Cooperative Visual-Inertial Odometry

Pengxiang Zhu, Yulin Yang, Wei Ren and Guoquan Huang

***Abstract*— This paper studies the problem of multi-robot cooperative visual-inertial localization where each robot is equipped with *only* a single camera and IMU. We develop two cooperative visual-inertial odometry (C-VIO) algorithms within the multi-state constraint Kalman filter (MSCKF) framework, in which each robot utilizes not only its own measurements but constraints of common features co-observed with its neighbors within the current sliding window in order to improve the localization accuracy. The first centralized-equivalent algorithm tracks the robot-to-robot cross correlations and prioritizes the pose accuracy while requiring full capacity communication among all the robots during update. The second distributed algorithm ignores the robot-to-robot cross correlations to obtain a scalable, robust and efficient fully distributed structure where each robot only keeps its own states and communicates with its neighbors, while a covariance intersection (CI)-based update strategy is leveraged to guarantee consistency. The proposed algorithms are validated extensively in both Monte-Carlo simulations and real-world datasets, and shown to be able to achieve better accuracy with competitive efficiency.**

I. Introduction and Related Work

Multi-robot autonomous systems with the ability of communication and perception can collaborate to accomplish missions (e.g., area surveillance, environmental monitoring and disastrous rescue) more efficiently and robustly than a single robot. To successfully accomplish these tasks, robots are required to have robust high-precision 3D localization, in particular, when navigating GPS-denied environments. One promising solution is the visual-inertial navigation system (VINS), which is to fuse both the measurements from cameras and inertial measurement units (IMUs) [1]. These sensors are cheap and light-weight but they are complementary and able to provide rich environmental information, hence enabling highly-accurate motion estimation. To date, many VINS algorithms have been developed [2]–[7], and among them, multi-state constraint Kalman filter (MSCKF) [5] is arguably one of the most popular ones. However, most of current VINS are designed for the case of a *single* robot.

One intuitive strategy to localize a group of robots is to let each member run a single-robot VINS algorithm independently. However, additional geometric constraints (e.g., common feature observations, relative robot-to-robot measurements) can be explored in multi-robot systems to improve the localization performance, if robots communicate with each other. Then, it holds great potential to design cooperative VINS (C-VINS) algorithms for multi-robot systems. One of the challenges for developing a C-VINS algorithm, especially for a large robot team, is the stringent resource limitations associated with communication bandwidth, battery power and computational capacity. Therefore, it is highly desirable to design a $distributed$ algorithm which is scalable, robust and efficient for multi-robot systems.

Relative robot-to-robot pose or distance observations have been utilized to perform cooperative localization (CL) for a robot group [8]–[11]. Roumeliotis et al. [8] proposed an algorithm that distributes the computation of a centralized EKF to each robot. Though each robot propagates distributively, all-to-all robot communication is required in the update steps. To relax the communication requirements, Lukas et al. [9] approximated the robot-to-robot covariances instead of tracking the exact values. However, the filter's consistency can not be guaranteed. Carrillo et al. [10] introduced a consistent algorithm based on covariance intersection (CI) where each robot only estimates its own state and covariance while robot-to-robot cross correlations are ignored. However, these works address CL *only in 2D* where each robot's state only contains the planar position and heading.

Recently, relative robot-to-robot bearing observations are used in [12], [13] to localize a group of *two* robots in 3D environments. Martinelli et al. [12] provided an analytic solution to determine the relative state between two robots by establishing a Polynomial Equation System (PES). All the singularities and minimal cases that could harm the estimation performance were analyzed in [13]. However, the results are limited to the two-robot case using synchronized-calibrated cameras and unbiased inertial measurements. Nguyen et al. [14] introduced the Coupled Probabilistic Data Association Filter (CPDAF) to estimate the robots' poses in a common frame using relative robot-to-robot bearing and distance measurements. But the algorithm is centralized and is only validated in simulation environments. In addition, [8]–[14] all rely on relative robot-to-robot measurements which are difficult to achieve in many realistic scenarios due to robot sensing ability and environmental occlusion.

Alternatively, environmental common features can be used to improve the CL performance [15]–[18]. Cooperative SLAM (C-SLAM) algorithms enable multiple robots to build a joint map which is used to perform CL simultaneously. Paull et al. [15] developed a graph-based C-SLAM algorithm for autonomous underwater vehicles (AUVs) among which communications using acoustics is challenging. Lajoie et al. [16] introduced a fully distributed C-SLAM algorithm with a distributed outlier rejection for incorrect loop closures. The robots' estimated trajectories are retrieved through a distributed graph optimization. However, C-SLAM may become computational and memory expensive as the map size increases, which could be prohibitive in resource-constrained platforms. Being closest to our work, Melnyk et al. [17] introduced the CL-MSCKF algorithm, which utilized the

This work was partially supported by the University of Delaware College of Engineering, the NSF (IIS-1924897) and the NSF (CMMI-2027139).

P. Zhu and W. Ren are with the Department of Electrical and Computer Engineering, University of California, Riverside, CA 92521, USA.{pzhu008@ucr.edu,ren@ee.ucr.edu}

Y. Yang and G. Huang are with the Robot Perception and Navigation Group (RPNG), University of Delaware, Newark, DE 19716, USA. {yuyang,ghuang}@udel.edu

common feature measurements to improve the VIO accuracy, without building a map. The authors performed observability analysis to determine the minimum common features needed for computing the relative transformation between two robots. However, the CL-MSCKF follows a *centralized* formulation with *only* simulation results where all robots' IMU and camera measurements are collected and processed in a fusion center to estimate all the robots' states.

