2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
September 27 - October 1, 2021. Prague, Czech Republic

Distributed Visual-Inertial Cooperative Localization

Pengxiang Zhu*, Patrick Geneva*, Wei Ren, and Guoquan Huang

***Abstract*— In this paper we present a consistent and distributed state estimator for multi-robot cooperative localization (CL) which efficiently fuses environmental features and loop-closure constraints across time *and* robots. In particular, we leverage covariance intersection (CI) to allow each robot to only estimate its own state and autocovariance and compensate for the unknown correlations between robots. Two novel multi-robot methods for utilizing common environmental SLAM features are introduced and evaluated in terms of accuracy and efficiency. Moreover, we adapt CI to enable drift-free estimation through the use of loop-closure measurement constraints to other robots' historical poses without a significant increase in computational cost. The proposed distributed CL estimator is validated against its non-realtime centralized counterpart extensively in both simulations and real-world experiments.**

I. Introduction

Camera and inertial measurement unit (IMU) pairs have been at the forefront of multi-robot (or mobile device) applications due to their complementary nature, low cost and small size. Accurate and efficient cooperative localization (CL) that enables multi-user augmented reality (AR) experiences, multi-device cooperative mapping, and multi-vehicle formation control, is a key barrier to overcome due to challenges of communication, distributed computation, and complexity of multi-robot asynchronous measurement constraints.

In this paper, building upon our recent work [1], we propose a fully distributed multi-robot visual-inertial CL estimator by delicately exploiting information contained in both environmental SLAM landmarks and loop-closures across robots and time. Specifically, we extend our prior CI-based cooperative visual-inertial odometry (VIO) system [1] to include both SLAM features and incorporate loop-closure constraints to historical states of other robots, thus limiting the current robot's localization drift essentially using *asynchronous* common views seen from other robots' historical poses. As a result, the proposed distributed CL estimator does *not* require *simultaneous* viewing of the same location due to leveraging of historical common features (e.g., a robot can gain information if another robot had previously explored the same location), while significantly improving the localization performance thanks to such common multi-robot measurement information. In summary, the main contributions of this work are the following:

*These authors contributed equally to this work.

Zhu and Ren are with the Department of Electrical and Computer Engineering, University of California, Riverside, CA 92521, USA. Email: pzhu008@ucr.edu, ren@ee.ucr.edu. Geneva and Huang are with the Robot Perception and Navigation Group (RPNG), University of Delaware, Newark, DE 19716, USA. Email: {pgeneva, ghuang}@udel.edu

This work was partially supported by the University of Delaware (UD) College of Engineering, the NSF (IIS-1924897, CMMI-2027139), and the ARL (W911NF-19-2-0226).

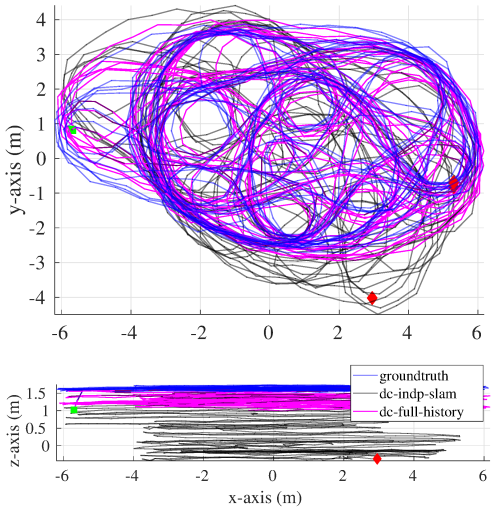

Fig. 1: Trajectory of groundtruth, independent, and distributed historical trajectory for Robot 0 in the Vicon room dataset. It can be seen that the use of common historical features limit drift in the z-axis along with improvements in x-y accuracy. Please refer to the color figure.

- We develop a fully distributed CI-based visual-inertial CL estimation algorithm, which allows for accurate, efficient and consistent estimation of all robot states.
- We propose two different SLAM feature measurement models that allow for cooperative estimation of common long-lived environmental features, and validate their relative accuracy and computational complexity through a series of simulations.
- We introduce a computationally efficient method for long-term loop-closure to reduce localization drift, which enables multi-robot constraints between historical poses and features, allowing for robots to gain additional constraints even in the case when other robots are not actively in the same location.
- We thoroughly validate the proposed approach in Monte Carlo simulations and real world experiments by comparing to centralized CL algorithms.

II. Related Work

Significant research efforts have recently been devoted to visual-inertial navigation system (VINS) [2], while primarily focusing on improving *single-robot* VINS accuracy, efficiency, and robustness [3]–[5]. The extension to the *multi-robot* case is not sufficiently explored as a naive approach would be prohibitively costly and non-realtime. For example, one could communicate all measurements generated from itself to each other (or fusion center), where all measurements could be optimally fused and all states can be refined jointly. While this does allow for accurate estimation, both the requirement for constant communication and the

joint estimation of robot states requires cubic computational complexity in terms of the number of robots. As such, a multi-robot *distributed* estimator is needed to address these shortcomings by relaxing communication requirements and distributing the computation cost across all robots.

Efficient 2D CL has focused on the fusion of relative measurements between robots (e.g., relative robot-to-robot bearing or distance range measurements). Roumeliotis et al. [6] proposed a decentralized algorithm that achieves performance equivalent to the centralized formulation, but required communication between all robots and increases in computational cost due to its centralized nature as the number of robots grow. Other works such as [7] have investigated the approximation of the robot-to-robot cross-covariances that are not involved in a relative measurement update to reduce the computational cost, and while it performs close to its centralized, it is unable to guarantee consistency and thus can easily diverge. More recently, Jung et al. [8] extended this work to the 3D case, but inherits the same underlying issues and requires maintaining of the approximated robot-to-robot cross-covariances. There exist other works aiming at estimating the relative poses between robots using relative measurements [9], [10]. Alternative approaches have leveraged CI [11], [12] to guarantee consistency and only requires that each robot maintains its own state and auto-covariance (the correlations between robots are ignored). By contrast, in this work we specifically take advantage of the CI formulation for 3D multi-robot state estimation, enabling a consistent distributed algorithm which fuses inertial and visual sparse environmental feature information.

As compared to CL with relative distance, bearing, or poses between robots [6]–[15], common sparse environmental features are used in [16]–[19], which is appealing as getting relative robot information can be difficult with visual-inertial sensors in practice and requires both the detection and tracking of other robots. For example, Melnyk et al. [18] introduced CL-MSCKF using common environmental feature constraints within a centralized formulation that jointly estimated all robot states. They required that robots communicate all sensor data to a common fusion center and demonstrated its use for the two robot case in simulation. Karrer et al. [17] developed a graph-based centralized server which handled non-realtime computationally expensive loop closure detection and optimization of all robot maps to find the joint global optimal. In this paper, we instead focus on the computationally efficient *distributed* localization problem where each robot only estimates its *own* state and tries to leverage information from other robots without a centralized server or joint optimization.

As closest to our work, Sartipi et al. [19] introduced a distributed method for multi-user AR experiences through the use of multi-map feature constraints. Common features were detected in environmental maps received from other users and the transmitted feature position estimates were used to constrain the user's state directly. Instead of inflating measurement noise to compensate for the unknown correlations between the current user and the other user's map, we leverage CI that theoretically guarantee consistency to handle the unknown correlations. Also, instead of requiring that all common features must match to sparse features in the other user's map, we leverage the other user's common feature measurements directly allowing for update with additional measurements.

