Data-Efficient Collaborative Decentralized Thermal-Inertial Odometry

Vincenzo Polizzi[1,2], Robert Hewitt[2], Javier Hidalgo-Carrió[1], Jeff Delaune[2] and Davide Scaramuzza[1]

***Abstract*—We propose a system solution to achieve data-efficient, decentralized state estimation for a team of flying robots using thermal images and inertial measurements. Each robot can fly independently, and exchange data when possible to refine its state estimate. Our system front-end applies an online photometric calibration to refine the thermal images so as to enhance feature tracking and place recognition. Our system back-end uses a covariance-intersection fusion strategy to neglect the cross-correlation between agents so as to lower memory usage and computational cost. The communication pipeline uses Vector of Locally Aggregated Descriptors (VLAD) to construct a request-response policy that requires low bandwidth usage. We test our collaborative method on both synthetic and real-world data. Our results show that the proposed method improves by up to 46 % trajectory estimation with respect to an individual-agent approach, while reducing up to 89 % the communication exchange. Datasets and code are released to the public, extending the already-public JPL xVIO library.**

***Index Terms*—Collaborative Localization, Thermal-Inertial Odometry, Space Robotics and Automation**

Supplementary Material

For code and datasets please visit: https://rpg.ifi.uzh.ch/xctio

I. Introduction

With the successful deployment of the Ingenuity Mars Helicopter and the Perseverance rover on Mars,[1] multi-agent collaborative tasks are now being carried out on another world for the first time. Ingenuity performed 19 flights during its first year of operations, many of which were done in direct support of Perseverance's exploration of Jezero crater. This continues a new era of exploration for NASA, started by the MarCO mission in 2018 [1], which saw a pair of co-dependent cubesats investigate Mars for the first time. Next, NASA's CADRE mission plans to send a group of small rovers to cooperatively explore the Lunar surface in 2024 as part of its Commercial Lunar Payload Services program [2].

Manuscript received: February, 24, 2022; Revised May, 19, 2022; Accepted July, 4, 2022.

This paper was recommended for publication by Editor Pauline Pounds upon evaluation of the Associate Editor and Reviewers' comments.

This work was funded by the Combat Capabilities Development Command Soldier Center and Army Research Laboratory. This research was carried out at the Jet Propulsion Laboratory, California Institute of Technology, and was sponsored by the JPL Visiting Student Research Program (JVSRP) and the National Aeronautics and Space Administration (80NM0018D0004).

[1]Robotics and Perception Group, University of Zurich, Switzerland

[2]Jet Propulsion Laboratory, California Institute of Technology, USA

Digital Object Identifier (DOI): see top of this page.

[1]https://www.nytimes.com/2022/02/15/science/mars-nasa-perseverance.html

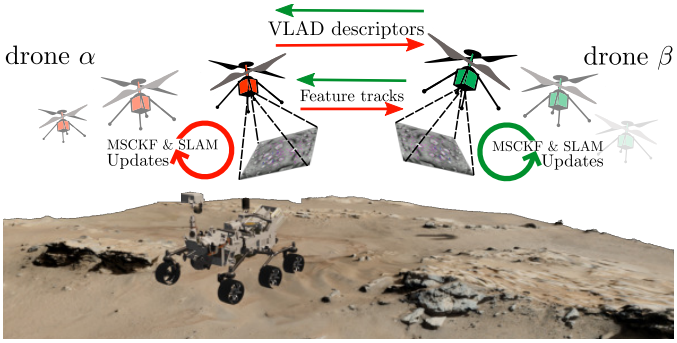

Fig. 1: Schematic representation of the proposed method with two flying robots in a Mars scenario.

This trend towards smaller, cooperative agents is reflected by the more dangerous terrain NASA wishes to explore in future missions to the Moon, Mars and beyond. These areas include Recurring Slope Linae on Mars and lava tubes on the Moon and Mars. Multiple small agents can be released from larger rovers or deployed from landers to explore areas that are too risky for a single agent to explore. It's also possible for multiple agents to explore large regions quickly, and deviate from the primary mission task without risk of jeopardizing the primary mission timeline.

The ability for agents to localize themselves in an unknown environment with no prior infrastructure (e.g., GPS) is a key capability for autonomous operations of multiple agents (see Fig. 1). Visual-Inertial Odometry (VIO) [3], [4] is one navigation alternative that the Ingenuity Mars Helicopter utilizes to estimate its pose during flights [5]. Additionally, pre-cursors to the CADRE mission, such as the Autonomous-PUFFER research task [6], have demonstrated the ability of multiple agents to exchange information while performing localization to improve their overall estimate by working together while exploring their work area.

At the same time, there has also been growing interest and work to incorporate new sensing inputs to autonomous navigation algorithms that do not rely on day light conditions and can be used to operate these algorithms in extreme locations such as lava tubes that are deprived of external lighting. One of the sensor technologies being investigated for this purpose are thermal cameras, which measure thermal radiation from all objects in their field of view [7]. These types of cameras allow autonomous agents to operate in all types of lighting conditions and require no external lighting source.

This paper aims to incorporate these two separate threads (multi-agent localization and thermal sensing) into a single system based around multiple aerial vehicles to investigate the

benefits that such a system could bring to future missions. We present a collaborative VIO system architecture that allows agents to operate in a decentralized way while still making use of information that is received from other agents. In addition, we have incorporated recent advancements in thermal image calibration into our system to improve VIO performance on thermal imagery. Our contributions are:

- The development of a scalable, from 1 to N agents, Collaborative Thermal-Inertial Odometry using a photometric calibration algorithm.
- The incorporation of Binary VLAD descriptors in a thermal-inertial odometry solution.
- A scalable and light communication approach over a decentralized flying robot system.
- Code and datasets are released as part of the already public JPL xVIO library[2] [8].

II. Related Work

State estimation for multi-agent systems is a well-known and established problem which has recently gained attention. The solutions can be divided into centralized and decentralized approaches. The former relies on a central unit acting as a server, fusing incoming information from all the agents [9], [10]. The latter directly fuses data onboard requiring efficient fusion processes and low data exchange [11]. While a centralized version is easier to implement, it relies on a central entity to always be reachable, never fail, and scale with the number of agents, limiting the range of exploration. This work focuses on the decentralized approach to be robust and scalable [12].

