2024 21st International Conference on Ubiquitous Robots (UR)
June 24 - 27, 2024, New York University, USA

# Multi-robot Cooperative Localization with Single UWB Error Correction

Kevin Christiansen Marsim, Junho Choi, Myeongwoo Jeong, Kihwan Ryoo, Jeewon Kim, Taehyun Kim, and Hyun Myung*

***Abstract*— Deployment of multiple robots in real-world scenarios requires simultaneous information exchange from all platforms to ensure effective task performance. The robot's relative position to its peers is an important data needed to predict collision between robots or task distribution. This paper introduces a robust and simple method for achieving cooperative localization among multiple robots, utilizing a single ultra-wideband (UWB) sensor for each platform. Each robot uses a visual-inertial odometry (VIO) system to track its own trajectory. Given the inherent drift associated with VIO systems, we leverage UWB data to estimate and correct this drift, enhancing each robot's localization accuracy. Our approach substantially improves the result compared with other cooperative localization methods and can even correct the VIO ego-motion.**

***Index Terms*— cooperative localization, multi-robot, sensor fusion, error-based optimization, ultra-wideband (UWB), drift correction.**

## I. Introduction

In recent years, cooperation in a multi-robot system has garnered significant attention among researchers. Implementing multi-robot systems enhances adaptability to diverse environments and expands the scope of autonomous tasks that can be performed [1]–[3].

Cooperative Localization is one key component that enables a multi-robot system to cooperate with each other [4]–[6]. Recent advancements have utilized multi-sensor fusion to estimate the relative pose between robots. Visual-inertial odometry (VIO) facilitates collaborative localization by analyzing shared visual scenes among robots [7]. On the other hand, LiDAR-inertial odometry (LIO) achieves this through the mutual recognition of the mapped environment via recorded point clouds [8].

Despite the effectiveness of these sensor fusion methods in providing reliable ego-motion estimates, they depend on loop-closing mechanisms for cooperative localization systems [9]. Without matching data between robots, it becomes difficult to determine their relative positions accurately. Furthermore, incorrect loop closures can lead to significant errors in recovering the relative pose, hindering the system's overall reliability.

*Corresponding author: Hyun Myung
All the authors are with the School of Electrical Engineering, KAIST (Korea Advanced Institute of Science and Technology), Daejeon, 34141, Republic of Korea. {kevinmarsim, cjh6685kr, wjdauddn1477, rkh137, ddarong2000, ktw1404, hmyung}@kaist.ac.kr

This work has been supported by the Unmanned Swarm CPS Research Laboratory program of Defense Acquisition Program Administration and Agency for Defense Development(UD220005VD).

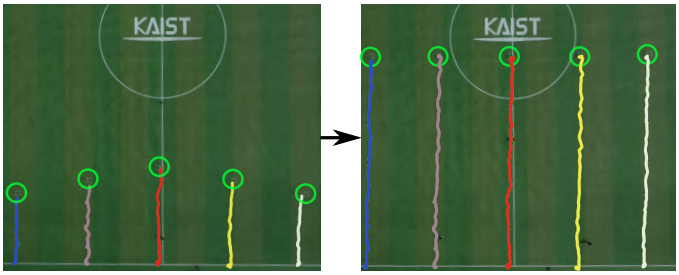


Fig. 1. Our algorithm effectively tracks five robots equipped with only low-cost sensors in an open field, eliminating the need for additional infrastructure such as UWB anchors for references in the global frame.

Ultra-wideband (UWB) sensors have recently been utilized to obtain relative distance from each robot. However, their deployment typically requires structural prior information via UWB anchors [10], [11]. These anchors facilitate the triangulation of each robot's location from known positions. Despite their utility, the application of UWB anchors is impractical in open or unknown environments due to installation issues.

In this paper, we propose a simple but effective cooperative localization algorithm to obtain an accurate relative pose between robots and their individually corrected trajectory. As shown in Fig. 1, our approach allows the localization of each robot in an open field without the need for UWB anchors. First, we employ VIO algorithm to track the motion of each robot. Since VIO algorithm will inevitably drift, we attempt to correct this error by utilizing the UWB data. Unlike other approaches, our strategy focuses on refining the VIO drift instead of directly optimizing the robot's state, thereby enhancing the correction of the VIO trajectory.