In this paper, we propose a fully distributed multi-robot pose estimation algorithm with a new centralized-equivalent counterpart as the benchmark. Commonly observed features from each robot are exploited to improve the CL accuracy. The proposed C-VIO algorithms are based on the state-of-the-art OpenVINS [19], possessing advanced features such as full online calibration (including camera intrinsic, camera-to-IMU spatiotemporal parameters) and First-Estimates Jacobian (FEJ) treatments [20] to improve estimation consistency. In particular, the main contributions of this paper include:

- We design a fully distributed cooperative (DISC)-VIO algorithm based on CI where each robot only maintains and estimates its own states by communicating with its neighbors. DISC-VIO is scalable, robust, and computationally and communicationally efficient while still improving the localization performance compared to a single robot system.
- We design a centralized-equivalent cooperative (CEC)-VIO algorithm as the benchmark, where, in contrast to [17], only camera measurements are processed centrally while each robot propagates and clones poses distributedly. CEC-VIO achieves equivalent accuracy as the centralized one in [17].
- Extensive Monte-Carlo simulations and real-world datasets are used to validate the proposed C-VIO algorithms in a variety of environments.

II. Problem Formulation

We define a robot set of N robots as $\mathcal{V}$. Each robot i ($i \in \mathcal{V}$) is mounted with a *monocular* VI sensor that measures the robot's self-motion and observes environmental features. The tracked features are divided into common features (observed by more than one robot) and independent features (observed by only one robot). We propose, respectively, a centralized-equivalent and a fully distributed C-VIO algorithm to estimate all robots' states by leveraging the MSCKF [21].

A. Visual-Inertial State

For any robot i, the state vector $\mathbf{x}_i$ consists of the current inertial navigation state $\mathbf{x}_{I_i}$, a sliding window of c historical IMU clones $\mathbf{x}_{C_i}$, and robot i's camera intrinsic and camera-to-IMU spatial/temporal parameters $\mathbf{x}_{W_i}$

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

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

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

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

where ${}^{I_{i,k}}_{G}\bar{q}$ is the JPL unit quaternion [22] parameterizing the rotation ${}^{I_{i,k}}_{G}\mathbf{R}$ from the global frame $\{G\}$ to robot i's IMU local frame $\{I_i\}$ at time step k , ${}^{G}\mathbf{p}_{I_{i,k}}$ and ${}^{G}\mathbf{v}_{I_{i,k}}$ are the position and velocity of the IMU expressed in the global frame, and $\mathbf{b}_{\omega_{i,k}}$ and $\mathbf{b}_{a_{i,k}}$ are the IMU's gyroscope and accelerometer biases, respectively. The constant calibration parameters $\mathbf{x}_{W_i}$ includes the time-offset ${}^{C_i}t_{I_i}$ between robot i's camera and IMU clocks, the spatial transformations $\{{}^{C_i}_{I_i}\bar{q}, {}^{C_i}\mathbf{p}_{I_i}\}$ from the IMU frame to the camera frame, and the camera intrinsics $\boldsymbol{\zeta}_i$ containing the focal lengths, the position of the principal point and the distortion parameters.

To perform EKF estimation, we use a minimal 3-dimensional representation $\delta\boldsymbol{\theta}$ to parameterize the orientation error as $\bar{q} \simeq \begin{bmatrix} \frac{\delta\boldsymbol{\theta}}{2} \\ 1 \end{bmatrix} \otimes \hat{\bar{q}}$ where $\otimes$ denotes quaternion multiplication [22] and $\hat{\cdot}$ denotes the estimated value. While for all the other variables in the state vector, we use the standard additive error definition (e.g., ${}^{G}\mathbf{p}_{I_i} = {}^{G}\hat{\mathbf{p}}_{I_i} + {}^{G}\tilde{\mathbf{p}}_{I_i}$ with $\tilde{\cdot}$ denoting the error).

B. IMU and Visual Feature Measurements

Each robot i's IMU measures the angular velocity, ${}^{I_i}\boldsymbol{\omega}_m$, and the linear acceleration, ${}^{I_i}\mathbf{a}_m$ as follows

$${}^{I_{i,k}}\boldsymbol{\omega}_m = {}^{I_{i,k}}\boldsymbol{\omega} + \mathbf{b}_{\omega_{i,k}} + \mathbf{n}_{\omega_{i,k}}, \tag{5}$$

$${}^{I_{i,k}}\mathbf{a}_m = {}^{I_{i,k}}\mathbf{a} + {}^{I_{i,k}}_{G}\mathbf{R}\,{}^{G}\mathbf{g} + \mathbf{b}_{a_{i,k}} + \mathbf{n}_{a_{i,k}}, \tag{6}$$

where ${}^{I_i}\boldsymbol{\omega}$ and ${}^{I_i}\mathbf{a}$ denote the IMU's true angular velocity and the true linear acceleration, $\mathbf{n}_{\omega_i}$ and $\mathbf{n}_{a_i}$ are the corresponding continue-time Gaussian white noise, and ${}^{G}\mathbf{g}$ is the gravitational acceleration.

Let us consider the observation of a feature f obtained by robot i's camera at time step k. Note that as time-offset inevitably exists between the IMU and camera clocks, the time step k expressed in robot i's camera clock, ${}^{C_i}t_k$, differs from the same instant expressed in its IMU clock, ${}^{I_i}t_k$. These two time instants are related by the time-offset

$${}^{I_i}t_k = {}^{C_i}t_k + {}^{C_i}t_{I_i}. \tag{7}$$

The perspective projection measurement function for this feature is given by

$$\mathbf{z}_i({}^{C_i}t_k) = \mathbf{w}_i\left(\boldsymbol{\Pi}\left({}^{C_i}\mathbf{p}_f({}^{I_i}t_k)\right), \boldsymbol{\zeta}_i\right) + \mathbf{n}_i({}^{C_i}t_k), \tag{8}$$

where $\mathbf{n}_i$ is the raw pixel noise assumed to be zero-mean Gaussian, $\mathbf{w}_i(\cdot)$ is the function mapping the normalized image coordinates onto the image plane according to the camera model (e.g., radial-tangential or fisheye [23]) and the corresponding intrinsics $\boldsymbol{\zeta}_i$. ${}^{C_i}\mathbf{p}_f({}^{I_i}t_k) = [x, y, z]^\top$ is the feature's position expressed in robot i's camera frame at timestamp ${}^{I_i}t_k$, which is related to the normalized image coordinates through the projection function $\boldsymbol{\Pi}({}^{C_i}\mathbf{p}_f) = \frac{1}{z}[x, y]^\top$. Further, we have

$${}^{C_i}\mathbf{p}_f({}^{I_i}t_k) = {}^{C_i}_{I_i}\mathbf{R}\,{}^{I_i}_{G}\mathbf{R}({}^{I_i}t_k)\left({}^{G}\mathbf{p}_f - {}^{G}\mathbf{p}_{I_i}({}^{I_i}t_k)\right) + {}^{C_i}\mathbf{p}_{I_i}, \tag{9}$$

where ${}^{G}\mathbf{p}_f$ is the feature's global position which is time invariant and unknown.