Decentralized Simultaneous Localization and Mapping (SLAM) faces the problem of exploiting communication. Indirect measurements through the registration of observations are the preferable approach in many scenarios [13]. Place recognition has emerged as a compact and data-efficient solution [14]. The work in [15] uses Vector of Locally Aggregated Descriptors (VLADs) generated by the NetVLAD [16] to reduce the communication bandwidth. Instead of sending information to all the other agents, the agent sends data only to the robot assigned with a specific descriptor. Although this strategy lowers the used bandwidth, it limits the system's scalability due to the a priori assignment of a centroid of the clusterized feature space. We overcome this problem and generate binary VLADs from the feature descriptors extracted from the frames.

The evaluation of graph-based optimization vs. filter-based approaches for structure from motion has been rigorously analyzed in [17]. The conclusion is that optimization approaches outperform filtering in terms of accuracy per unit of computing time. Visual SLAM improves accuracy by increasing the number of features while having minor effects when increasing the number of keyframes. However, the analysis was done with a standard Extended Kalman filter (EKF) formulation, which models the features in the vector state, and the computational time grows quadratically with the number of observations. Other approaches such as the Multi-State Constrained Kalman Filter (MSCKF) [18] express the constraints without including the 3D feature position in the filter state vector, resulting in computational complexity linear in the number of features. The work in [19] showed that MSCKF competes with graph-based optimization approaches in terms of accuracy and computational cost. We focus our method on a filter-based approach using the MSCKF formulation of the JPL xVIO library, targeting onboard rotorcraft flight applications. We extend the xVIO back-end to fuse multi-agent data, developing a collaborative formulation of the MSCKF. We use the Covariance Intersection (CI) algorithm [20] in order to avoid the filter being overconfident in the resulting updates. The CI overestimates the resulting covariance after fusing the measurements coming from several agents, ensuring filter consistency. Our back-end is similar to the research in [21] and [22].

[2]https://github.com/jpl-x/x

	Thermal	Collaborative	Decentralized	Data-Efficient	Remarks
Schmuck et al. [9], [10]	✗	✓	✗	✗	Centralized with loop closure
Cieslewski et al. [15]	✗	✓	✓	✓	Graph-based NetVLAD
Zhu et al. [21], [22]	✗	✓	✓	✗	SLAM and Kalman features
Delaune et al. [23]	✓	✗	✗	✗	Thermal intertial odometry
This work	✓	✓	✓	✓	Filter-based Thermal VLAD

TABLE I: Overview of the strengths of state-of-the-art visual-inertial odometry frameworks.

Thermal-inertial solutions have recently received considerable attention for odometry systems in challenging scenarios. The technology was not widely used due to its cost and poor resolution. However, recent advances in thermal imaging have made thermal cameras more affordable, with a smaller footprint, higher quality and modest power consumption enabling the portability to robotics applications [7]. The work by Delaune et al. [23], and Khattak et al. [24], have shown the use of raw thermal data to navigate in dark scenarios where conventional visual cameras fail. Despite the outstanding results obtained by these works, thermal-infrared cameras suffer from photometric inconsistencies due to the thermography generation process. As a result, a standard computer vision program that relies on the photometric consistency assumption might fail to deliver reliable perception data in a sequence of frames. In this direction, the work by Das et al. [25], develops a photometric model of Thermal Infrared (TIR) cameras to attenuate such disturbances, obtaining photometric consistency over time and space. However, place recognition with thermal imaging remains an open challenge, and its application in collaborative systems has not been investigated.

Tab. I summarizes the related work on collaborative visual odometry systems. Our proposal is a fully decentralized thermal inertial odometry (TIO) that uses place recognition with thermal imaging and benefits from a data-efficient collaborative strategy. To the best of our knowledge, this path has not been explored.

III. Methodology

We have a set $\mathbb{U}$ of U Unmanned Aerial Vehicles (UAVs), equipped with a monocular thermal-infrared camera and IMU. Each drone can fly independently by running the JPL xVIO [8]. At start-up, each drone is initialized with the same inertial reference frame. The system extracts Features from accelerated segment test (FAST) corners [26] and uses the pyramidal implementation of the Lucas-Kanade tracker [27] to build tracks out of the extracted features. We associate an

Oriented FAST and Rotated BRIEF (ORB) descriptor [28] to each track to perform track matching between the current UAV tracks and the ones received from the other agents. In xVIO there are three kinds of tracks:

- **Opportunistic**: concatenation of the Lucas-Kanade tracker results over a sequence of frames.
- **SLAM**: tracks corresponding to the feature points saved in the state vector. For these points we know the inverse depth parameterization.
- **MSCKF**: opportunistic tracks that have a size between two and M frames and fulfill the requirements to perform an MSCKF update, that is the baseline between two features in the track is higher than a certain threshold.

An MSCKF track contains a wide baseline between the tracked points, hence a match between opportunistic tracks of two different UAVs fulfill the requirements to define a single MSCKF track. For this reason, hereafter, we will talk only about MSCKF and SLAM tracks. Figure 2 depicts a diagram of our method implemented in the xVIO library. The front-end provides feature point descriptors and generates a VLAD descriptor contained in the request message. The Track Manager unit receives the matches, stores them and forms tracks. The tracks are employed to perform MSCKF and SLAM updates: when a message is received, the back-end finds correspondences between the current tracks and the received ones. The matches generated from the correspondences are then processed as SLAM-SLAM matches updates, or stored and processed afterward as collaborative MSCKF updates.

Details of the the EKF updates, the state and covariance propagation, the thermal front-end and the communication pipeline are described in the following.

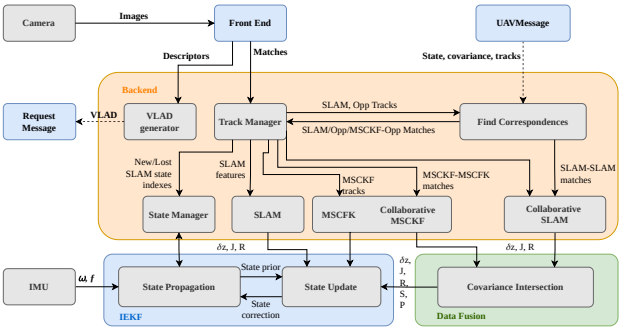

Fig. 2: Block diagram of our proposed method. Our solution can receive messages from the other agents and find SLAM-SLAM and MSKCF-MSCKF matches between the received data and the local tracks.