The main contributions of this paper are threefold:

1) We propose a real-time cooperative localization framework utilizing low-cost sensors to estimate the state of each robot simultaneously.
2) We propose an error-based optimization method to correct long-term VIO drift effectively.
3) We show our algorithm's proficiency in estimating the relative pose of robots within an anchorless environment.

## II. Related Work

### A. Vision-Based Cooperative Localization

Vision-based approaches rely on matching images recorded during deployment. The most prevalent technique for obtaining relative pose is the Perspective-n-Point method, which estimates the relative pose between cameras from

2024 21st International Conference on Ubiquitous Robots (UR) | 979-8-3503-6107-0/24/$31.00 ©2024 IEEE | DOI: 10.1109/UR61395.2024.10597444

matching features [12]. The first vision-based cooperative localization can be traced back to using landmark-based occupancy grid mapping for localizing in another robot's local frame [13]. Over the next few years, Zhu *et al.* employed a Kalman filter that utilizes shared common features to enhance relative localization accuracy [14]. CoVINS, proposed by Patel *et al.*, uses keyframe matching to determine the relative pose between robots [7]. CoVINS-G further improved the system by only requiring 2D matching keypoints to optimize multiple robot trajectories [15]. Loianno *et al.* integrated depth camera measurements to fuse multiple trajectories based on matching map points during deployment [16].

However, the success of vision-based cooperative localization depends on multiple robots encountering similar images, which may not always happen due to the different conditions in which they operate. Furthermore, changes in illumination can affect scene appearance, leading to unsuccessful image matching.

### B. LiDAR-Based Cooperative Localization

Similar to vision-based cooperative localization, LiDAR-based approaches require that robots traverse similar environments to determine their relative poses. Hess *et al.* employed a 2D LiDAR loop closure using branch-and-bound techniques to align local and global frame [17]. Xie *et al.* introduced RDC-SLAM, which allows for the sharing and matching map data between robots using feature-based and intensity-based descriptors [8]. Another method proposed by Fang *et al.* uses an iterated split covariance intersection filter to maintain error consistency in neighboring vehicles' LiDAR measurements [18]. Although LiDAR-based methods show potential, LiDAR cooperative localization is less developed than vision-based methods.

LiDAR-based localization can provide accurate estimations of relative pose, but high costs and computational demands hinder its usage. Each robot must be equipped with costly LiDAR and a powerful computer to process the point clouds. Furthermore, sharing maps in open environments without robust communication infrastructure remains challenging for multi-robot cooperative localization.

### C. UWB-Assisted Cooperative Localization

UWB offers straightforward one-dimensional ranging data between two sensors at a high rate. For relative localization, Choi *et al.* proposed a simple UWB-only method for estimating the relative pose of multiple unmanned aerial vehicles (UAV) [19]. Additionally, UWB-based localization is often paired with visual [20] or LiDAR odometry system [21]. Zheng *et al.* proposed a method combining UWB and VIO to measure triangulation uncertainty with multiple UWB sensors on each robot [22]. Song *et al.* developed a global UWB localization system enhanced by 2D LiDAR scan for map matching within an extended Kalman filter framework [23].

In this paper, we use a similar method by integrating VIO results with UWB measurements. Unlike other methods, we estimate the VIO drift to accurately determine the relative position between robots rather than directly estimating its position. This approach effectively eliminates the need for loop closure from vision-based or LiDAR-based cooperative localization systems.

## III. Proposed Method

The system depicted in Fig. 2 provides an overview of our method. Our algorithm obtains data from stereo camera, IMU, and UWB for processing. Each robot utilizes a VIO algorithm to determine its motion. UWB data is stored in a sliding window manner and employed to refine the trajectory through triangulation and error optimization modules.