III. Centralized-equivalent Algorithm

In this section, we present the CEC-VIO which serves as the benchmark for the ensuing DISC-VIO.

A. Propagation and Stochastic Cloning

When any robot i's camera captures the k-th image with timestamp ${}^{C_i}t_k$, by using robot i's collected IMU measurements (5) and (6) over the time window $[{}^{I_i}t_{k-1}, {}^{I_i}t_k]$ (the corresponding time interval expressed in the IMU clock from time step $k-1$ to k), which is denoted as $\mathcal{I}_i$, we propagate the current inertial navigation state $\mathbf{x}_{I_i}$ at time step $k-1$ to the next time step k based on the IMU dynamics [24]:

$$\mathbf{x}_{I_{i,k}} = f(\mathbf{x}_{I_{i,k-1}}, \mathcal{I}_i, \mathbf{n}_{I_i}), \tag{10}$$

$$\Rightarrow \hat{\mathbf{x}}_{I_{i,k|k-1}} = f(\hat{\mathbf{x}}_{I_{i,k-1|k-1}}, \mathcal{I}_i, \mathbf{0}), \tag{11}$$

where $\mathbf{n}_{I_i}$ is the stacked vector of robot i's IMU measurement noises. The subscript $k|k-1$ for $\hat{\mathbf{x}}_{I_{i,k|k-1}}$ denotes the predicted estimate at time step k given the measurements up to time step $k-1$. Note that the clones $\mathbf{x}_{C_i}$ and the calibration parameters $\mathbf{x}_{W_i}$ do not evolve over time.

Based on the IMU dynamics, we can compute the state-transition matrix $\mathbf{\Phi}_i$ and the discrete noise covariance $\mathbf{Q}_i$ across the time interval $[{}^{I_i}t_{k-1}, {}^{I_i}t_k]$ [21]. The corresponding entries for the non-evolving state $\mathbf{x}_{C_i}$ and $\mathbf{x}_{W_i}$ in $\mathbf{\Phi}_i$ and $\mathbf{Q}_i$ are identity and zero, respectively. Then, robot i's state covariance is propagated as

$$\mathbf{P}_{i,k|k-1} = \mathbf{\Phi}_{i,k-1}\mathbf{P}_{i,k-1|k-1}\mathbf{\Phi}_{i,k-1}^\top + \mathbf{Q}_{i,k-1}. \tag{12}$$

It is important to note that once robot i has used another robot j's measurements of the common features to update its own state $\mathbf{x}_i$, these two robots' states are correlated afterwards. So in general, we have $\mathbf{P}_{ij} \neq 0$ ($j \in \mathcal{V}$, $j \neq i$). To achieve a distributed propagation step, we split the cross-covariance according to [8], [9]

$$\mathbf{P}_{ij} = \boldsymbol{\sigma}_{ij}\boldsymbol{\sigma}_{ji}^\top, \tag{13}$$

where the factor $\boldsymbol{\sigma}_{ij}$ maintained at robot i equals to $\mathbf{P}_{ij}$, and the factor $\boldsymbol{\sigma}_{ji}$ maintained at robot j is identity. Then, robot i can propagate the cross-covariance to all the other robots by updating the factors

$$\boldsymbol{\sigma}_{ij,k} = \mathbf{\Phi}_{i,k-1}\boldsymbol{\sigma}_{ij,k-1}. \tag{14}$$

After propagating to time step k, the state vector is augmented with an estimate of the IMU pose at the *true* time ${}^{I_i}t_k$, which will be a clone state in $\mathbf{x}_{C_i}$. This augmented state can be written as a function $\mathbf{g}(\mathbf{x}_i)$ of the current state vector [25]. We can compute the state Jacobian $\mathbf{J}_i$ from $\mathbf{g}(\mathbf{x}_i)$ and then augment robot i's current covariance by [26]

$$\mathbf{P}_{i,k} \leftarrow \begin{bmatrix} \mathbf{P}_{i,k} & \mathbf{P}_{i,k}\mathbf{J}_{i,k}^\top \\ \mathbf{J}_{i,k}\mathbf{P}_{i,k} & \mathbf{J}_{i,k}\mathbf{P}_{i,k}\mathbf{J}_{i,k}^\top \end{bmatrix}. \tag{15}$$

Note that we also need to augment the cross-covariance factors $\boldsymbol{\sigma}_{ij}$ by $\boldsymbol{\sigma}_{ij,k} \leftarrow \begin{bmatrix} \boldsymbol{\sigma}_{ij,k} \\ \mathbf{J}_{i,k}\boldsymbol{\sigma}_{ij,k} \end{bmatrix}$. Later, the cross correlations that are needed for updating the estimate will be reproduced using these factors when an update is triggered.

B. Update

To check if there exist common features, the extracted features' measurements with their descriptors from each robot's k-th image are collected at a centering robot, where runs a *centralized* update. For an independent feature, if it is lost or reaches the maximum track length (the sliding window size), we perform MSCKF update. While to optimally utilize the common feature measurements, MSCKF update is triggered when a common feature is lost in *all* the associated robots, or reaches the maximum track length of *any* associated robot, or contains the oldest clone in *any* associated robot's state.