III. Cooperative Visual-Inertial System

In this section, we briefly describe the cooperative visual-inertial system that serves the basis for the proposed distributed CI-based estimator. The state vector for the i'th robot contains its current IMU navigation state $\mathbf{x}_{I_i}$, sliding window of cloned IMU poses $\mathbf{x}_{C_i}$, spatial-temporal calibration parameters $\mathbf{x}_{W_i}$, along with a small temporal map (i.e., SLAM features) $\mathbf{x}_{M_i}$ (see [1], [20]).

$$\mathbf{x}_{i,k} = \begin{bmatrix} \mathbf{x}_{I_i}^\top & \mathbf{x}_{W_i}^\top & \mathbf{x}_{C_i}^\top & \mathbf{x}_{M_i}^\top \end{bmatrix}^\top \tag{1}$$

$$\mathbf{x}_{I_i} = \begin{bmatrix} {}^{I_{i,k}}_{G}\bar{q}^\top & {}^G\mathbf{p}_{I_{i,k}}^\top & {}^G\mathbf{v}_{I_{i,k}}^\top & \mathbf{b}_{\omega_{i,k}}^\top & \mathbf{b}_{a_{i,k}}^\top \end{bmatrix}^\top \tag{2}$$

$$\mathbf{x}_{W_i} = \begin{bmatrix} {}^{C_i}t_{I_i} & {}^{C_i}_{I_i}\bar{q}^\top & {}^{C_i}\mathbf{p}_{I_i}^\top & \boldsymbol{\zeta}_i^\top \end{bmatrix}^\top \tag{3}$$

$$\mathbf{x}_{C_i} = \begin{bmatrix} {}^{I_{i,k-1}}_{G}\bar{q}^\top & {}^G\mathbf{p}_{I_{i,k-1}}^\top & \cdots & {}^{I_{i,k-c}}_{G}\bar{q}^\top & {}^G\mathbf{p}_{I_{i,k-c}}^\top \end{bmatrix}^\top \tag{4}$$

$$\mathbf{x}_{M_i} = \begin{bmatrix} {}^G\mathbf{p}_{f1}^\top & \cdots & {}^G\mathbf{p}_{fm}^\top \end{bmatrix}^\top \tag{5}$$

where ${}^{I_{i,k}}_{G}\bar{q}$ is the unit quaternion parameterizing the rotation $\mathbf{C}({}^{I_{i,k}}_{G}\bar{q}) = {}^{I_{i,k}}_{G}\mathbf{R}$ from the global frame of reference $\{G\}$ to the IMU local frame $\{I_k\}$ at time k for the i'th robot [21], $\mathbf{b}_{\omega_{i,k}}$ and $\mathbf{b}_{a_{i,k}}$ are the gyroscope and accelerometer biases, and ${}^G\mathbf{v}_{I_{i,k}}$ and ${}^G\mathbf{p}_{I_{i,k}}$ are the velocity and position of the IMU expressed in the global frame, respectively. The clone state $\mathbf{x}_C$ contains c historical IMU poses in a sliding window, while the temporal map state $\mathbf{x}_M$ has m features. Each robot additionally calibrates its camera intrinsics $\boldsymbol{\zeta}_i$, camera-IMU extrinsics, and camera-IMU temporal offset ${}^{C_i}t_{I_i}$ [20]. Finally, given a group of n robots, we have the following combined state and covariance matrix decomposition:

$$\mathbf{x}_k = \begin{bmatrix} \mathbf{x}_{1,k}^\top & \cdots & \mathbf{x}_{n,k}^\top \end{bmatrix}^\top \tag{6}$$

$$\mathbf{P}_k = \begin{bmatrix} \mathbf{P}_{11_k} & \cdots & \mathbf{P}_{N1_k} \\ \vdots & \ddots & \vdots \\ \mathbf{P}_{1N_k} & \cdots & \mathbf{P}_{NN_k} \end{bmatrix} \tag{7}$$

Here we note that in the centralized formulation this is the state that we jointly estimate along with the cross-covariance terms, while in the distributed case each robot only estimates a sub-set of the total state and correlations between robots are dropped (e.g., robot i only tracks $\mathbf{x}_{i,k}$ and $\mathbf{P}_{ii_k}$).

A. Inertial Propagation

The inertial state of the i'th robot $\mathbf{x}_{I_i}$ is propagated forward using its own IMU measurements of linear accelerations ($\mathbf{a}_{m_i}$) and angular velocities ($\boldsymbol{\omega}_{m_i}$) based on the following generic nonlinear IMU kinematics [22]:

$$\mathbf{x}_{i,k+1} = \mathbf{f}(\mathbf{x}_{i,k}, \mathbf{a}_{m_k} - \mathbf{n}_{a_k}, \boldsymbol{\omega}_{m_k} - \mathbf{n}_{\omega_k}) \tag{8}$$

where $\mathbf{n}_a$ and $\mathbf{n}_\omega$ are the zero-mean white Gaussian noise of the IMU measurements. We linearize this nonlinear model at the current estimate for all robots, and can then propagate the state covariance matrix forward in time:

$$\mathbf{P}_{k|k-1} = \boldsymbol{\Phi}_{k-1}\mathbf{P}_{k-1|k-1}\boldsymbol{\Phi}_{k-1}^\top + \mathbf{Q}_{k-1} \tag{9}$$

$$\boldsymbol{\Phi}_{k-1} = \mathbf{Diag}\left(\boldsymbol{\Phi}_{1,k-1}, \ldots, \boldsymbol{\Phi}_{N,k-1}\right) \tag{10}$$

$$\mathbf{Q}_{k-1} = \mathbf{Diag}\left(\mathbf{Q}_{1,k-1}, \ldots, \mathbf{Q}_{N,k-1}\right) \tag{11}$$

where $\mathbf{\Phi}_{i,k}$ and $\mathbf{Q}_{i,k}$ are respectively the system Jacobian and discrete noise covariance for the i'th robot [3], and $\mathbf{Diag}(\cdots)$ creates a block diagonal matrix from the specified values. In the distributed case, all states can be propagated independently since cross-covariance are not tracked.

B. Camera Measurement Update

A corner feature at time-step k can be be written as the distortion of a perspective projection of a 3D point ${}^{C_{i,k}}\mathbf{p}_f$, expressed in the i'th robot's camera frame:

$$\mathbf{z}_k = \mathbf{h}_{dist}(\mathbf{z}_{k,n}, \boldsymbol{\zeta}_i) + \mathbf{n}_{f_k} \tag{12}$$

$$\mathbf{z}_{k,n} = \frac{1}{{}^{C_{i,k}}z_f}\begin{bmatrix} {}^{C_{i,k}}x_f \\ {}^{C_{i,k}}y_f \end{bmatrix} \tag{13}$$

$${}^{C_{i,k}}\mathbf{p}_f = {}^{C_i}_{I_i}\mathbf{R}\,{}^{I_{i,k}}_{G}\mathbf{R}\left({}^{G}\mathbf{p}_f - {}^{G}\mathbf{p}_{I_{i,k}}\right) + {}^{C_i}\mathbf{p}_{I_i} \tag{14}$$

where $\mathbf{n}_{f_k}$ is the zero-mean white Gaussian measurement noise with covariance $\mathbf{R}_k$, and $\mathbf{h}_{dist}(\cdot)$ is the camera distortion function which maps a normalized bearing $\mathbf{z}_{k,n}$ to the raw distorted image plane. The linearization of this measurement model (12) yields the following:

$$\mathbf{r}_{f_k} = \mathbf{H}_k\tilde{\mathbf{x}}_k + \mathbf{n}_{f_k} = \mathbf{H}_{x_{i,k}}\tilde{\mathbf{x}}_{I_i} + \mathbf{H}_{f_k}{}^{G}\tilde{\mathbf{p}}_f + \mathbf{n}_{f_k} \tag{15}$$

Once the measurement residual and Jacobian are computed the state and error covariance can be updated using the standard EKF update equations [23].