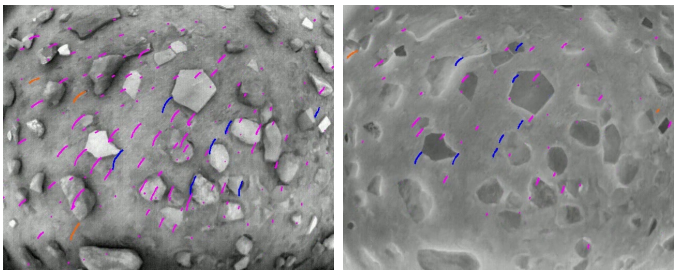

Fig. 3: Tracks representation in (a) calibrated and (b) uncalibrated thermal image. Blue SLAM tracks, orange MSCKF tracks, purple Opportunistic tracks.

A. Thermal Front-end

Thermal images show low contrast and bring photometric inconsistency over a sequence of frames. To deal with these problems, we employed the work of Das et al. [25] that attenuates the low-frequency non-uniformness and photometric inconsistency. Figure 4 shows that calibrated images return a higher Harris cornerness response than the uncalibrated ones, and hence make it possible to detect and track stronger corners. FAST corner detector compares the intensity value of a central pixel with a ring of pixels around it. A corner is detected if the intensity of N contiguous pixels in the ring is higher or lower than the central one by a threshold t. With calibrated thermal frames, the gradient of the image is high, and hence the value of t can be set to detect the most distinctive corners. We counted the number of tracks and evaluated the feature life for both calibrated and uncalibrated frames. As shown in Figure 3, applying spatial and temporal parameters calibration results in approximately four times the number of FAST corners being successfully detected and tracked with the Lucas-Kanade tracker than on uncalibrated images, with average feature track length on calibrated images also being increased. Namely, it is harder to lose the tracked corners since the image has higher contrast than the original thermal frame. This result implies that since four times more tracks are detected, we can potentially have four times more track correspondences between the robots, hence more collaborative updates for the EKF.

B. State definition

In the xVIO back-end, the state vector x (1), can be divided into the states concerning the IMU x_I and the visual part x_V.

The first term, $x_I \in \mathbb{R}^{16}$, defined in (2) includes the position, velocity and orientation quaternion of the IMU frame $\{i\}$ with respect to the world frame $\{w\}$, the gyroscope and the accelerometer biases, b_g and b_a respectively.

The second term, $x_V \in \mathbb{R}^{7M+3N}$, defined in (3) is split into the sliding window x_S, that has the positions $\{p_w^{c_i}\}_i$ and the orientations $\{q_w^{c_i}\}_i$ of the camera frame of the last M image time instances, and the feature states x_F (4), that contains the inverse depth parametrization of the landmarks f_i with $i = 1,...,N$.

$$x = \begin{bmatrix} x_I{}^T & x_V{}^T \end{bmatrix}^T \tag{1}$$

$$x_I = \begin{bmatrix} p_w^i{}^T & v_w^i{}^T & q_w^i{}^T & b_g{}^T & b_a{}^T \end{bmatrix}^T \tag{2}$$

$$x_S = \begin{bmatrix} p_w^{c_1}{}^T & \dots & p_w^{c_M}{}^T & q_w^{c_1}{}^T & \dots & q_w^{c_M}{}^T \end{bmatrix}^T \tag{3}$$

$$x_F = \begin{bmatrix} f_1{}^T & \dots & f_N{}^T \end{bmatrix}^T \tag{4}$$

The error state vector δx can also be divided as the state vector. The error state is defined for the position vectors as the difference between the true and estimated position,

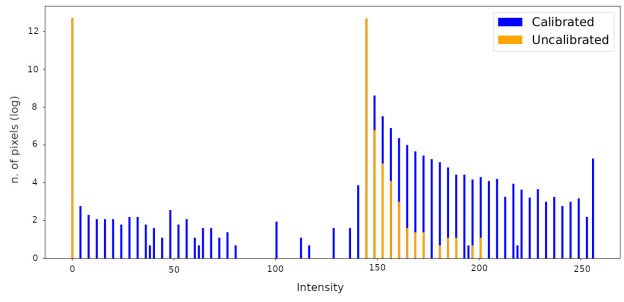

Fig. 4: The histogram of the corner response of the calibrated and uncalibrated data, shows that the calibrated images have a higher Harris cornerness response.

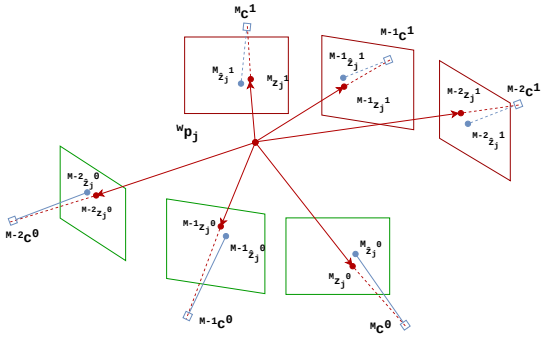

Fig. 5: Collaborative MSCKF representation. ${}^{i}c^{0}$ and ${}^{i}c^{1}$ are the cameras of two different agents 0 (green) and 1 (red) respectively, that are tracking the same landmark ${}^{w}p_j$.

$p^i_w - \hat{p}^i_w = \delta p^i_w \in \mathbb{R}^3$, and similarly for the velocity and inertial bias errors. Differently, for the quaternions, the error vector is given by $q^i_w = \hat{q}^i_w \otimes \delta q^i_w \in \mathbb{R}^4$, where $\otimes$ is the quaternion product. For a minimal representation of the error quaternion, we use the small angle approximation, namely $\delta q \simeq \begin{bmatrix} 1 & \frac{1}{2}\delta\theta^T \end{bmatrix}^T$, hence $\delta\theta \in \mathbb{R}^3$ represents the quaternion error in a lower dimension space.