Note that all the robots' states will be updated either using common features or independent features, as the robots' states are correlated. When the update is triggered for a feature, all the robots' states, covariances and cross-covariance factors together with this feature's measurements are collected at the centering robot. The current cross correlations are reproduced by

$$\mathbf{P}_{ij,k|k-1} = \boldsymbol{\sigma}_{ij,k}\boldsymbol{\sigma}_{ji,k}^\top, \quad \mathbf{P}_{ji,k|k-1} = \mathbf{P}_{ij,k|k-1}^\top, \tag{16}$$

where $i, j \in \mathcal{V}$ and $j \neq i$. Then, by using all the measurements and the corresponding clones, we triangulate the feature's global position ${}^G\mathbf{p}_f$ and compute the stacked vector of measurement residuals, $\mathbf{r}$, associated with this feature based on the measurement model (8), (9). The linearized residual system can be written as

$$\mathbf{r} = \mathbf{H}_x\tilde{\mathbf{x}}_{k|k-1} + \mathbf{H}_f{}^G\tilde{\mathbf{p}}_f + \mathbf{n}, \tag{17}$$

where $\tilde{\mathbf{x}} = [\tilde{\mathbf{x}}_1, \cdots, \tilde{\mathbf{x}}_N]$, $\mathbf{H}_x$ and $\mathbf{H}_f$ denote the corresponding stacked Jacobians with respect to the state and the feature, and $\mathbf{n}$ is the stacked noise vector. To avoid storing the feature's position in our state, we project system (17) onto the nullspace of feature Jacobian $\mathbf{H}_f$. With matrix $\mathbf{N}$ whose columns form a basis of this nullspace, we can have:

$$\mathbf{N}^\top\mathbf{r} = \mathbf{N}^\top\mathbf{H}_x\tilde{\mathbf{x}}_{k|k-1} + \mathbf{N}^\top\mathbf{H}_f{}^G\tilde{\mathbf{p}}_f + \mathbf{N}^\top\mathbf{n}, \tag{18}$$

$$\Rightarrow \mathbf{r}'_k = \mathbf{H}'_x\tilde{\mathbf{x}}_{k|k-1} + \mathbf{n}'. \tag{19}$$

Equation (19) can be directly used to perform the EKF update. After that, the new cross correlation $\mathbf{P}_{ij,k|k}$ are decomposed using (13) again. The resulting factors together with the states and covariances will be sent back to the corresponding robots.

IV. Fully Distributed Algorithm

In this section, we present the DISC-VIO in which the robot-to-robot cross-correlations are conservatively dropped off via covariance intersection.

A. Propagation and Stochastic Cloning

When any robot i receives the k-th image with timestamp ${}^{C_i}t_k$, robot i's IMU measurements of the angular velocity, ${}^{I_i}\boldsymbol{\omega}_m$, and the linear acceleration, ${}^{I_i}\mathbf{a}_m$, over the time interval $[{}^{I_i}t_{k-1}, {}^{I_i}t_k]$ are utilized to propagate the state and the covariances from time step $k-1$ to k based on (11) and (12). After that, an estimate of the current IMU pose is appended to the state vector as a clone state and the covariance is augmented as (15). Each robot performs these two steps individually.

B. Update

When any robot i's camera captures the k-th image, it receives the extracted features' descriptors of the latest images from its communication neighbors. Feature matching is performed in pair between robot i's k-th image features and the received ones to check if there exist common features. Then, the features tracked at robot i can be classified as

an independent feature or a common feature. If a feature is a common feature, robot i will store its neighbors' measurements.

When a feature is lost or reaches its maximum track length at robot i, we trigger the update. If it is an independent feature, the standard single-robot MSCKF update [21] is performed. Note that if this feature is a common feature at robot i, it could have been tracked across multiple frames in more than two robots. Let $\mathcal{N}_f$ be a robot set in which this feature is tracked. So $i \in \mathcal{N}_f$. Further, we denote $\{j_1, \cdots, j_L\}$ as a subset of $\mathcal{N}_f$ that excludes robot i. Additionally, we let $\mathbf{n}_{(\cdot)}$ be the corresponding stacked noise vector for a stacked residual system. To process this common feature, robot i receives the associated clones, states and the corresponding covariances from robot j_l $(l = 1, \cdots, L)$. By using all available measurements from the robots in $\mathcal{N}_f$, we triangulate the feature's global position. After that, we use robot i's measurements to compute a stacked linearized residual system

$$\mathbf{r}_i = \mathbf{H}_{i,x}\tilde{\mathbf{x}}_{i,k} + \mathbf{H}_{i,f}{}^{G}\tilde{\mathbf{p}}_f + \mathbf{n}_i. \tag{20}$$

We perform Givens rotations [27] to zero-out rows in $\mathbf{H}_{i,f}$ with indices larger than 3, and apply the same Givens rotations to $\mathbf{H}_{i,x}$ and $\mathbf{r}_{i,k}$. Then, system (20) is split into a subsystem that depends on the feature's position and another subsystem that does not.

$$\begin{bmatrix}\mathbf{r}_{i1}\\ \mathbf{r}_{i2}\end{bmatrix} = \begin{bmatrix}\mathbf{H}_{i1,x}\\ \mathbf{H}_{i2,x}\end{bmatrix}\tilde{\mathbf{x}}_{i,k} + \begin{bmatrix}\mathbf{H}_{i1,f}\\ \mathbf{0}\end{bmatrix}{}^{G}\tilde{\mathbf{p}}_f + \begin{bmatrix}\mathbf{n}_{i1}\\ \mathbf{n}_{i2}\end{bmatrix}. \tag{21}$$

Similarly, we can use robot j_l's measurements to compute a stacked linearized residual system

$$\mathbf{r}_{j_l,k} = \mathbf{H}_{j_l,x}\tilde{\mathbf{x}}_{j_l,k} + \mathbf{H}_{j_l,f}{}^{G}\tilde{\mathbf{p}}_f + \mathbf{n}_{j_l}, \tag{22}$$

which is separated into the following two subsystems by performing Givens rotations on $\mathbf{H}_{j_l,f}$, $\mathbf{r}_{j_l,k}$ and $\mathbf{H}_{j_l,x}$.