IV. Distributed Visual-Inertial CL

As it is known that the standard EKF in the worst case has cubic computation complexity due to its covariance update, a naive implementation of the multi-robot visual-inertial CL can become prohibitively expensive as the number of robots grow in size. Note also that due to communication constraints, the robots might not be able to communicate with all the other robots or a common fusion center. To address these issues, the key idea of our CL approach is to leverage CI [24] to reduce the estimation cost, by only updating the state and error covariance of the current robot (i.e., robot i only updates $\mathbf{x}_{i,k}$ and $\mathbf{P}_{ii_k}$) while ensuring consistency.

In particular, each robot independently propagates its own state and updates with measurements that are only a function of its own state. When updating with measurements of features observed from multiple robots, CI is employed to consistently handle the unknown and untracked cross-covariance terms between the involved robots. This means that robots need to communicate their state and covariance, along with visual feature information to the other robots. Each robot tracks a set of visual features using KLT optical flow [25], and communicates its latest tracks and extracted ORB descriptors [26] to the other robots in communication range. A robot then performs descriptor-based feature matching and loop-closure detection to find correspondences between its most recent features and other robots' feature tracks. After tracking and matching, feature tracks are categorized as follows:

(A) VIO features which have only been tracked for a short period of time.
(B) Temporal SLAM features which have been tracked beyond the current sliding window.
(C) Common VIO features which have been matched to features in another robot and tracked for only a short period of time.
(D) Common SLAM features which have been matched to features in another robot. Note that this feature might be either a VIO or SLAM feature in the other robot.

In the following, we present in detail how we update our state with these different feature variants. Note that for the centralized case independent features update the full state and covariance since cross-covariances are tracked, while in the distributed case only the i'th robot state and covariance is updated thus allowing for computational savings.

A. Independent VIO Feature: MSCKF Update

For VIO features that have lost active track in the current window, we perform MSCKF update [3]. In particular, we first triangulate these features for computing the feature Jacobians $\mathbf{H}_{f_k}$, and then project $\mathbf{r}_{f_k}$ [see (15)] onto the left nullspace of $\mathbf{H}_{f_k}$ (i.e., $\mathbf{Q}_2^\top\mathbf{H}_{f_k} = \mathbf{0}$) to yield the measurement noise independent of state:

$$\mathbf{Q}_2^\top\mathbf{r}_{f_k} = \mathbf{Q}_2^\top\mathbf{H}_{x_{i,k}}\tilde{\mathbf{x}}_{i,k} + \mathbf{Q}_2^\top\mathbf{H}_{f_k}{}^{G}\tilde{\mathbf{p}}_f + \mathbf{Q}_2^\top\mathbf{n}_{f_k} \tag{16}$$

$$\Rightarrow\ \mathbf{r}'_{f_k} = \mathbf{H}'_x\tilde{\mathbf{x}}_k + \mathbf{n}'_{f_k} \tag{17}$$

where $\mathbf{H}_{x_{i,k}}$ is the stacked measurement Jacobians with respect to the navigation states in the current robot's window.

B. Independent SLAM Feature: FEJ-EKF Update

SLAM features which a robot is able to reliably track longer then its sliding window in length, will be initialized into the SLAM map state vector $\mathbf{x}_{M_i}$. These features are directly updated using the linearized system (15) and will remain in the state until they have lost tracking. To improve consistency, we employ First Estimate Jacobians (FEJ) [27], [28] ensuring Jacobians are evaluated at the same linearization points to prevent spurious information gain.

C. Common VIO Feature: CI-EKF Update

Consider we find a feature which has been seen from multiple robots and want to use this information to update the state. In the centralized case, we would directly update our state with all available measurements (15) through the standard EKF since we track the cross-covariance (e.g., $\mathbf{P}_{iN_k}$). In the distributed case, a robot only tracks its own state and autocovariance to ensure computational efficiency and scalability with respect to the robot team size. This presents two key challenges: (i) how to efficiently and consistently fuse multiple robots' autocovariances, and (ii) how to find the data association between different features, which motivates us to leverage CI to fuse estimates and covariances transmitted from other robots.

1) CI-EKF Update: Consider the i'th robot has a measurement which is a function of L other robot states. The linearized measurement model can be computed as:

$$\mathbf{r}_{f_k} = \mathbf{H}_{x_{i,k}}\tilde{\mathbf{x}}_{i,k} + \mathbf{H}_{x_{1..L,k}}\tilde{\mathbf{x}}_{1..L,k} + \mathbf{H}_{f_k}{}^{G}\tilde{\mathbf{p}}_f + \mathbf{n}_{f_k} \tag{18}$$

where $\mathbf{H}_{x_{i,k}}$ is the Jacobian in respect to the i'th robot state using the k'th estimates, and $\mathbf{H}_{x_{1..L,k}}$ is the stacked Jacobian with respect to all other robots the measurement is a function of. To guarantee consistency when updating

with this measurement, we adopt the CI-EKF update [24] to construct a prior covariance such that:

$$\mathbf{Diag}\left({}^{1}/_{\omega_i}\mathbf{P}_{ii_k}, {}^{1}/_{\omega_1}\mathbf{P}_{11_k}, \cdots, {}^{1}/_{\omega_L}\mathbf{P}_{LL_k}\right) \geq \mathbf{P}_k \tag{19}$$

where the left side is the CI covariance with zero off-diagonal elements and the right hand side is the unknown true covariance of the state with cross-covariances [see (7)]. The weights $\omega_l > 0$ and $\sum_l \omega_l = 1$, for $l \in \{i, 1..L\}$, can be found optimally [24]. Substituting (19) into the standard EKF equations and only selecting the portion that updates the current robot's state (say robot i) yields:

$$\delta\mathbf{x}_{i,k} = \frac{1}{\omega_i}\mathbf{P}_{ii,k|k-1}\mathbf{H}_{x_{i,k}}^{\top}\mathbf{S}_k^{-1}\mathbf{r}'_{f_k} \tag{20}$$

$$\mathbf{P}_{ii,k|k} = \frac{1}{\omega_i}\mathbf{P}_{ii,k|k-1} - \frac{1}{\omega_i^2}\mathbf{P}_{ii,k|k-1}\mathbf{H}_{x_{i,k}}^{\top}\mathbf{S}_k^{-1}\mathbf{H}_{x_{i,k}}\mathbf{P}_{ii,k|k-1} \tag{21}$$

$$\mathbf{S}_k = \sum_{o\in\{i,1..L\}} \frac{1}{\omega_o}\mathbf{H}_{x_{o,k}}\mathbf{P}_{oo,k|k-1}\mathbf{H}_{x_{o,k}}^{\top} + \mathbf{R}_{f_k} \tag{22}$$

where $\delta\mathbf{x}_{i,k}$ is the correction to the state estimate $\hat{\mathbf{x}}_{i,k}$.