The measurement model of 3D landmark ${}^{w}p_j$ in the world frame $\{w\}$, observed by the camera $\{c_i\}$, is the normalized projection on the unit focal length camera plane:

$$ {}^{i}z_j = \frac{1}{{}^{c_i}z_j} \begin{bmatrix} {}^{c_i}x_j \\ {}^{c_i}y_j \end{bmatrix} + {}^{i}n_j \tag{5} $$

where

$$ {}^{c_i}p_j = \begin{bmatrix} {}^{c_i}x_j & {}^{c_i}y_j & {}^{c_i}z_j \end{bmatrix}^T \tag{6} $$

$$ = C(q^{c_i}_w)({}^{w}p_j - p^{c_i}_w) \tag{7} $$

and ${}^{i}n_j$ is a zero-mean white Gaussian measurement noise with covariance matrix ${}^{i}R_j = \sigma^2_V I_2$. We assume the standard deviation σ_V is uniform over the image and it depends on the performance of the visual front-end. $C(q^b_a)$ is the rotation matrix associated with the quaternion q^b_a such that ${}^{b}x = C(q^b_a){}^{a}x$.

C. MSCKF-MSCKF update

We used the same approach proposed by Zhu [21]. Here we want to give a detailed overview of the method, and how this is applied to the *xVIO* stack.

The *xVIO* back-end performs MSCKF [18] updates. MSCFK minimizes the residual between tracked feature points and the back-projection on the camera plane of the triangulated landmark. Figure 5 represents how the MSCKF residuals can be interpreted in a collaborative setup. The tracked landmark ${}^{w}p_j$ is triangulated by using all the measurements of robots' cameras, ${}^{i}c^{0}$ and ${}^{i}c^{1}$. Then the resulting 3D point is projected into the camera planes that observed the feature.

We express the measurement model ${}^{i}z^l_j \in \mathbb{R}^2$ of the observed feature ${}^{i}\hat{z}^l_j$ in the camera frame $i \in \mathbb{I} \subseteq \{1,...,M\}$ of the robot $l \in \mathbb{U}$ using the model in (7) as a nonlinear function of the state x and the landmark ${}^{w}p_j$ (8), such that the measurement innovation ${}^{i}\delta z^l_j = {}^{i}z^l_j - {}^{i}\hat{z}^l_j$ can be linearized as in (9).

$$ {}^{i}z^l_j = h(x^l, {}^{w}p_j) + {}^{i}n^l_j \tag{8} $$

$$ {}^{i}\delta z^l_j \simeq {}^{(i,j)}H^l_x \delta x^l + {}^{(i,j)}H^l_p \, {}^{w}\delta p_j + {}^{i}n^l_j \tag{9} $$

By stacking together the residuals for all the observations of the landmark ${}^{w}\delta p_j$ for the robot l we write:

$$ \delta z^l_j \simeq {}^{j}H^l_x \delta x^l + {}^{j}H^l_p \, {}^{w}\delta p_j + n^l_j \tag{10} $$

Since ${}^{w}\delta p_j$ is neither part of the state, nor can be included in the noise, we need to get rid of it to perform an EKF update. With this purpose, we multiply on each side by the left nullspace A_j of ${}^{j}H^l_p$. By doing so we split the system into two subsystems, one $\delta z^l_{0_j}$ that is a function of δx^l only and one $\delta \bar{z}^l_{0_j}$ that also depends on ${}^{w}\delta p_j$ (11).

$$ \begin{bmatrix} \delta z^l_{0_j} \\ \delta \bar{z}^l_{0_j} \end{bmatrix} = \begin{bmatrix} {}^{j}H^l_{0_x} \\ {}^{j}\bar{H}^l_{0_x} \end{bmatrix} \delta x^l + \begin{bmatrix} {}^{j}H^l_{0_p} \\ 0 \end{bmatrix} {}^{w}\delta p_j + \begin{bmatrix} n^l_j \\ \bar{n}^l_j \end{bmatrix} \tag{11} $$

The equations (9)-(11), describe the standard procedure to perform an MSCKF update for the agent l. In particular in a single UAV update, we feed the EKF with the bottom part of the system (11). To perform the collaborative MSCKF update, we need to write a residual that depends on the states of all the agents that observed the same landmark ${}^{w}p_j$. Writing such a residual is possible by stacking together all the $\delta z^l_{0_j}$ in (11) with $l \in \mathbb{L}$ where $\mathbb{L} = \{l_0, ...l_L\}$ is the set of the L robots that observe ${}^{w}p_j$ (12).

$$ \begin{bmatrix} \delta \bar{z}^{l_0}_{0_j} \\ \vdots \\ \delta \bar{z}^{l_L}_{0_j} \end{bmatrix} = D\left(\begin{bmatrix} {}^{j}H^{l_0}_{0_x} \\ \vdots \\ {}^{j}H^{l_L}_{0_x} \end{bmatrix} \right) \begin{bmatrix} \delta x^{l_0} \\ \vdots \\ \delta x^{l_L} \end{bmatrix} + \begin{bmatrix} {}^{j}H^{l_0}_{0_p} \\ \vdots \\ {}^{j}H^{l_L}_{0_p} \end{bmatrix} {}^{w}\delta p_j + \begin{bmatrix} n^{l_0}_j \\ \vdots \\ n^{l_L}_j \end{bmatrix} $$

$$ \Rightarrow \delta \bar{z}_{0_j} = {}^{j}H_{0_x} \delta x + {}^{j}H_{0_p} \, {}^{w}\delta p_j + n_j \tag{12} $$

Where $D(\cdot)$ represents the matrix having a diagonal that corresponds to the elements of the input vector.

To perform an EKF measurement update with the resulting equation (12), we need to multiply on each side by the left nullspace A_j of ${}^{j}H_p$ and by doing so obtain two subsystems as in (11). Finally, we take the part that depends on the state only,

$$ \delta \bar{z}_j = {}^{j}H_x \delta x + n_j \tag{13} $$

and perform the CI-EKF (19)-(22) update to take into account the possible correlations between the agents states.

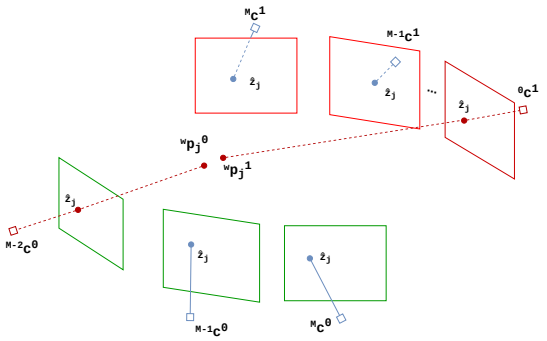

Fig. 6: Collaborative SLAM representation. ${}^{M-2}c^0$ and ${}^0c^1$ are the anchor poses of two different agents 0 (green) and 1 (red) respectively. These anchors observed the landmark ${}^w p_j$ first with respect to the other frames, ${}^w p_j$ is parametrized according to ${}^{M-2}c^0$ for UAV 0 and to ${}^0c^1$ for UAV 1.