$$\begin{bmatrix}\mathbf{r}_{j_l1}\\ \mathbf{r}_{j_l2}\end{bmatrix} = \begin{bmatrix}\mathbf{H}_{j_l1,x}\\ \mathbf{H}_{j_l2,x}\end{bmatrix}\tilde{\mathbf{x}}_{j_l,k} + \begin{bmatrix}\mathbf{H}_{j_l1,f}\\ \mathbf{0}\end{bmatrix}{}^{G}\tilde{\mathbf{p}}_f + \begin{bmatrix}\mathbf{n}_{j_l1}\\ \mathbf{n}_{j_l2}\end{bmatrix}, \tag{23}$$

Note that the bottom reduced system in (21) is only associated with robot i's state, which can be directly used to update robot i's estimate by using the EKF. Stacking the top systems in (21) and (23) for all j_l $(l = 1, \cdots, L)$, we obtain a new system that dependents on the feature's position.

$$\begin{bmatrix}\mathbf{r}_{i1}\\ \mathbf{r}_{j_11}\\ \vdots\\ \mathbf{r}_{j_L1}\end{bmatrix} = \mathbf{Diag}\{\begin{bmatrix}\mathbf{H}_{i1,x}\\ \mathbf{H}_{j_11,x}\\ \vdots\\ \mathbf{H}_{j_L1,x}\end{bmatrix}\}\begin{bmatrix}\tilde{\mathbf{x}}_{i,k}\\ \tilde{\mathbf{x}}_{j_1,k}\\ \vdots\\ \tilde{\mathbf{x}}_{j_L,k}\end{bmatrix} + \begin{bmatrix}\mathbf{H}_{i1,f}\\ \mathbf{H}_{j_11,f}\\ \vdots\\ \mathbf{H}_{j_L1,f}\end{bmatrix}{}^{G}\tilde{\mathbf{p}}_f + \begin{bmatrix}\mathbf{n}_{i1}\\ \mathbf{n}_{j_11}\\ \vdots\\ \mathbf{n}_{j_L1}\end{bmatrix}$$

$$\Rightarrow \mathbf{r}_i = [\bar{\mathbf{H}}_{i,x}\ \bar{\mathbf{H}}_{j_1,x}\ \cdots\ \bar{\mathbf{H}}_{j_L,x}]\begin{bmatrix}\tilde{\mathbf{x}}_{i,k}\\ \tilde{\mathbf{x}}_{j_1,k}\\ \vdots\\ \tilde{\mathbf{x}}_{j_L,k}\end{bmatrix} + \bar{\mathbf{H}}_f{}^{G}\tilde{\mathbf{p}}_f + \bar{\mathbf{n}}_i, \tag{24}$$

where $\mathbf{Diag}\{\cdot\}$ denotes the block-diagonal matrix constructed from the elements. Then, we project system (24) onto the nullspace of $\bar{\mathbf{H}}_f$ and obtain a system that is independent of ${}^{G}\tilde{\mathbf{p}}_f$.

$$\mathbf{r}'_i = \begin{bmatrix}\bar{\mathbf{H}}'_{i,x} & \bar{\mathbf{H}}'_{j_1,x} & \cdots & \bar{\mathbf{H}}'_{j_L,x}\end{bmatrix}\begin{bmatrix}\tilde{\mathbf{x}}_{i,k}\\ \tilde{\mathbf{x}}_{j_1,k}\\ \vdots\\ \tilde{\mathbf{x}}_{j_L,k}\end{bmatrix} + \bar{\mathbf{n}}'_i, \tag{25}$$

which contains additional constrains of the common feature. Here, the noise vector $\bar{\mathbf{n}}'_i$ is assumed to be zero-mean Gaussian with covariance $\bar{\mathbf{R}}_i$. It is important to note that performing the standard EKF update using (25) will yield an inconsistent estimate which degrades the performance seriously, since $\tilde{\mathbf{x}}_i$ and $\tilde{\mathbf{x}}_{j_l}$ $(l = 1, \cdots, L)$ are in general correlated with *unknown* correlations. To guarantee consistency, we adopt the CI algorithm [28]:

$$\begin{bmatrix}\frac{1}{\omega_i}\mathbf{P}_i & & \\ & \ddots & \\ & & \frac{1}{\omega_{j_L}}\mathbf{P}_{j_L}\end{bmatrix} > \begin{bmatrix}\mathbf{P}_i & \cdots & \mathbf{P}_{ij_L}\\ \vdots & \ddots & \vdots\\ \mathbf{P}_{ij_L}^{\top} & \cdots & \mathbf{P}_{ij_L}\end{bmatrix}, \tag{26}$$

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 $\begin{bmatrix}\tilde{\mathbf{x}}_i, \tilde{\mathbf{x}}_{j_1}, \cdots, \tilde{\mathbf{x}}_{j_L}\end{bmatrix}^{\top}$. The weights $\omega_i > 0$, $\omega_{j_l} > 0$ and $\omega_i + \sum_l \omega_{j_l} = 1$, for $l = 1, \cdots, L$. Substitute (26) into the EKF using system (25), we obtain the following equations to update robot i' estimate.

$$\delta\mathbf{x}_{i,k} = \frac{1}{\omega_i}\mathbf{P}_{i,k|k-1}\bar{\mathbf{H}}'^{\top}_{i,x}\mathbf{S}_k^{-1}\mathbf{r}'_{ij}, \tag{27}$$

$$\mathbf{P}_{i,k|k} = \frac{1}{\omega_i}\mathbf{P}_{i,k|k-1} - \frac{1}{\omega_i^2}\mathbf{P}_{i,k|k-1}\bar{\mathbf{H}}'^{\top}_{i,x}\mathbf{S}_k^{-1}\bar{\mathbf{H}}'_i\mathbf{P}_{i,k|k-1}, \tag{28}$$

$$\mathbf{S}_k = \sum_{o\in\mathcal{N}_f}\frac{1}{\omega_o}\bar{\mathbf{H}}'_{o,x}\mathbf{P}_{o,k|k-1}\bar{\mathbf{H}}'^{\top}_{o,x} + \mathbf{R}_i, \tag{29}$$

where $\delta\mathbf{x}_i$ is the correction to the state estimate $\hat{\mathbf{x}}_i$.