2) Efficient Nullspace Projection: To process common features which are short in length, we leverage the similar logic as in Sec. IV-A. For example, we have multiple measurements from two different robots and wish to update our state:

$$\begin{bmatrix}\mathbf{r}_{f_{i,k}}\\ \mathbf{r}_{f_{2,k}}\end{bmatrix} = \begin{bmatrix}\mathbf{H}_{x_{i,k}} & \mathbf{0} & \mathbf{H}_{f_{i,k}}\\ \mathbf{0} & \mathbf{H}_{x_{2,k}} & \mathbf{H}_{f_{2,k}}\end{bmatrix}\begin{bmatrix}\widetilde{\mathbf{x}}_{I_i}\\ \widetilde{\mathbf{x}}_{I_2}\\ {}^{G}\widetilde{\mathbf{p}}_f\end{bmatrix} + \begin{bmatrix}\mathbf{n}_{f_{i,k}}\\ \mathbf{n}_{f_{2,k}}\end{bmatrix} \tag{23}$$

We can then project both equations onto their left range and nullspace (e.g., $\mathbf{H}_{f_{i,k}} = [\mathbf{Q}_{i,1}\ \mathbf{Q}_{i,2}][\mathbf{U}_i\ \mathbf{0}]^{\top}$):

$$\begin{bmatrix}\mathbf{r}^1_{f_{i,k}}\\ \mathbf{r}^2_{f_{i,k}}\\ \mathbf{r}^1_{f_{2,k}}\\ \mathbf{r}^2_{f_{2,k}}\end{bmatrix} = \begin{bmatrix}\mathbf{Q}_{i,1}^{\top}\mathbf{H}_{x_{i,k}} & \mathbf{0} & \mathbf{U}_i\\ \mathbf{Q}_{i,2}^{\top}\mathbf{H}_{x_{i,k}} & \mathbf{0} & \mathbf{0}\\ \mathbf{0} & \mathbf{Q}_{2,1}^{\top}\mathbf{H}_{x_{2,k}} & \mathbf{U}_2\\ \mathbf{0} & \mathbf{Q}_{2,2}^{\top}\mathbf{H}_{x_{2,k}} & \mathbf{0}\end{bmatrix}\begin{bmatrix}\widetilde{\mathbf{x}}_{I_i}\\ \widetilde{\mathbf{x}}_{I_2}\\ {}^{G}\widetilde{\mathbf{p}}_f\end{bmatrix} + \begin{bmatrix}\mathbf{n}^1_{f_{i,k}}\\ \mathbf{n}^2_{f_{i,k}}\\ \mathbf{n}^1_{f_{2,k}}\\ \mathbf{n}^2_{f_{2,k}}\end{bmatrix}$$

where we have defined that $\mathbf{r}^1_{f_{i,k}} = \mathbf{Q}_{i,1}^{\top}\mathbf{r}_{f_{i,k}}$ and $\mathbf{n}^1_{f_{i,k}} = \mathbf{Q}_{i,1}^{\top}\mathbf{n}_{f_{i,k}}$. Note that the last row is no longer dependent on the current robot's state, $\mathbf{x}_{I_i}$, and thus, this can be discarded since it will not update the state or covariance due to the lack of tracked cross-covariances. This directly reduces the number of measurements involved during update and makes the computation of $\mathbf{S}_k^{-1}$ substantially cheaper [see (22)]. We then have the following linear systems:

$$\begin{bmatrix}\mathbf{r}^1_{f_{i,k}}\\ \mathbf{r}^1_{f_{2,k}}\end{bmatrix} = \begin{bmatrix}\mathbf{Q}_{i,1}^{\top}\mathbf{H}_{x_{i,k}} & \mathbf{0} & \mathbf{U}_i\\ \mathbf{0} & \mathbf{Q}_{2,1}^{\top}\mathbf{H}_{x_{2,k}} & \mathbf{U}_2\end{bmatrix}\begin{bmatrix}\widetilde{\mathbf{x}}_{I_i}\\ \widetilde{\mathbf{x}}_{I_2}\\ {}^{G}\widetilde{\mathbf{p}}_f\end{bmatrix} + \begin{bmatrix}\mathbf{n}^1_{f_{i,k}}\\ \mathbf{n}^1_{f_{2,k}}\end{bmatrix}$$

$$\mathbf{r}^2_{f_{i,k}} = \mathbf{Q}_{i,2}^{\top}\mathbf{H}_{x_{i,k}}\widetilde{\mathbf{x}}_{I_i} + \mathbf{n}^2_{f_{i,k}} \tag{24}$$

A second nullspace projection onto the left nullspace of $\mathbf{H}_f = [\mathbf{U}_i\ \mathbf{U}_2]^{\top}$ is performed to create a linear system which is only a function of the $\mathbf{x}_{I_i}$ and $\mathbf{x}_{I_2}$ states. The CI-EKF update [see (20) and (21)] is then used to update the state $\mathbf{x}_{I_i}$. The second equation [see (24)] can update the current robot state without CI through the standard EKF equations since it is only a function of the current robot state. This update contains the same information as in the case that we performed a "large" nullspace projection using the full feature Jacobians in (23), but results in a much smaller measurement size since we can drop measurement residuals which are not a function of the i'th robot's state.

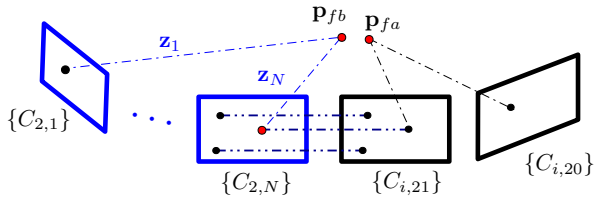

Fig. 2: Illustration of the keyframe-aided 2D-to-2D matching for data association. Assuming robot i's 21st frame $\{C_{i,21}\}$ matches to the 2nd robot's N'th frame $\{C_{2,N}\}$. We are able to find all feature correspondences between the features the robot's observer, namely $\mathbf{z}_{1..N}$.

D. Common SLAM Feature: CI-EKF Update

There are two different cases for temporal SLAM features: (i) a SLAM feature in the current robot state matches to a feature that is not a SLAM feature in another robot, and (ii) a SLAM feature matches to another robot's SLAM feature. For example as in Fig. 2, in the first case we collect the measurements from the other robot ($\mathbf{z}_{1..N}$) and directly apply (18) and update both the current robot's poses and its estimate of the SLAM feature. In the second case, we can either follow this same logic (i.e., grab the measurements from the other robot and update current robot's estimate) or we can leverage the knowledge that the 3D position of these two features should be equal. This SLAM feature constraint model is similar to the one introduced in [29] for cooperative mapping. Consider we have the following two robots:

$$\mathbf{x}_{i,k} = \begin{bmatrix}\mathbf{x}_{I_i}^{\top} & \mathbf{x}_{W_i}^{\top} & \mathbf{x}_{C_i}^{\top} & {}^{G}\mathbf{p}_{fa}^{\top}\end{bmatrix}^{\top} \tag{25}$$

$$\mathbf{x}_{2,k} = \begin{bmatrix}\mathbf{x}_{I_2}^{\top} & \mathbf{x}_{W_2}^{\top} & \mathbf{x}_{C_2}^{\top} & {}^{G}\mathbf{p}_{fb}^{\top}\end{bmatrix}^{\top} \tag{26}$$