The Jacobian ${}^j H_x$ can be rewritten according to the state the various columns refer to as:

$$\delta \bar{z}_j = \begin{bmatrix} H^{l_0} & \dots & H^{l_L} \end{bmatrix} \delta x + n_j \tag{14}$$

D. SLAM-SLAM update

To perform the collaborative update between two SLAM features, we adopted the approach proposed by Zhu [22] for its efficiency. The method constrains the EKF, that is, when two SLAM features of two different agents 0 and 1 match, it implies that they refer to the same 3D point in the world frame and so the difference between the landmarks in the world frame must be zero:

$${}^w p_j{}^0 - {}^w p_j{}^1 = 0 \tag{15}$$

as Figure 6 shows.

In *xVIO* the SLAM landmarks are stored in the EKF state in terms of an inverse depth parametrization. The parametrization is calculated with respect to the anchor pose ${}^w p^l_{a_j}$, also part of the state, that is the pose of the camera frame that observed the feature point first. Hence, we can write (15) in terms of the states of the two robots:

$${}^w p^0_{a_j} + \frac{1}{\rho^0_j} C({}^w p^0_{a_j})^T \begin{bmatrix} \alpha^0_j \\ \beta^0_j \\ 1 \end{bmatrix} - {}^w p^1_{a_j} - \frac{1}{\rho^1_j} C({}^w p^1_{a_j})^T \begin{bmatrix} \alpha^1_j \\ \beta^1_j \\ 1 \end{bmatrix} = 0 \tag{16}$$

where α^l_j, β^l_j, and ρ^l_j are the inverse depth parametrization of the landmark ${}^w p_j{}^l$ observed by the drone l. Linearizing (16), we can write the residual obtaining a model that depends on the states of the two UAVs involved in the update as in (14), that is then used to perform the CI-EKF (19)-(22) update.

E. EKF prediction step

The prediction step is the same as in *xVIO*, that is the IMU measurements are propagated using the inertial integration model described in [29]. For sake of completeness we report the IMU measurement model but we refer the reader to Delaune et al. [8] work for more details:

$$\omega_{IMU} = {}^i\omega^i_w + b_g + n_g \tag{17}$$

$$a_{IMU} = C(q^i_w)(a^i_w - {}^w g) + b_a + n_a \tag{18}$$

where n_g and n_a are zero-mean Gaussian white noises, the biases b_g and b_a are modeled as random walk with zero-mean Gaussian white noise. The gravity vector g is considered constant, the Coriolis forces and the planet curvature are neglected.

F. EKF measurements update step

To fuse the data coming from different agents, we should keep track of the cross-correlation between the various states. However, doing so is expensive in terms of memory since the number of elements in each drone's covariance will increase quadratically with the number of agents sharing data. The Covariance-Intersection (CI) algorithm [20] allows us to neglect the correlation information between the robots, and perform data fusion with consistent covariance information. Namely, the resulting covariance out of the CI algorithm is larger than the real estimate one could obtain considering the cross-correlation. By doing so, the EKF is not overconfident, and it is kept consistent. The CI algorithm consists of a convex minimization problem that finds the best weights ω_l to scale the covariances of each agent $l \in \{l_0, \dots l_L\}$ involved in the update to obtain a covariance with the minimum eigen values. We use the CI-EKF to update the drone state in the same form Zhu proposed, where l_0 denotes the drone where the update is performed:

$$S = R + \sum_{l \in \{l_0, \dots l_L\}} \frac{1}{\omega_l} H^{l^T} P^l_{k-1} H^l \tag{19}$$

$$K = \frac{1}{\omega_{l_0}} P^{l_0}_{k-1} H^{l_0{}^T} S^{-1} \tag{20}$$

$$P^{l_0}_k = (I_{\omega_{l_0}} - K H^{l_0{}^T}) \frac{1}{\omega_{l_0}} P^{l_0}_{k-1} \tag{21}$$

$$\delta x^{l_0} = K \delta z^{l_0} \tag{22}$$

Where R is the covariance matrix of the Gaussian noise described in Section III-B. P^l_{k-1} is the covariance term of the drone l, whereas $P^{l_0}_k$ is the new covariance of the agent that is performing the update. Notice that $I_{\omega_{l_0}}$ is a diagonal matrix where each element $a_i^{\omega_{l_0}}$ of the diagonal is defined as:

$$a_i^{\omega_{l_0}} = \begin{cases} 1 & \text{if } i \in \mathbb{J} \\ \omega_{l_0} & \text{otherwise} \end{cases} \tag{23}$$

$\mathbb{J}$ is the set of ids in the state vector x that are directly involved in the update (i.e., the parts of the state that will be updated). This formulation is crucial to perform the CI update without making the EKF underconfident on the part of the state not involved in the collaborative update.

Before propagating the state and the covariance to remove outliers, the innovation of each collaborative update undergoes a χ^2 test with 95% confidence. We compute the Mahalanobis distance with the real covariance of each agent and not with the one scaled by the CI algorithm. The reasons behind this choice are twofold:

- Being conservative: the original covariance is smaller than the scaled one and so the resulting distance is larger, hance is more difficult to pass the χ^2 test.
- Avoid unuseful computation: we compute the CI covariance only if the test is passed.

G. Communication Pipeline

To perform the collaborative MSCKF and SLAM updates, the agents need the tracks, the state, and the covariance of the other robots. We define a message called *MessageUAV* which contains all the information needed to perform the collaborative updates. A naive approach to share data between agents would be one in which each robot sends a *MessageUAV* to all the others at the camera frame rate. The result would be an extremely high usage of the communication channel considering that an estimate of the message size is around $190.7kB$. We computed this estimate by averaging the weights of the *MessageUAV* sent by the robots in simulation.

To avoid this significant usage of the bandwidth, we developed a keyframe-based request/response data exchange system to lower the data sent over the network. To perform the collaborative updates, the agents need to find correspondences between the features they are tracking and the tracks they receive. Hence, this is the same as performing place recognition and then loop-closure among a series of keyframes.