V. Experimental Results

We compared our algorithms to the case where each robot performs independent MSCKF using a single camera and IMU. We represent this case as OpenVINS for simplicity. Compared with the proposed CEC-VIO and DISC-VIO, the only difference of settings in OpenVINS is that no common features are detected and used. Note that we do not include any SLAM features [29], the features that can be tracked beyond the clone window, in the state vector. To fully test the proposed algorithms, we first performed extensive Monte-Carlo simulations on the synthetic data generated from realistic trajectories in three different scenarios. Following that, a real-world test was performed.

The weights in CI (26) are usually chosen by minimizing the uncertainty of the posterior covariance [10], [28]. In DISC-VIO, one can gain these weights by minimizing the trace of $\mathbf{P}_{i,k|k}$ in (28) at every update step. This optimization problem is non-convex and could be time consuming as the size of $\mathcal{N}_f$ increases. Through extensive simulations and the real-world test, we find that fixed weights in DISC-VIO work well and these weights are easy to choose. To speed up the process of DISC-VIO and demonstrates its applicability in resource-constrained platforms, we test DISC-VIO with

TABLE I: Simulation parameters. Note that all algorithms and robots use the same parameters as reported in this table.

Parameter	Value	Parameter	Value
Cam Freq(hz)	10	IMU Freq(hz)	200
Max Clones	12	Num Feats Per Frame	120
Gyro White Noise	1.6968e-4	Gyro Rand Walk	1.9393e-5
Accel White Noise	2.0000e-3	Accel Rand Walk	3.0000e-3
Pixel Noise	1	FEJ	True

fixed weights. In both simulations and the real-world tests, for any robot i running DISC-VIO, we let $\omega_{j_l} = 0.008$ $(l = 1, \cdots, L)$ and then $\omega_i = 1 - \sum_l \omega_{j_l}$. For example, when updating robot 0's estimate using a common features that is also tracked by robot 1 and robot 2, we have $\omega_1 = \omega_2 = 0.008$ and $\omega_0 = 0.984$.

A. Monte-Carlo Simulations

We simulated a team of three robots in different environments depicted in Fig. 1. In each group, one robot follows a realistic trajectory which is used to create the trajectories of the other two robots by adding orientation and position offsets. The IMU and camera measurements are generated from these simulated trajectories. The first simulated dataset, termed as "EuRoC_MH5", is based on the 97 meters "Machine_Hall_05" dataset from EuRoC MAV datasets [30]. The second simulated dataset is called "Udel_gore" based on a 240 meters dataset which traverses three floors in the University of Delaware's Gore Hall [19]. The third dataset is called "Tum_corridor" which is highly dynamic and based on the 295 meters "Tum Corridor 1" dataset from the TUM VI datasets [31]. To ensure a fair comparison, the same parameters summarized in Table I were used in all algorithms for three robots.

We performed 50 Monte-Carlo runs on each simulated dataset. The Root Mean Square Error (RMSE) and Normalized Estimation Error Squared (NEES) results averaged over all runs for robot 0 (R0) are given in Fig. 1. Results for robots 1 (R1) and 2 (R2) are similar and omitted here. Table II provides the RMSE results averaged over all runs and time steps for three robots. Fig. 1 and Table II show that the proposed CEC-VIO and DISC-VIO outperform OpenVINS in all three simulated scenarios. Especially for CEC-VIO, the errors can be reduced by more than half. As shown on the right in Fig. 1, DISC-VIO has the smallest NEES as expected, since CI makes the estimate more conservative.

B. Real-World Tests

We further evaluated the proposed CEC-VIO and DISC-VIO in the room scenarios from the TUM VI datasets [31] which provide 20 Hz stereo image (*only* the left image is leveraged in the test), 200 Hz IMU measurements and the accurate pose ground truths from a motion capture system for entire trajectories. We simultaneously read three bags collected in the same room to mimic a three-robot cooperative case. Three robots' trajectories shown in Fig. 2 are 146, 135 and 131 meters long, respectively.

We initialized three robots' inertial navigation states with the corresponding pose ground truths, zero velocities and zero gyroscope and accelerometer biases. All extrinsics and intrinsics were initialized with the values provided in the datasets and each robot's camera-to-IMU time offsets were set to zeros. Features were uniformly extracted using FAST [32] and tracked for each robot's image stream or matched across different robots' images using ORB [33] with 8-point RANSAC to reject outliers. Up to 150 features were extracted and tracked over a sliding window with size 11. For a fair comparison, filters on each robot were initialized with the same values, and the same settings above were used for all the robots and algorithms.

TABLE II: The average RMSE in degree/meters for three simulated datasets using different algorithms.

	EuRoC	Gore	Tum	Average
R0 OpenVINS	0.68 / 0.22	0.45 / 0.23	0.48 / 0.24	0.54 / 0.23
R0 DISC-VIO	0.35 / 0.11	0.32 / 0.18	0.26 / 0.16	0.31 / 0.15
R0 CEC-VIO	0.22 / 0.06	0.24 / 0.12	0.22 / 0.09	0.22 / 0.09
R1 OpenVINS	0.62 / 0.21	0.46 / 0.21	0.50 / 0.25	0.53 / 0.22
R1 DISC-VIO	0.33 / 0.11	0.31 / 0.18	0.26 / 0.16	0.30 / 0.15
R1 CEC-VIO	0.22 / 0.06	0.24 / 0.12	0.22/ 0.10	0.22 / 0.09
R2 OpenVINS	0.59 / 0.17	0.50 / 0.23	0.49 / 0.23	0.53 / 0.21
R2 DISC-VIO	0.32 / 0.11	0.32 / 0.18	0.26 / 0.16	0.30 / 0.15
R2 CEC-VIO	0.22 / 0.06	0.23 / 0.12	0.22 / 0.09	0.22 / 0.09

TABLE III: The average RMSE resutls in degree/meters for three robots using different algorithms.