If we have matched feature ${}^{G}\mathbf{p}_{fa}$ in the current i'th robot to the ${}^{G}\mathbf{p}_{fb}$ in the other robot, then we can construct the following feature constraint (see Fig. 2):

$${}^{G}\mathbf{p}_{fa} - {}^{G}\mathbf{p}_{fb} = \mathbf{0} \Rightarrow\ \mathbf{r}_c\left(\mathbf{x}_{i,k}, \mathbf{x}_{2,k}\right) = \mathbf{0} \tag{27}$$

which can be linearized to yield:

$$\mathbf{r}_c\left(\hat{\mathbf{x}}_{i,k}, \hat{\mathbf{x}}_{2,k}\right) + \mathbf{H}_{fa}{}^{G}\widetilde{\mathbf{p}}_{fa} + \mathbf{H}_{fb}{}^{G}\widetilde{\mathbf{p}}_{fb} \approx \mathbf{0} \tag{28}$$

$$\Rightarrow \mathbf{0} - \mathbf{r}_c\left(\hat{\mathbf{x}}_{i,k}, \hat{\mathbf{x}}_{2,k}\right) \approx \mathbf{H}_{fa}{}^{G}\widetilde{\mathbf{p}}_{fa} + \mathbf{H}_{fb}{}^{G}\widetilde{\mathbf{p}}_{fb} \tag{29}$$

This linearized system can then update the i'th robot state estimate using the CI-EKF update [see (20) and (21)]. Note that this is a very efficient update, as it is only a function of the two estimated feature positions.

E. Historical Features: CI-EKF Update

We now explain how to leverage loop-closure constraints to previous robot states. First, to find the feature correspondences between robots, as in [5], [30], each robot create DBoW2 [31] databases for all other robots. When a robot receives feature tracks and descriptors from other robots they are appended to their corresponding DBoW2 database. The current image can then be queried against the other robots' databases to see if any other robots are or have been at the current location. If a loop-closure is detected and verified using a fundamental matrix geometric check, then we assume that we have detected that another robot has been at our current location. After matching descriptors, we know the correspondences between a feature in the current robot, and that of the features in the other robot (see Fig. 2). We can then grab the history of measurements and formulate a common feature update.

TABLE I: Simulation parameters and prior standard deviations that perturbations of measurements and initial states were drawn from.

Parameter	Value	Parameter	Value
Gyro. White Noise	1.6968e-04	Gyro. Rand. Walk	1.9393e-05
Accel. White Noise	2.0000e-3	Accel. Rand. Walk	3.0000e-3
Pixel Proj. (px)	1	Robot Num.	3
IMU Freq. (hz)	400	Cam Freq. (hz)	10
AR Avg. Feats	25	AR Num. SLAM	3
ETH Avg. Feats	50	ETH Num. SLAM	5
Num. Clones	11	Feat. Rep.	GLOBAL

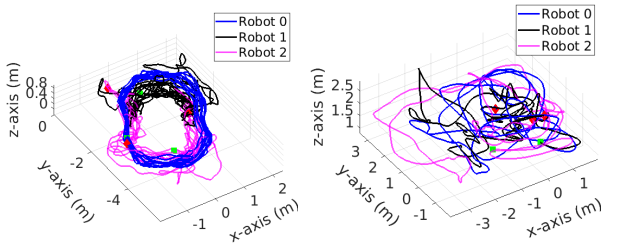

Fig. 3: Simulated trajectories, axes are in units of meters. General hand-held AR dataset (left) are 147, 93, and 100 meters long, while ETH EuRoC MAV Vicon room datasets (right) are 70, 58, and 59 meters long for each robot. Green square denotes the start and red diamond denotes the end.

To incorporate these measurements from historical states, each robot records the measurement and previous states received from the other agents.[1] Outside of the most recent sliding window, these historical states can provide loop-closure information if we are able to generate measurement constraints to them. Specifically we store the following historical states and covariances in addition to their most recent states published:

$$\mathbf{x}_i = \{\mathbf{x}_{i,0}, \cdots, \mathbf{x}_{i,k-1}\}, \ \mathbf{P}_i = \{\mathbf{P}_{ii_0}, \cdots, \mathbf{P}_{ii_{k-1}}\} \quad (30)$$

Since each one of these historical states contain a sliding window of poses and SLAM features, we only store non-overlapping sliding windows. To accelerate lookup we only store historical descriptor information at a fixed rate (normally 1Hz) since recent frames in the same sliding window contain redundant loop-closure information. More ideal heuristic could be leveraged here to increase match rates. Once loop-closure is detected, we know old historical feature correspondences which we can then use to retrieve measurements and update our current robot state. This update is identical to the CI-EKF update as in Sec. IV-C–IV-D, which only needs to involve the historical windows that contain the historical measurements, and thus is efficient since historical states are not updated.

V. Simulation Results

To validate the proposed method, we have simulated two realistic scenarios both with three robots (see Fig. 3). The first is a hand-held mobile AR dataset which has a series of users look and move around a central table, while the second is a series of trajectories from the ETH EuRoC MAV dataset [32]. We employ the OpenVINS simulator [30] to generate realistic visual-bearing and inertial measurements from these supplied trajectories. On average each robot is able to find common features on, respectively, 79.0% and 83.5% (43.7% and 62.7%) of the frames without or with loop-closure in AR datasets (ETH dataset). This clearly shows that advantage of historical loop-closure on datasets which have limited temporal view overlaps between robots. Simulation parameters used are documented in Tab. I. We fix the weight of other robots' covariance in the CI-EKF update as $\omega_o = 0.001$. While for the constraint measurement update presented in Sec. IV-D, we use the value $\omega_o = 0.005$ and a synthetic measurement noise of 2cm. Note that while these weights can be found by minimizing the trace or determinant of $\mathbf{P}_{ii,k|k}$ [24], we have empirically found that using fixed weights still ensures consistent performance. For fair and thorough comparison, we define the following variations of the centralized and proposed distributed CL estimators:

indp – No common features are found between robots and all measurements are processed as independent features which only relate to the current robot.

indp-slam – Same as *indp*, but temporal SLAM features are included in each robot to show the relative improvement.

ce-cmsckf – The centralized estimator using the common VIO features over the sliding window.

ce-cmsckf-cslam – The centralized estimator using the common VIO and SLAM features over the sliding window.

dc-cmsckf [1] – The distributed estimator using the common VIO features over the sliding window.

dc-cmsckf-cslam – The distributed estimator using the common VIO and SLAM features over the sliding window without enforcing the same feature constraint. For example, even if a common SLAM feature is a SLAM feature in another robot's state, we grab the measurements from the other robot and update as the first case in Sec. IV-D.

dc-full-window – The distributed estimator using the common VIO and SLAM features over the sliding window with enforcing the same feature constraint.

dc-full-history – The distributed estimator using both the common VIO and SLAM features over the sliding window and from historical matching.

Note that the observed independent VIO features and SLAM features are used in all these estimators. To ensure a fair comparison, the same parameters reported in Tab. I are used for all algorithms and for all robots.

TABLE II: ATE on simulated AR datasets in degrees / meters for each algorithm variation. Green denotes the best, while blue is second best.