Each drone sends to all the others a message named *RequestUAV* at the camera frame rate. This message contains a Binary VLAD descriptor of the frame the agent is viewing. We use a modified version of the BVLAD [30], to perform place recognition between the UAVs. The ORB descriptor is binary and so we create a VLAD by using binary operations only, in particular, we use the OR ($\vee$) and the AND ($\wedge$) operations, instead of sums and subtractions. Doing so we do not need to perform normalization nor exponential decay law to reduce the impact of recurring features. VLADs are of a fixed length, and they can be shortened applying dimensionality reduction through PCA [31]. The resulting request message has a fixed size that is much smaller than a *MessageUAV*. We created a coarse vocabulary of ORB visual words of 64 centroids. Hence, without applying any dimensionality reduction sharing a VLAD is the same as sharing 64 ORB descriptors. The result is that the *RequestUAV* message weighs $2.05kB$, including the timestamp and the id of the sender, resulting in lower data consumption. When a robot receives the request message from another agent, it looks among its keyframes to find a loop-closure. A keyframe contains the information needed by the *MessageUAV* and a VLAD. The score between the received descriptor r_j and the keyframe one r_k is given by:

$$s \doteq \frac{Hamming(r_j \wedge r_k)}{D_{max}} \tag{24}$$

Where D_{max} is the maximum distance that can be computed between two descriptors and $Hamming(\cdot)$ is the Hamming distance of the resulting vector. If the score is over a certain threshold, the keyframe is wrapped into a message and shipped to the agent who sent the request.

Each agent maintains a database of keyframes. To populate the database, we follow the same approach proposed for PTAM [32]. Namely, every time the ratio between the baseline b, that is the distance between the previous keyframe and the current camera pose, and the average depth $\bar{z}$ of the tracked features is higher than a threshold that is usually around $10-20\%$, a new keyframe is defined.

Dataset	Agent	Collaborative **Ours**	xVIO [8] baseline	Improvement
Castle Parallel	UAV0	**0.4312±0.2632**	0.6183±0.4136	30.26%
	UAV1	**0.3871±0.1860**	0.4817±0.3039	19.64%
	UAV2	0.6108±0.3730	**0.5169±0.3153**	-18.17%
Castle Around	UAV0	**0.3437±0.1678**	0.3788±0.1973	9.26%
	UAV1	**0.3663±0.1531**	0.7186±0.3547	49.02%
	UAV2	**0.5670±0.2851**	0.5995±0.3276	5.42%
	UAV3	0.8717±0.4094	**0.7240±0.3524**	-20.40%

TABLE II: Absolute trajectory error and standard deviation for the visual datasets.

Dataset	Agent	Collaborative Calibrated **Ours**	Collaborative **Ours**	Independent Calibrated **Ours**	xVIO [8] Baseline
Mars Yard	UAV0	**0.1610±0.0731**	0.1860±0.0928	0.1906±0.0882	0.1835±0.0926
	UAV1	**0.0438±0.0189**	0.0812±0.0381	0.0621±0.0248	0.0813±0.0382

TABLE III: Absolute trajectory error and standard deviation for thermal Mars Yard dataset.

IV. Experiments

Figure 8 shows the 3D representation of the environments we recorded the data for testing our system. To validate our system, we used real thermal recordings from the JPL Mars Yard. The data was recorded from the industrial version of the FLIR Boson camera with a lens providing a 95-deg field of view (FOV). It generates 640 x 512 images in the $7.5\mu m$ - $13.5\mu m$ longwave infrared spectral range, with a thermal sensitivity under $40mK$ and an 8-ms time constant. The camera was calibrated using as a target a circle grid pattern laminated over a Gatorfoam board left for 5 minutes in the sun, then the Kalibr toolbox [33] was used to find the calibration parameters. The IMU used was the ICM20608 on-board the Pixhawk mini. This dataset is the same as the collection used for the work by Delaune et al. [23]. We merged two Robot Operating System (ROS) bag files where two UAVs fly two squared trajectories with different altitudes, over the JPL Mars Yard for almost $10m$, Figure 7 illustrates the two drones trajectories with their estimates. The data is collected with a five-minute distance in time. Therefore, we can rely on the assumption that the thermal fingerprint of the objects does not drastically change over time. The results of our system compared against the baseline that is the standard xVIO implementation are reported in Table III. We used the RPG trajectory evaluator tool [34] to compute the Absolute Trajectory Error between the ground-truth and the estimates. The collaborative setup with photometric calibrated thermal images outperforms the standard xVIO by $12.26\,\%$ and $46.12\,\%$.

We also created two synthetic datasets with visual data to

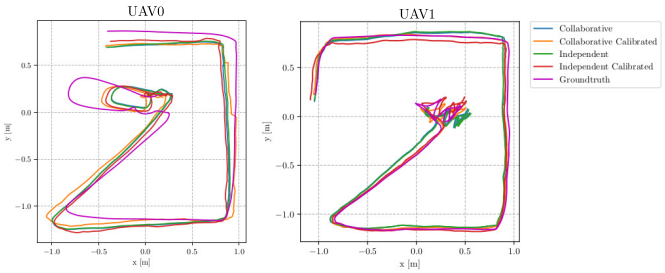

Fig. 7: Trajectories estimate results for UAV0 and UAV1 in the Mars Yard dataset.

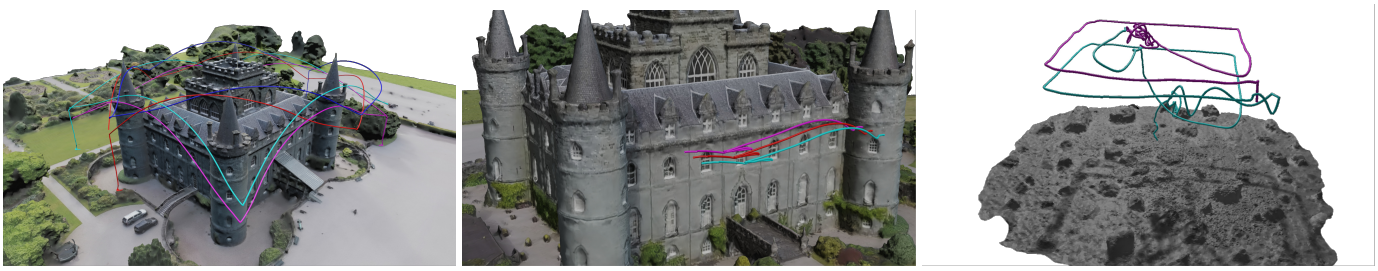

Fig. 8: Dataset scenarios (a) "Castle Around" four drones fly around the Inveraray Castle model (https://skfb.ly/6z7Rr) by Andrea Spognetta licensed under Creative Commons Attribution-NonCommercial. (b) "Castle Parallel" three drones fly parallel trajectories in front of the castle. (c) Two drones fly squared trajectories on the Mars Yard at JPL.