	OpenVINS	DISC-VIO	CEC-VIO
Robot 0	2.978 / 0.132	1.430 / 0.059	1.145 / 0.056
Robot 1	2.220 / 0.112	1.375 / 0.070	0.904 / 0.059
Robot 2	2.224 / 0.130	1.244 / 0.077	1.126 / 0.059

It is worth noting that throughout the entire trajectories, common features only occasionally appear. Table III shows the average RMSE results of three robots and Fig. 3 shows the Relative Pose Error (RPE) results computed over three robots' trajectories. It is clear that in term of both RMSE and RPE, DISC-VIO and CEC-VIO improve the accuracy for all three robots with limited common features, and CEC-VIO achieves the best performance as expected. Additionally, we timed the complete execution time of all three robots at each frame that mainly includes the timing for propagation, tracking, matching across robots for C-VIO, update and marginalization. As shown in Fig. 4, probably due to the fixed weights used in CI, DISC-VIO only takes a bit more time than OpenVINS.

VI. CONCLUSIONS AND FUTURE WORK

In this paper, we have developed two cooperative VIO (C-VIO) algorithms for multi-robot cooperative localization. (i) The first one is CEC-VIO that has centralized performance but distributed state propagation and clone, which sets up the benchmark performance; (ii) The second one is DISC-VIO which leverages CI to design a fully distributed C-VIO algorithm. DISC-VIO is shown to be more scalable and efficient and robust due to its fully distributed architecture. Common features tracked over the sliding-window are exploited to improve the localization accuracy for both algorithms. The

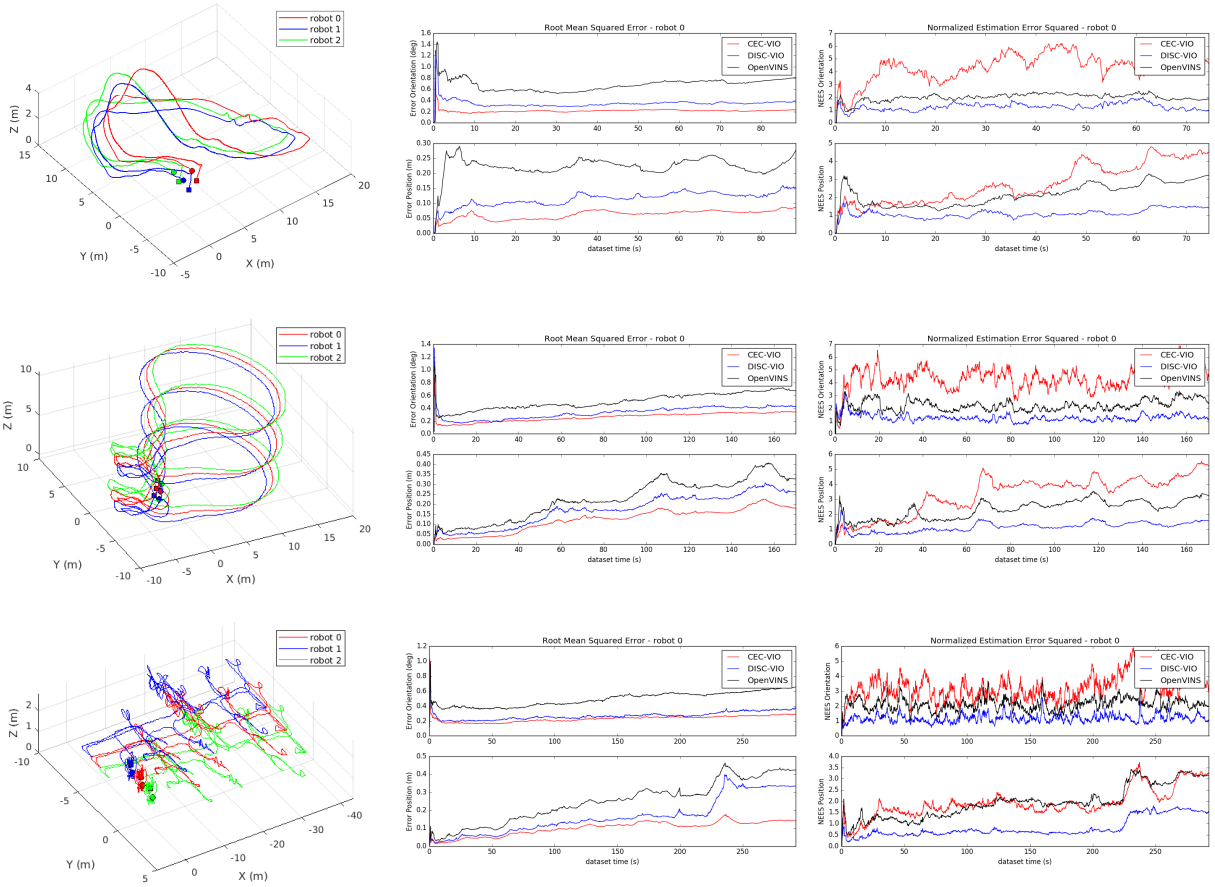

Fig. 1: Three robots' true trajectories (squares denote the starts and circles denote the ends) and robot 0's average RMSE and NEES results in the simulated EuRoC_MH (upper), Udel_gore (middle) and Tum_corridor datasets (bottom).