Algorithm	Robot 0	Robot 1	Robot 2	Average
indp	1.957 / 0.072	0.811 / 0.041	0.742 / 0.039	1.170 / 0.051
indp-slam	1.396 / 0.046	0.602 / 0.029	0.557 / 0.022	0.852 / 0.032
ce-cmsckf	0.364 / 0.017	0.323 / 0.015	0.355 / 0.015	0.347 / 0.016
ce-cmsckf-cslam	**0.232** / **0.011**	**0.228** / **0.011**	**0.220** / **0.010**	**0.227** / **0.011**
dc-cmsckf	0.759 / 0.029	0.540 / 0.025	0.553 / 0.020	0.617 / 0.025
dc-cmsckf-cslam	0.643 / 0.025	0.496 / 0.022	0.478 / 0.017	0.539 / 0.022
dc-full-window	0.644 / 0.024	0.547 / 0.022	0.480 / 0.017	0.557 / 0.021
dc-full-history	**0.356** / **0.017**	**0.299** / **0.014**	**0.319** / **0.013**	**0.325** / **0.014**

A. Accuracy and Consistency Evaluation

We performed 20 Monte Carlo simulations on each dataset. The average Absolute Trajectory Error (ATE) [33] can be found in Tab. II and III. It is clear from the top two rows that the additional SLAM features improve `indp`. In the cooperative case, when using the common VIO features, both `ce-msckf` and `dc-msckf` outperform the `indp-slam`, and when including common SLAM features,

[1] In the future we plan to investigate the latency introduced due to communication constraints, but historical matching ensures that the robot will leverage *all* available information at the current time including delayed information recently communicated.

TABLE III: ATE on simulated ETH datasets in degrees / meters for each algorithm variation. Green denotes the best, while blue is second best.

Algorithm	Robot 0	Robot 1	Robot 2	Average
indp	0.569 / 0.088	0.578 / 0.092	0.560 / 0.093	0.569 / 0.091
indp-slam	0.371 / 0.070	0.406 / 0.069	0.444 / 0.075	0.407 / 0.071
ce-cmsckf	0.221 / 0.052	0.221 / 0.049	0.221 / 0.051	0.221 / 0.050
ce-cmsckf-cslam	**0.151** / **0.042**	**0.143** / **0.038**	**0.144** / **0.040**	**0.146** / **0.040**
dc-cmsckf	0.329 / 0.064	0.342 / 0.061	0.319 / 0.062	0.330 / 0.062
dc-cmsckf-cslam	0.298 / 0.054	0.325 / 0.050	0.290 / 0.052	0.304 / 0.052
dc-full-window	0.285 / 0.052	0.287 / 0.047	0.268 / 0.047	0.280 / 0.049
dc-full-history	**0.211** / **0.029**	**0.207** / **0.031**	**0.218** / **0.030**	**0.212** / **0.030**

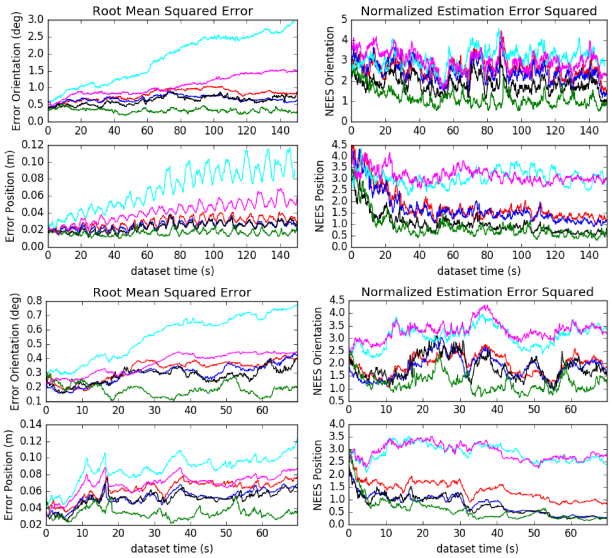

Fig. 4: Robot 0's average RMSE (left) and NEES (right) results in the simulated AR (top) and ETH datasets (bottom). Cyan represents indp, magenta represents indp-slam, red represents dc-msckf, blue represents dc-cmsckf-cslam, green represents dc-full-window and green represent dc-full-history. Please refer to the color figure.

the accuracy is further improved. It is worth noting that the efficient `dc-full-window` with feature constraint has close accuracy to its counterpart `dc-cmsckf-cslam`. Moreover, when including the historical common features, the distributed estimator becomes more accurate as expected. Interestingly, with only the common features over the sliding window, the `ce-cmsckf-cslam` can achieve the best performance on the AR dataset even without loop-closure. This is likely due to the fact that over the whole dataset all robots look in the same general location thus negating any benefit of loop-closure detection. As show in Tab. III and in the following real-world experiments, when robots do not have many overlapping views, the historical information plays an important role.

We additionally show the average Root Mean Square Error (RMSE) [33] and Normalized Estimation Error Squared (NEES) [34] of the distributed algorithms for Robot 0 in Fig. 4. The results for the other two robots are similar and are omitted here for space. The `indp` has the largest drift that can be reduced as shown by `indp-slam` and leveraging common features. The `dc-cmsckf-cslam` and `dc-full-window` have almost the same performance while the `dc-full-history` achieves the best accuracy. It is clear that all the distributed algorithms are conservative in nature (NEES is smaller than three) and have smaller NEES than the centralized ones.

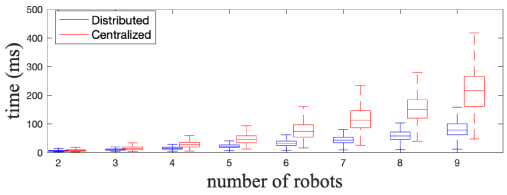

Fig. 5: Sequential propagation and update time (ms). Note that while decentralized can update in parallel, here we report its sequential timings.

TABLE IV: Timing for AR dataset. Millisecond mean and deviation.

Algorithm	Proposed	Combined
MSCKF update (window)	1.20 ± 0.94	2.88 ± 3.90
MSCKF update (hist)	4.11 ± 5.52	22.75 ± 159.65
Algorithm	**Constraint**	**No Constraint**
SLAM update (window)	0.10 ± 0.03	0.15 ± 0.16
SLAM update (hist)	0.17 ± 0.06	27.11 ± 160.95

B. Timing Analysis

1) Multiple Robots: We now investigate the computational efficiency of the proposed work in comparison to the centralized estimator using only common features over the sliding window. We compare the timing results of `dc-full-window` and `ce-cmsckf-cslam` while processing the same amount of measurements. We first investigate the performance as more robots are added to show the efficiency gains from the distributed formulation. The results in Fig. 5 show that as more robots are added, the centralized estimator quickly becomes computationally expensive while the distributed one is able to remain efficient since each robot only needs to propagate and update its own state and auto-covariance. Additionally, if one robot does not find common features in a given frame, the robot can update the estimator independently in the distributed case. On the contrary, the centralized estimator needs to collected all data, propagate, and update the whole state even if there are no common features. The distributed algorithm does have a slight increase in cost, which is due to the increase of common measurements from the additional robots.

2) Common VIO Features: We next investigate the efficiency of the common VIO feature nullspace projection and subsequent CI-EKF update introduced in Sec. IV-C.2. We report the update time for `dc-full-window` (window) and `dc-full-history` (history) without common SLAM features. The results presented in Tab. IV show that if we use the proposed method to first perform nullspace projection and separate each robot's systems into two systems (Proposed) we are able to outperform the naive way of performing nullspace projection on a "stacked" Jacobian containing all robot feature Jacobians (Combined). It is clear that in both algorithms, the proposed method is able to have less computational cost, especially in the historical case due to the proposed system reducing the number of measurements in the update. We also note that there is a high level of variance in the historical case due to loop-closure introducing large amounts of measurements in short intervals.