### A. State Notation

For clarity, we introduce the notations used throughout this paper. We denote the pose of robot $i$ in world frame $w$ at time $m$ as a transformation matrix $\mathbf{T}_m^{w,i}$ consisting of rotation $\mathbf{R}_m^{w,i} \in SO(3)$ and position $\mathbf{p}_m^{w,i} \in (x_m^{w,i}, y_m^{w,i}, z_m^{w,i})$ element. Other robot $i$'s reference frames, such as the body frame centered at the IMU and VIO local frame, are denoted as $(\cdot)^{b_i}$ and $(\cdot)^{l_i}$, respectively. UWB sensor measurement from robot $i$'s to robot $j$'s position at time $m$ is denoted as $d_m^{i,j}$.

We record the state vector from VIO in a sliding window manner as follows:

$$\begin{aligned} \boldsymbol{\chi} &= [\mathbf{x}_0 \ \mathbf{x}_1 \ \dots \ \mathbf{x}_M], \\ \mathbf{x}_m &= [\mathbf{T}_m^{l_1,1} \ \dots \ \mathbf{T}_m^{l_N,N} \ \hat{\mathbf{T}}_m^{w,1} \ \dots \ \hat{\mathbf{T}}_m^{w,N} \\ &\qquad \Delta\hat{\mathbf{T}}_m^{w,1} \ \dots \ \Delta\hat{\mathbf{T}}_m^{w,N}], \end{aligned} \tag{1}$$

where $\mathbf{x}_m$ is the recorded state at time $m$. $\hat{\mathbf{T}}_m^{w,i}$ and $\Delta\hat{\mathbf{T}}_m^{w,i}$ denotes the estimated pose from UWB triangulation explained in Section III.*C* and estimated drift correction explained in Section III.*D*, respectively. All estimated drift corrections are initiaized according to the result of Section III.*B*. $N$ and $M$ are the number of deployed robots and sliding window size, respectively.

### B. Relative pose initialization

We followed the initialization method of [19] to obtain a consistent transformation. First, we define other robot's poses with respect to robot 1's initial pose. In other words, we maintain robot 1's initial pose $\mathbf{T}_0^{w,1}$ as $(0, 0, 0)$, which is the origin of the world frame. Robot 2's initial pose is aligned with the $x$-axis of the world frame and estimated as follows:

$$x_0^{w,2^2} = d_0^{1,2^2}. \tag{2}$$

Subsequently, robot 3's initial pose is used to define the $xy$-plane by resolving the equation as follows:

$$\begin{cases} x_0^{w,3^2} + y_0^{w,3^2} = d_0^{1,3^2} \\ (x_0^{w,3} - x_0^{w,2})^2 + (y_0^{w,3} - y_0^{w,2})^2 = d_0^{2,3^2} \end{cases}, \tag{3}$$

