autonomous robotic systems, Springer, 2016, pp. 119–132.

[16] F. Li, S. Yang, X. Yi, X. Yang, CORB-SLAM: a collaborative visual slam system for multiple robots, in: International Conference on Collaborative Computing: Networking, Applications and Worksharing, Springer, 2017, pp. 480–490.

[17] C. Forster, S. Lynen, L. Kneip, D. Scaramuzza, Collaborative monocular slam with multiple micro aerial vehicles, in: 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, 2013, pp. 3962–3970.

[18] P. Schmuck, M. Chli, CCM-SLAM: Robust and efficient centralized collaborative monocular simultaneous localization and mapping for robotic teams, Journal of Field Robotics 36 (4) (2019) 763–781.

[19] M. Karrer, P. Schmuck, M. Chli, CVI-SLAM: Collaborative Visual-Inertial SLAM, IEEE Robotics and Automation Letters 3 (4) (2018) 2762–2769.

[20] T. Qin, J. Pan, S. Cao, S. Shen, A general optimization-based framework for local odometry estimation with multiple sensors, arXiv preprint arXiv:1901.03638.

[21] I. Deutsch, M. Liu, R. Siegwart, A framework for multi-robot pose graph SLAM, in: 2016 IEEE International Conference on Real-time Computing and Robotics (RCAR), IEEE, 2016, pp. 567–572.

[22] A. Cunningham, V. Indelman, F. Dellaert, DDF-SAM 2.0: Consistent distributed smoothing and mapping, in: 2013 IEEE international conference on robotics and automation, IEEE, 2013, pp. 5220–5227.

[23] L. Paull, G. Huang, M. Seto, J. J. Leonard, Communication-constrained multi-auv cooperative SLAM, in: 2015 IEEE international conference on robotics and automation (ICRA), IEEE, 2015, pp. 509–516.

[24] M. J. Schuster, K. Schmid, C. Brand, M. Beetz, Distributed stereo vision-based 6d localization and mapping for multi-robot teams, Journal of Field Robotics 36 (2) (2019) 305–332.

[25] K. Sartipi, R. C. DuToit, C. B. Cobar, S. I. Roumeliotis, Decentralized visual-inertial localization and mapping on mobile devices for augmented reality., in: IROS, 2019, pp. 2145–2152.

[26] T. Cieslewski, S. Choudhary, D. Scaramuzza, Data-efficient decentralized visual SLAM, in: 2018 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2018, pp. 2466–2473.

[27] P.-Y. Lajoie, B. Ramtoula, Y. Chang, L. Carlone, G. Beltrame, DOOR-SLAM: Distributed, online, and outlier resilient SLAM for robotic teams, IEEE Robotics and Automation Letters 5 (2) (2020) 1656–1663.

[28] S. Choudhary, L. Carlone, C. Nieto, J. Rogers, H. I. Christensen, F. Dellaert, Distributed trajectory estimation with privacy and communication constraints: a two-stage distributed gauss-seidel approach, in: 2016 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2016, pp. 5261–5268.

[29] P. Koch, S. Lacroix, Managing environment models in multi-robot teams, in: 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 2016, pp. 5722–5728.

[30] A. Quraishi, T. Cieslewski, S. Lynen, R. Siegwart, Robustness to connectivity loss for collaborative mapping, in: 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 2016, pp. 4580–4585.

[31] D. Van Opdenbosch, E. Steinbach, Collaborative visual SLAM using compressed feature exchange, IEEE Robotics and Automation Letters 4 (1) (2018) 57–64.

[32] M. T. Lazaro, L. M. Paz, P. Pinies, J. A. Castellanos, G. Grisetti, Multi-robot SLAM using condensed measurements, in: 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, 2013, pp. 1069–1076.

[33] Y. Tian, K. Khosoussi, M. Giamou, J. P. How, J. Kelly, Near-optimal budgeted data exchange for distributed loop closure detection, arXiv preprint arXiv:1806.00188.

[34] J. Wang, E. Olson, AprilTag 2: Efficient and robust fiducial detection, in: 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 2016, pp. 4193–4198.

[35] R. Madhavan, K. Fregene, L. E. Parker, Distributed cooperative outdoor multirobot localization and mapping, Autonomous Robots 17 (1) (2004) 23–39.

[36] R. Käslin, P. Fankhauser, E. Stumm, Z. Taylor, E. Mueggler, J. Delmerico, D. Scaramuzza, R. Siegwart, M. Hutter, Collaborative localization of aerial and ground robots through elevation maps, in: 2016 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR), IEEE, 2016, pp. 284–290.

[37] M. Giamou, K. Khosoussi, J. P. How, Talk resource-efficiently to me: Optimal communication planning for distributed slam front-ends, in: Proceedings of the IEEE international conference on robotics and automation, 2018.

[38] A. Howard, Multi-robot simultaneous localization and mapping using particle filters, The International Journal of Robotics Research 25 (12) (2006) 1243–1256.

[39] B. Kim, M. Kaess, L. Fletcher, J. Leonard, A. Bachrach, N. Roy, S. Teller, Multiple relative pose graphs for robust cooperative mapping, in: 2010 IEEE International Conference on Robotics and Automation, IEEE, 2010, pp. 3185–3192.

[40] V. Indelman, E. Nelson, J. Dong, N. Michael, F. Dellaert, Incremental distributed inference from arbitrary poses and unknown data association: Using collaborating robots to establish a common reference, IEEE Control Systems Magazine 36 (2) (2016) 41–74.