Dataset	**Input**	Naive [kB]	(**Ours**) [kB]	Improvement
Castle Around	Message UAV	639.60	**264.30**	57.60%
	Request UAV	-	**6.89**	
Castle Parallel	Message UAV	152.56	**141.12**	6.42%
	Request UAV	-	**1.64**	
Mars Yard	Message UAV	234.37	**22.88**	89.17%
	Request UAV	-	**2.51**	

TABLE IV: A *MessageUAV* object weighs 190.7 kB while a request message occupies 2.05 kB. The data reported by the table are estimated according to the number of messages the agents exchange in simulation and real data scenarios. The camera frame rate for all the datasets is 30Hz. The length in seconds for each dataset is 30s, 9s, 20s for the **Castle Around**, **Castle Parallel**, **Mars Yard** respectively.

Algorithm	**Agent**	**N. of Messages sent**	**Max CPU [%]**	**Max Mem [MiB]**	**ATE [m]**
COVINS [10]	UAV0	4587	208.47	710.8	**0.1641±0.0624**
	UAV1	4524	211.28	643.5	**0.3626±0.1214**
	UAV2	4457	204.87	681.1	**0.2058±0.0614**
	UAV3	4788	201.18	701.8	**0.2897±0.1132**
	Server	-	325.78	710.9	-
Ours	UAV0	**1482**	**165.66**	**133.9**	0.3437±0.1678
	UAV1	**1414**	**158.56**	**124.8**	0.3663±0.1531
	UAV2	**1475**	**154.18**	**139.1**	0.5670±0.2851
	UAV3	**1465**	**170.79**	**136.2**	0.8717±0.4094

TABLE V: Comparison between COVINS [10] and our approach.

stress different aspects of our method. We modified the VI-Sensor Simulator [35] to generate ground-truth visual, inertial and landmark data to test the back-end of the system. The drones simulated by the VI-Sensor Simulator mount a visual camera that generates 752 x 480 images with a focal length of 455 px at 30 Hz. The IMU sensor simulated is the ADIS16448 that generates data at 200 Hz. The datasets *Castle Around* and *Castle Parallel* are recorded using a 3D reconstruction of the Scottish Inveraray Castle. The former simulates four UAVs flying 220m squared trajectories around the castle with different altitudes and rotations. The latter consists of 3 drones flying 30m parallel trajectories at the same altitude which stresses the communication pipeline. Table II shows the results using the collaborative approach.

Table IV reports the different data usage between the proposed communication pipeline and the naive approach in the different datasets. Notably, our approach shows a decrease of data exchanged by 89 % in one of the scenarios. In the Castle Parallel the improvement is lower due to the trajectory design, where the drones observe the same scene and loop closure

VLAD Generation	Feature Matching	Loop closure	Photometric calibration	Tracker	MultiSLAM	MultiMSCKF & MSCKF	**Tot**
0.08	7.19	3.03	22.53	13.45	2.46	1.15 + 3.18	53.07
0%	13.55%	5.71%	42.44%	25.34%	4.63%	8.16%	100%

TABLE VI: In the first row the average time needed by each module expressed in [ms]. The second row shows the computational time percentage occupied by each block.

is performed all along the path. To simulate the different drones we used containerization, where each drone is a Docker container and runs independently from the others. The docker base image we used to run the collaborative setup is a modified version of the *dt-base-environment* of the Duckietown platform [36].

Finally, we also compared our system with visual data against the state-of-the-art Centralized Cooperative system COVINS [10]. The results, reported in Table V, show that the centralized method outperforms our system in terms of accuracy of the trajectory estimation in almost all of the agents. This result was expected since the COVINS system performs loop closure detection and global pose and map optimization. In contrast, our filter-based approach shows lower memory and CPU usage as well as a much lower number of messages exchanged in the network compared to COVINS.

For completeness, we evaluated the time performance of our methods. Table VI reports the average time in milliseconds needed by each block of the proposed system to process the received information. The total time needed to update the agent's state is about 53ms. Hence our system can perform a collaborative state estimate with a frequency of 20Hz. The implemented pipeline leaves a margin of improvement considering more efficient multi-threading approaches.

V. Conclusion

With this work, we extended the JPL xVIO library with a decentralized multi UAV system that can work with thermal and visual data. To the best of our knowledge, the described project is the first work that implements a collaborative Thermal-Inertial Odometry system using a photometric calibration algorithm that improves feature matching for loop closure detection. We developed a multi-threading communication pipeline that reduces the bandwidth compared to a naive approach and maintains system scalability. Our Code and associated datasets are open-source.

ACKNOWLEDGMENTS

The research was funded by the Combat Capabilities Development Command Soldier Center and Army Research Laboratory. This research was carried out at the Jet Propulsion Laboratory, California Institute of Technology, and was sponsored by the JPL Visiting Student Research Program (JVSRP) and the National Aeronautics and Space Administration (80NM0018D0004).

REFERENCES

[1] J. Schoolcraft, A. T. Klesh, and T. Werne, *MarCO: Interplanetary Mission Development On a CubeSat Scale*, 05 2016, vol. 2491. [Online]. Available: https://arc.aiaa.org/doi/abs/10.2514/6.2016-2491

[2] A. Pirrami and H. Smith, "Cooperative autonomous distributed robotic explorers (cadre)," 2021. [Online]. Available: https://www.nasa.gov/directorates/spacetech/game_changing_development/projects/CADRE

[3] Z. Zhang and D. Scaramuzza, "Visual-inertial odometry of aerial robots," *Encyclopedia of Robotics, Springer*, 2019.

[4] G. Huang, "Visual-inertial navigation: A concise review," in *IEEE Int. Conf. Robot. Autom. (ICRA)*, 2019.