3) SLAM Constraint Update: Now we investigate the efficiency of the common SLAM feature update introduced in Sec. IV-D. Only common SLAM features that can be matched to another robot's SLAM feature are used to ensure

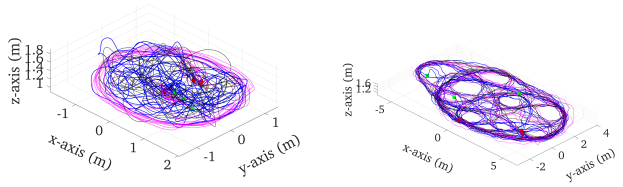

Fig. 6: TUM-VI groundtruth (left) and Vicon room groundtruth trajectories (right) TUM-VI trajectories are 146, 131, and 134 meters long, while the Vicon room datasets are 507, 509, and 501 meters long.

TABLE V: Relative pose error (RPE) on TUM-VI datasets in degrees / meters averaged over all robots for the dataset.

Algorithm	40m	60m	80m	100m	120m
indp-slam	1.818 / 0.093	2.833 / 0.126	2.604 / 0.154	2.774 / 0.185	2.716 / 0.215
ce-cmsckf	**1.358** / 0.071	**1.321** / 0.091	1.357 / 0.108	0.843 / 0.128	0.932 / 0.140
ce-cmsckf-cslam	1.758 / **0.069**	1.350 /**0.079**	**1.027** / **0.100**	**0.718** / 0.119	0.938 / **0.130**
dc-cmsckf	1.662 / 0.075	2.005 / 0.104	1.605 / 0.129	1.142 / 0.141	1.531 / 0.170
dc-cmsckf-cslam	1.800 / 0.080	2.642 / 0.093	2.233 / 0.106	1.544 / **0.114**	0.934 / 0.157
dc-full-window	1.768 / 0.075	2.218 / 0.091	1.788 / 0.109	1.257 / 0.123	**0.854** / 0.159
dc-full-history	**1.213** / **0.067**	**1.232** / **0.061**	**1.029** / **0.065**	**1.004** / **0.068**	**0.784** / **0.072**

TABLE VI: Relative pose error (RPE) on Vicon room dataset in degrees / meters averaged over all robots.

Algorithm	80m	100m	200m	300m	420m
indp-slam	**2.022** / **0.276**	2.416 / 0.334	3.872 / 0.613	5.222 / 0.870	8.045 / 1.189
ce-cmsckf-cslam	2.180 / 0.288	2.603 / 0.333	**2.771** / **0.548**	**3.050** / **0.770**	**3.557** / **1.044**
dc-full-window	2.197 / 0.281	**2.340** / **0.332**	3.322 / 0.580	3.670 / 0.804	5.977 / 1.102
dc-full-history	**1.271** / **0.145**	**1.307** / **0.151**	**1.346** / **0.158**	**1.267** / **0.157**	**1.343** / **0.160**

that both variants have the same number of measurements in the update. When we match features in the current window, the constraint update (Constraint) is slight more efficient than the naive way of grabbing all the measurements from the other robots (No Constraint) since all robots only have the most recent measurements (in most cases just one). During historical SLAM matching, by definition SLAM features are long feature tracks, and thus many measurements and clones states are associated with a historical SLAM feature. This means that after loop-closure in the naive case (No Constraint) we will process all measurements ever recorded for a SLAM feature which can easily reach many sliding windows in length. If instead we use the constraint update, only the two feature positions are involved, thus the update is extremely efficient in nature (bottom Tab. IV).

VI. Experimental Results

We have also evaluated the proposed distributed CL estimators on the TUM-VI dataset [35] and a hand collected 10 minute long Vicon room dataset (see Fig. 6).[2] Both datasets provide monochrome stereo images at 20Hz and IMU readings at 200Hz. We only leverage the left camera and initialize all robots based on the groundtruth orientation and position with zero velocity. The specific datasets we run on for the TUM-VI are the room1, room3, and room5. For the Vicon room dataset, the groundtruth has been generated using the vicon2gt utility [36]. The shorter TUM-VI dataset has more time periods where multiple robots are looking at the same environmental location (26.7% and 41.8% of the frames detected common features without and with loop-closure), thus provides a good insight into an expected performance in a multi-user AR case where many users are observing the same environment at the *same time*. On the other hand, the Vicon room dataset has near-zero time periods where we are able to detect common features between robots by matching the most recent features. Thus, we use the Vicon room dataset to show the accuracy gain from leveraging historical loop-closure information by matching to historical states (28.8% of the frames detected common loop-closure features).

[2] A video demo https://youtu.be/boHBcVoMKk8

A. TUM-VI Dataset

We use a sliding window of 11, a max of 5 SLAM features, max 30 VIO features per update, 300 active tracks, and perform online calibration of all parameters. For the historical method, we insert keyframes into our database at 5Hz and detect and match to historical keyframes at each timestep. We used a static weight of $\omega_i = 0.99$ and distribute the remaining weight to all other robot covariances used in the CI-EKF update, and for constraint measurement updates [see Eq. (29)], we used a value of $\omega_i = 0.995$ and injected a synthetic measurement noise of 2cm to relax the hard constraint.

The Relative Pose Error (RPE) [33] results are shown in Tab. V solidify the performance gains due to leveraging common features from other robot agents. The independent methods which leverage only independent VIO and SLAM feature updates have about three times the error compared to the distributed method which leverages loop-closure information. Additionally, we can see that all variations which leverage common features are able to reduce errors due to the additional information. It is also important to note that even though the distributed variants do not track the cross-covariances between robotic states, the use of CI allows the accuracy to be near the same level as that of the centralized algorithm, and in the case where we leverage historical information (which the centralized algorithm is unable to do), we can slightly outperform for longer trajectory length. The `dc-full-history` method, which leverages loop-closure information, has a relatively constant error as the trajectory lengths increase as expected (showing its drift-free nature).

B. Vicon Room Dataset

We now present results on the longer hand-held, approximately 500 meter and 10 minute trajectory. We use a sliding window of 11, a max of 20 SLAM features, max 30 VIO features per update, 200 active tracks, and perform online calibration of all parameters. The RPE results for different segment lengths can be found in Tab. VI and give the same conclusion as the previous TUM-VI dataset. It is also important to note that there is very similar performance of the `indp-slam` and `ce-cmsckf-cslam` methods (and their distributed equivalents). This is expected as there are no time periods in any of the robotic trajectories where robots are looking at the same location at the *same* time. Compared to these cases, we have huge accuracy gains due to the inclusion of common feature measurement constraints in the historical case, with halved orientation errors and a quarter of the position error at long trajectory lengths. We also plot the groundtruth, `indp-slam`, and `dc-full-history` Robot 0 trajectories in Fig. 1, which reinforces that by leveraging historical information we are able to prevent inherent drift in the loop-closure-free case.