[41] J. Solà Ortega, J. Deray, D. Atchuthan, A micro lie theory for state estimation in robotics.

[42] S. Lynen, T. Sattler, M. Bosse, J. A. Hesch, M. Pollefeys, R. Siegwart, Get out of my lab: Large-scale, real-time visual-inertial localization., in: Robotics: Science and Systems, Vol. 1, 2015.

[43] T. Schneider, M. Dymczyk, M. Fehr, K. Egger, S. Lynen, I. Gilitschenski, R. Siegwart, Maplab: An open framework for research in visual-inertial mapping and localization, IEEE Robotics and Automation Letters 3 (3) (2018) 1418–1425.

[44] G. Grisetti, R. Kümmerle, C. Stachniss, W. Burgard, A tutorial on graph-based SLAM, IEEE Intelligent Transportation Systems Magazine 2 (4) (2010) 31–43.

[45] C. Forster, L. Carlone, F. Dellaert, D. Scaramuzza, On-manifold preintegration for real-time visual–inertial odometry, IEEE Transactions on Robotics 33 (1) (2016) 1–21.

[46] C. Chow, C. Liu, Approximating discrete probability distributions with dependence trees, IEEE transactions on Information Theory 14 (3) (1968) 462–467.

[47] N. Carlevaris-Bianco, M. Kaess, R. M. Eustice, Generic node removal for factor-graph SLAM, IEEE Transactions on Robotics 30 (6) (2014) 1371–1385.

[48] M. Mazuran, W. Burgard, G. D. Tipaldi, Nonlinear factor recovery for long-term SLAM, The International Journal of Robotics Research 35 (1-3) (2016) 50–72.

[49] K. Eckenhoff, L. Paull, G. Huang, Decoupled, consistent node removal and edge sparsification for graph-based SLAM, in: 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 2016, pp. 3275–3282.

[50] J. Vallvé, J. Solà, J. Andrade-Cetto, Factor descent optimization for sparsification in graph SLAM, in: 2017 European Conference on Mobile Robots (ECMR), IEEE, 2017, pp. 1–6.

[51] H. Liu, M. Chen, G. Zhang, H. Bao, Y. Bao, Ice-ba: Incremental, consistent and efficient bundle adjustment for visual-inertial SLAM, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 1974–1982.

[52] L. Paull, G. Huang, J. J. Leonard, A unified resource-constrained framework for graph SLAM, in: 2016 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2016, pp. 1346–1353.

[53] L. Vandenberghe, S. Boyd, S.-P. Wu, Determinant maximization with linear matrix inequality constraints, SIAM journal on matrix analysis and applications 19 (2) (1998) 499–533.

[54] C. Lanczos, An iteration method for the solution of the eigenvalue problem of linear differential and integral operators, United States Governm. Press Office Los Angeles, CA, 1950.

[55] H. Kretzschmar, C. Stachniss, Information-theoretic compression of pose graphs for laser-based SLAM, The International Journal of Robotics Research 31 (11) (2012) 1219–1230.

[56] Y. Wang, R. Xiong, Q. Li, S. Huang, Kullback-leibler divergence based graph pruning in robotic feature mapping, in: 2013 European Conference on Mobile Robots, IEEE, 2013, pp. 32–37.

[57] K. Khosoussi, S. Huang, G. Dissanayake, Novel insights into the impact of graph structure on SLAM, in: 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, 2014, pp. 2707–2714.

[58] S. Choudhary, V. Indelman, H. I. Christensen, F. Dellaert, Information-based reduced landmark SLAM, in: 2015 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2015, pp. 4620–4627.

[59] Y. Zhao, P. A. Vela, Good feature selection for least squares pose optimization in VO/VSLAM, in: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 2018, pp. 1183–1189.

[60] M. Dymczyk, S. Lynen, M. Bosse, R. Siegwart, Keep it brief: Scalable creation of compressed localization maps, in: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 2015, pp. 2536–2542.

[61] F. A. Cheein, G. Scaglia, F. Di Sciasio, R. Carelli, Feature selection criteria for real time EKF-SLAM algorithm, International Journal of Advanced Robotic Systems 6 (3) (2009) 21.

[62] B. Mu, A.-a. Agha-mohammadi, L. Paull, M. Graham, J. How, J. Leonard, Two-stage focused inference for resource-

constrained collision-free navigation.
[63] M. Dymczyk, T. Schneider, I. Gilitschenski, R. Siegwart, E. Stumm, Erasing bad memories: Agent-side summarization for long-term mapping, in: 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 2016, pp. 4572–4579.
[64] M. Dymczyk, S. Lynen, T. Cieslewski, M. Bosse, R. Siegwart, P. Furgale, The gist of maps-summarizing experience for lifelong localization, in: 2015 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2015, pp. 2767–2773.
[65] S. Cao, N. Snavely, Minimal scene descriptions from structure from motion models, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 461–468.
[66] R. Dubois, A. Eudes, J. Moras, V. Frémont, Dense decentralized multi-robot SLAM based on locally consistent TSDF submaps, in: 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2020.
[67] S. Baker, I. Matthews, Lucas-kanade 20 years on: A unifying framework, International journal of computer vision 56 (3) (2004) 221–255.
[68] C. Estrada, J. Neira, J. D. Tardós, Finding good cycle constraints for large scale multi-robot SLAM, in: 2009 IEEE International Conference on Robotics and Automation, IEEE, 2009, pp. 395–402.