[5] D. S. Bayard, D. T. Conway, R. Brockers, J. H. Delaune, L. H. Matthies, H. F. Grip, G. B. Merewether, T. L. Brown, and A. M. San Martin, "Vision-based navigation for the NASA Mars helicopter," in *AIAA Scitech 2019 Forum*, 2019, p. 1411.

[6] E. R. Boroson, R. Hewitt, N. Ayanian, and J.-P. de la Croix, "Inter-robot range measurements in pose graph optimization," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 4806–4813.

[7] R. Gade and T. Moeslund, "Thermal cameras and applications: A survey," *Machine Vision and Applications*, vol. 25, pp. 245–262, 01 2014.

[8] J. Delaune, D. S. Bayard, and R. Brockers, "xVIO: A Range-Visual-Inertial Odometry Framework," *arXiv e-prints*, p. arXiv:2010.06677, Oct. 2020.

[9] P. Schmuck and M. Chli, "Multi-UAV collaborative monocular SLAM," in *IEEE Int. Conf. Robot. Autom. (ICRA)*, 2017, pp. 3863–3870.

[10] P. Schmuck, T. Ziegler, M. Karrer, J. Perraudin, and M. Chli, "Covins: Visual-inertial slam for centralized collaboration," in *2021 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*, 2021, pp. 171–176.

[11] E. Nettleton, S. Thrun, H. Durrant-Whyte, and S. Sukkarieh, "Decentralised slam with low-bandwidth communication for teams of vehicles," vol. 24, 01 2003, pp. 179–188.

[12] A. Gautam and S. Mohan, "A review of research in multi-robot systems," in *2012 IEEE 7th International Conference on Industrial and Information Systems (ICIIS)*, 2012, pp. 1–5.

[13] J. Dong, E. Nelson, V. Indelman, N. Michael, and F. Dellaert, "Distributed real-time cooperative localization and mapping using an uncertainty-aware expectation maximization approach," in *IEEE Int. Conf. Robot. Autom. (ICRA)*, 2015, pp. 5807–5814.

[14] R. Arandjelović, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, "NetVLAD: CNN architecture for weakly supervised place recognition," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2016, pp. 5297–5307.

[15] T. Cieslewski, S. Choudhary, and D. Scaramuzza, "Data-efficient decentralized visual SLAM," *IEEE Int. Conf. Robot. Autom. (ICRA)*, 2018.

[16] R. Arandjelović, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, "NetVLAD: CNN architecture for weakly supervised place recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[17] H. Strasdat, J. Montiel, and A. Davison, "Visual SLAM: Why filter?" *Image Vis. Comput.*, 2012.

[18] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint Kalman filter for vision-aided inertial navigation," in *IEEE Int. Conf. Robot. Autom. (ICRA)*, 2007, pp. 3565–3572.

[19] M. Li and A. I. Mourikis, "High-precision, consistent EKF-based visual-inertial odometry," *Int. J. Robot. Research*, vol. 32, no. 6, pp. 690–711, 2013.

[20] S. Julier and J. Uhlmann, "A non-divergent estimation algorithm in the presence of unknown correlations," in *Proceedings of the 1997 American Control Conference (Cat. No.97CH36041)*, vol. 4, 1997, pp. 2369–2373 vol.4.

[21] P. Zhu, Y. Yang, W. Ren, and G. Huang, "Cooperative visual-inertial odometry," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 13 135–13 141.

[22] P. Zhu, P. Geneva, W. Ren, and G. Huang, "Distributed visual-inertial cooperative localization," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021, pp. 8714–8721.

[23] J. Delaune, R. Hewitt, L. Lytle, C. Sorice, R. Thakker, and L. Matthies, "Thermal-inertial odometry for autonomous flight throughout the night," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 1122–1128.

[24] S. Khattak, C. Papachristos, and K. Alexis, "Keyframe-based direct thermal–inertial odometry," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 3563–3569.

[25] M. P. Das, L. Matthies, and S. Daftry, "Online photometric calibration of automatic gain thermal infrared cameras," p. 2453–2460, Apr 2021. [Online]. Available: http://dx.doi.org/10.1109/LRA.2021.3061401

[26] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in *Computer Vision – ECCV 2006*, A. Leonardis, H. Bischof, and A. Pinz, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 430–443.

[27] J.-Y. Bouguet *et al.*, "Pyramidal implementation of the affine lucas kanade feature tracker description of the algorithm," *Intel corporation*, vol. 5, no. 1-10, p. 4, 2001.

[28] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in *2011 International Conference on Computer Vision*, 2011, pp. 2564–2571.

[29] S. Weiss and R. Siegwart, "Real-time metric state estimation for modular vision-inertial systems," in *2011 IEEE International Conference on Robotics and Automation*, 2011, pp. 4531–4537.

[30] D. Van Opdenbosch, G. Schroth, R. Huitl, S. Hilsenbeck, A. Garcea, and E. Steinbach, "Camera-based indoor positioning using scalable streaming of compressed binary image signatures," in *2014 IEEE International Conference on Image Processing (ICIP)*, 2014, pp. 2804–2808.

[31] R. Arandjelovic and A. Zisserman, "All about vlad," in *2013 IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 1578–1585.

[32] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," in *Proc. Sixth IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'07)*, Nara, Japan, November 2007.

[33] P. Furgale, J. Rehder, and R. Siegwart, "Unified temporal and spatial calibration for multi-sensor systems," in *IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)*, 2013.

[34] Z. Zhang and D. Scaramuzza, "A tutorial on quantitative trajectory evaluation for visual(-inertial) odometry," in *IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)*, 2018.

[35] L. Teixeira, M. R. Oswald, M. Pollefeys, and M. Chli, "Aerial Single-View Depth Completion with Image-Guided Uncertainty Estimation," *IEEE Robotics and Automation Letters (RA-L)*, 2020.

[36] L. Paull, J. Tani, H. Ahn, J. Alonso-Mora, L. Carlone, M. Cap, Y. F. Chen, C. Choi, J. Dusek, Y. Fang, D. Hoehener, S.-Y. Liu, M. Novitzky, I. F. Okuyama, J. Pazis, G. Rosman, V. Varricchio, H.-C. Wang, D. Yershov, H. Zhao, M. Benjamin, C. Carr, M. Zuber, S. Karaman, E. Frazzoli, D. Del Vecchio, D. Rus, J. How, J. Leonard, and A. Censi, "Duckietown: An open, inexpensive and flexible platform for autonomy education and research," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 1497–1504.