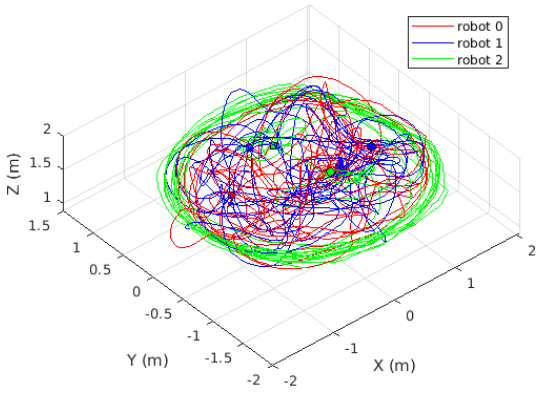

Fig. 2: Three robots' true trajectories. Squares denote the starts and circles denote the ends.

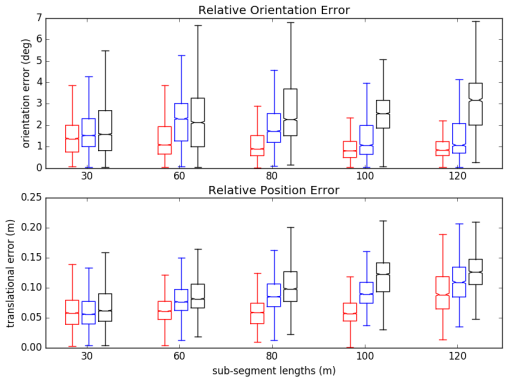

Fig. 3: The RPE results for different segment lengths computed over three robots' trajectories. Red corresponds to CEC-VIO, blue corresponds to DISC-VIO and black corresponds to OpenVINS.

effectiveness of our approaches have been validated through extensive Monte-Carlo simulations and real-world datasets. In the future, we will further improve our algorithms by including SLAM features in the state vector as well as using robot-to-robot measurements (if available), and deploy the the proposed algorithms on real multi-MAV systems.

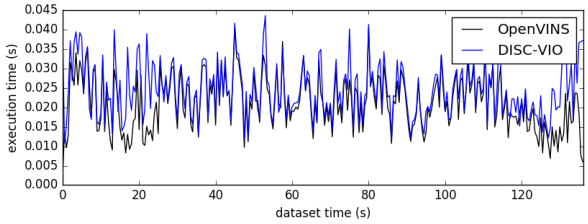

Fig. 4: The timing results for different algorithms.

REFERENCES

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

[2] R. Mur-Artal and J. D. Tardós, "Visual-inertial monocular slam with map reuse," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 796–803, 2017.

[3] 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.

[4] P. Geneva, J. Maley, and G. Huang, "An efficient schmidt-ekf for 3d visual-inertial slam," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2019, pp. 12 105–12 115.

[5] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint kalman filter for vision-aided inertial navigation," in *Proc. of the IEEE International Conference on Robotics and Automation*. IEEE, 2007, pp. 3565–3572.

[6] M. Li and A. Mourikis, "High-precision, consistent EKF-based visual-inertial odometry," *International Journal of Robotics Research*, vol. 32, no. 6, pp. 690–711, 2013.

[7] Z. Huai and G. Huang, "Robocentric visual–inertial odometry," *The International Journal of Robotics Research*, 2019.

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

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

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

[11] G. Huang, N. Trawny, A. I. Mourikis, and S. I. Roumeliotis, "On the consistency of multi-robot cooperative localization." in *Proc. of the Robotics: Science and Systems Conference*, 2009, pp. 65–72.

[12] 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.

[13] 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.

[14] T. Nguyen, K. Mohta, C. J. Taylor, and V. Kumar, "Vision-based multi-mav localization with anonymous relative measurements using coupled probabilistic data association filter," in *Proc. of the IEEE International Conference on Robotics and Automation*. IEEE, 2020, pp. 3349–3355.

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

[16] 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. 2, no. 2, pp. 1656–1663, 2020.

[17] I. Melnyk, J. Hesch, and S. Roumeliotis, "Cooperative vision-aided inertial navigation using overlapping views," in *Proc. of the IEEE International Conference on Robotics and Automation*, Saint Paul, MN, May 14–18, 2012, pp. 936–943.

[18] P. Zhu and W. Ren, "Multi-robot joint visual-inertial localization and 3-d moving object tracking," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2020, pp. 11 573–11 580.

[19] 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*. IEEE, 2020, pp. 4666–4672.

[20] G. Huang, A. I. Mourikis, and S. I. Roumeliotis, "Analysis and improvement of the consistency of extended Kalman filter-based SLAM," in *Proc. of the IEEE International Conference on Robotics and Automation*, Pasadena, CA, May 19-23 2008, pp. 473–479.

[21] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint Kalman filter for vision-aided inertial navigation," in *Proceedings of the IEEE International Conference on Robotics and Automation*, Rome, Italy, Apr. 10–14, 2007, pp. 3565–3572.

[22] 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.

[23] OpenCV Developers Team, "Open source computer vision (OpenCV) library," Available: http://opencv.org.

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

[25] K. Eckenhoff, P. Geneva, J. Bloecker, and G. Huang, "Multi-camera visual-inertial navigation with online intrinsic and extrinsic calibration," in *Proc. of the IEEE International Conference on Robotics and Automation*, Montreal, Canada, May 2019.

[26] M. Li and A. I. Mourikis, "Online temporal calibration for Camera-IMU systems: Theory and algorithms," *International Journal of Robotics Research*, vol. 33, no. 7, pp. 947–964, June 2014.

[27] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical recipes 3rd edition: The art of scientific computing*. Cambridge university press, 2007.

[28] 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.

[29] M. Li and A. I. Mourikis, "Optimization-based estimator design for vision-aided inertial navigation," in *Robotics: Science and Systems*, Berlin, Germany, June 2013, pp. 241–248.

[30] 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.

[31] D. Schubert, T. Goll, N. Demmel, V. Usenko, J. Stueckler, and D. Cremers, "The tum vi benchmark for evaluating visual-inertial odometry," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, October 2018.

[32] E. Rosten, R. Porter, and T. Drummond, "Faster and better: A machine learning approach to corner detection," *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, no. 1, pp. 105–119, 2008.

[33] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2011, pp. 2564–2571.