VII. Conclusions and Future Work

In this work we have presented a distributed visual-inertial cooperative CL estimator that efficiently fuses constraints between robots and leverages temporal SLAM and loop-closure information. We have introduced two different ways to incorporate temporal SLAM features: (i) directly update using the other robot's measurements, and (ii) if both robots are estimating the SLAM feature, a constraint between the two feature positions is leveraged. We have adapted CI to ensure consistent fusion of loop-closure constraints to other agent's historical poses and SLAM features whose cross-correlations are unknown. Extensive simulation and real-world evaluations have demonstrated the performance of the proposed method in realistic scenarios and showed impressive accuracy gains over the single robot case. In the future we will focus on the practical deployment to a low-cost low-power multi-robot application and incorporate relative robot-to-robot measurements to further increase accuracy gains.

References

[1] P. Zhu, Y. Yang, W. Ren, and G. Huang, "Cooperative visual-inertial odometry," in *Proc. International Conference on Robotics and Automation*, Xi'an, China, May 2021.

[2] G. Huang, "Visual-inertial navigation: A concise review," in *Proc. International Conference on Robotics and Automation*, Montreal, Canada, May 2019.

[3] A. I. Mourikis and 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.

[4] T. Qin, P. Li, and S. Shen, "Vins-mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.

[5] P. Geneva, J. Maley, and G. Huang, "An efficient schmidt-ekf for 3D visual-inertial SLAM," in *Proc. Conference on Computer Vision and Pattern Recognition (CVPR)*, Long Beach, CA, Jun. 2019.

[6] S. I. Roumeliotis and G. A. Bekey, "Distributed multirobot localization," *IEEE transactions on robotics and automation*, vol. 18, no. 5, pp. 781–795, 2002.

[7] L. Luft, T. Schubert, S. I. Roumeliotis, and W. Burgard, "Recursive decentralized localization for multi-robot systems with asynchronous pairwise communication," *The International Journal of Robotics Research*, vol. 37, no. 10, pp. 1152–1167, 2018.

[8] R. Jung, C. Brommer, and S. Weiss, "Decentralized collaborative state estimation for aided inertial navigation," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 4673–4679.

[9] A. Martinelli, A. Oliva, and B. Mourrain, "Cooperative visual-inertial sensor fusion: The analytic solution," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 453–460, 2019.

[10] H. Xu, L. Wang, Y. Zhang, K. Qiu, and S. Shen, "Decentralized visual-inertial-UWB fusion for relative state estimation of aerial swarm," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 8776–8782.

[11] L. C. Carrillo-Arce, E. D. Nerurkar, J. L. Gordillo, and S. I. Roumeliotis, "Decentralized multi-robot cooperative localization using covariance intersection," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2013, pp. 1412–1417.

[12] P. Zhu and W. Ren, "Fully distributed joint localization and target tracking with mobile robot networks," *IEEE Transactions on Control Systems Technology*, 2020.

[13] K. Y. Leung, Y. Halpern, T. D. Barfoot, and H. H. Liu, "The utias multi-robot cooperative localization and mapping dataset," *The International Journal of Robotics Research*, vol. 30, no. 8, pp. 969–974, 2011.

[14] A. Martinelli, "Cooperative visual-inertial odometry: Analysis of singularities, degeneracies and minimal cases," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 668–675, 2020.

[15] P.-Y. Lajoie, B. Ramtoula, Y. Chang, L. Carlone, and G. Beltrame, "Door-slam: Distributed, online, and outlier resilient slam for robotic teams," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1656–1663, 2020.

[16] L. Paull, G. Huang, M. Seto, and J. Leonard, "Communication-constrained multi-auv cooperative slam," in *Proc. of the IEEE International Conference on Robotics and Automation*, Seattle, WA, May 26-30 2015, pp. 509–516.

[17] M. Karrer, P. Schmuck, and M. Chli, "Cvi-slam—collaborative visual-inertial slam," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 2762–2769, 2018.

[18] I. V. Melnyk, J. A. Hesch, and S. I. Roumeliotis, "Cooperative vision-aided inertial navigation using overlapping views," in *2012 IEEE International Conference on Robotics and Automation*. IEEE, 2012, pp. 936–943.

[19] K. Sartipi, R. C. DuToit, C. B. Cobar, and S. I. Roumeliotis, "Decentralized visual-inertial localization and mapping on mobile devices for augmented reality," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 2145–2152.

[20] P. Geneva, K. Eckenhoff, W. Lee, Y. Yang, and G. Huang, "OpenVINS: A research platform for visual-inertial estimation," in *Proc. of the IEEE International Conference on Robotics and Automation*, Paris, France, 2020. [Online]. Available: https://github.com/rpng/open_vins

[21] N. Trawny and S. I. Roumeliotis, "Indirect Kalman filter for 3D attitude estimation," University of Minnesota, Dept. of Comp. Sci. & Eng., Tech. Rep., Mar. 2005.

[22] A. B. Chatfield, *Fundamentals of High Accuracy Inertial Navigation*. AIAA, 1997.

[23] P. S. Maybeck, *Stochastic Models, Estimation, and Control*. London: Academic Press, 1979, vol. 1.

[24] S. Julier and J. K. Uhlmann, "General decentralized data fusion with covariance intersection," *Handbook of multisensor data fusion: theory and practice*, pp. 319–344, 2009.

[25] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *International Joint Conference on Artificial Intelligence*, Vancouver, BC, August 1981, pp. 674–679.

[26] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in *2011 IEEE international conference on Computer Vision (ICCV)*. IEEE, 2011, pp. 2564–2571.

[27] G. Huang, A. I. Mourikis, and S. I. Roumeliotis, "A first-estimates Jacobian EKF for improving SLAM consistency," in *Proc. of the 11th International Symposium on Experimental Robotics*, Athens, Greece, Jul. 14–17, 2008.

[28] ——, "Observability-based rules for designing consistent EKF SLAM estimators," *International Journal of Robotics Research*, vol. 29, no. 5, pp. 502–528, Apr. 2010.

[29] C. X. Guo, K. Sartipi, R. C. DuToit, G. A. Georgiou, R. Li, J. O'Leary, E. D. Nerurkar, J. A. Hesch, and S. I. Roumeliotis, "Resource-aware large-scale cooperative three-dimensional mapping using multiple mobile devices," *IEEE Transactions on Robotics*, vol. 34, no. 5, pp. 1349–1369, 2018.

[30] P. Geneva, K. Eckenhoff, and G. Huang, "A linear-complexity EKF for visual-inertial navigation with loop closures," in *Proc. International Conference on Robotics and Automation*, Montreal, Canada, May 2019.

[31] D. Gálvez-López and J. D. Tardós, "Bags of binary words for fast place recognition in image sequences," *IEEE Transactions on Robotics*, vol. 28, no. 5, pp. 1188–1197, October 2012.

[32] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart, "The euroc micro aerial vehicle datasets," *The International Journal of Robotics Research*, vol. 35, no. 10, pp. 1157–1163, 2016.

[33] Z. Zhang and D. Scaramuzza, "A tutorial on quantitative trajectory evaluation for visual (-inertial) odometry," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 7244–7251.

[34] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan, *Estimation with applications to tracking and navigation: theory algorithms and software*. John Wiley & Sons, 2004.

[35] D. Schubert, T. Goll, N. Demmel, V. Usenko, J. Stückler, and D. Cremers, "The tum vi benchmark for evaluating visual-inertial odometry," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 1680–1687.

[36] P. Geneva and G. Huang, "vicon2gt: Derivations and analysis," University of Delaware, Tech. Rep. RPNG-2020-VICON2GT, 2020, available: http://udel.edu/~ghuang/papers/tr_vicon2gt.pdf.