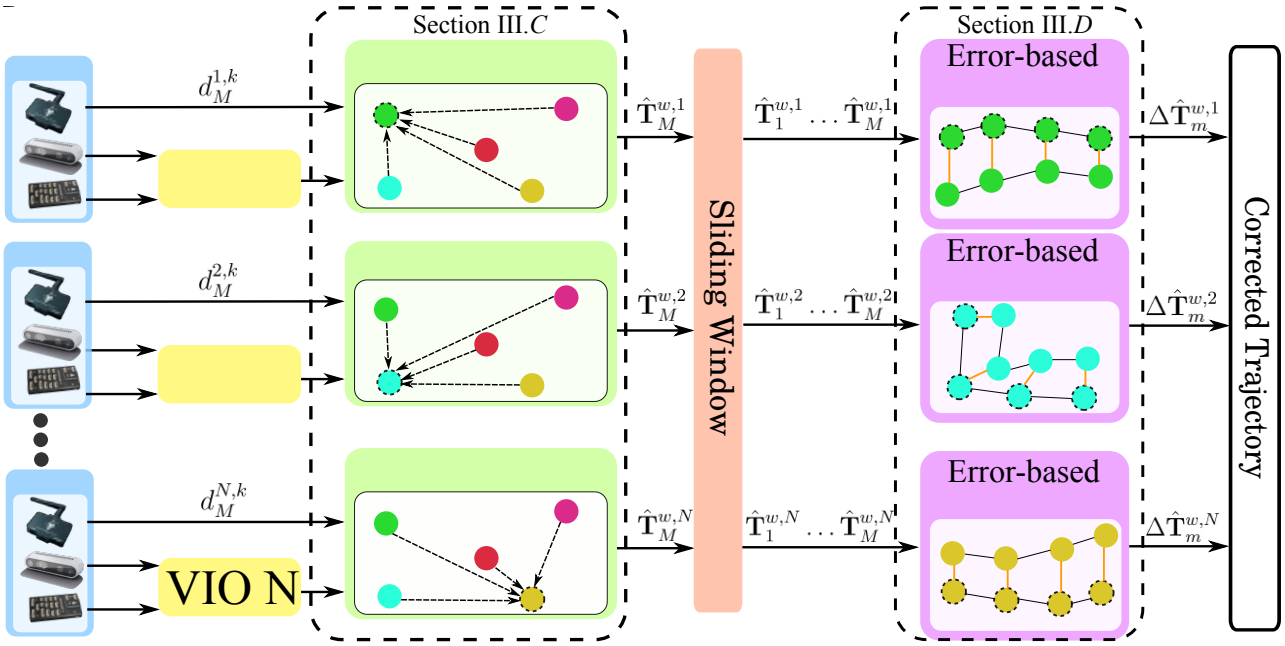


Fig. 2. Overall framework of our algorithm. We utilize the VIO algorithm to obtain the ego motion of the robot and couple it with UWB measurement to correct the trajectory. First, UWB-based triangulation is performed to obtain a triangulated pose based on other robots' VIO poses and UWB data $d_t^{i,k}$. The resulting triangulation result $\hat{\mathbf{T}}_t^{w,k}$ is stored in the sliding window. Second, error-based optimization uses the triangulated pose within the sliding window to estimate the VIO drift $\Delta\hat{\mathbf{T}}_t^{w,k}$.

Lastly, robot 4's measurements are utilized to define the $z$-axis orientation by solving the equation as follows:

$$\begin{cases} x_0^{w,4^2} + y_0^{w,4^2} + z_0^{w,4^2} = d_0^{1,4^2}, \\ (x_0^{w,4} - x_0^{w,2})^2 + (y_0^{w,4} - y_0^{w,2})^2 \\ \qquad + (z_0^{w,4} - z_0^{w,2})^2 = d_0^{2,4^2}, \\ (x_0^{w,4} - x_0^{w,3})^2 + (y_0^{w,4} - y_0^{w,3})^2 \\ \qquad + (z_0^{w,4} - z_0^{w,3})^2 = d_0^{3,4^2}. \end{cases} \tag{4}$$

The initial pose of other robot $i$'s can be computed as follows:

$$\begin{cases} x_0^{w,i^2} + y_0^{w,i^2} + z_0^{w,i^2} = d_0^{1,i^2}, \\ (x_0^{w,i} - x_0^{w,2})^2 + (y_0^{w,i} - y_0^{w,2})^2 \\ \qquad + (z_0^{w,i} - z_0^{w,2})^2 = d_0^{2,i^2}, \\ \qquad \cdots \\ (x_0^{w,i} - x_0^{w,i-1})^2 + (y_0^{w,i} - y_0^{w,i-1})^2 \\ \qquad + (z_0^{w,i} - z_0^{w,i-1})^2 = d_0^{i-1,i^2}. \end{cases} \tag{5}$$

The obtained initial pose of every robot can be utilized to transform the recorded VIO pose in the local frame to the world frame as follows:

$$\mathbf{T}_m^{w,i} = \mathbf{T}_0^{w,i}\ \mathbf{T}_m^{l_i,i} \tag{6}$$

### C. UWB Triangulation

VIO results will slightly deviate from its original position due to illumination or lost features. To address this problem, we use the UWB measurement and previously estimated drift correction to determine the triangulated pose, as shown in Fig. 2, as follows:

$$\min_{\hat{\mathbf{T}}_M^{w,i}} \sum_{k=1,k\neq i}^{N} \left\| d_M^{i,k} - (\hat{\mathbf{T}}_M^{w,i} - \mathbf{T}_M^{w,k}\Delta\hat{\mathbf{T}}_M^{w,k}) \right\|^2. \tag{7}$$

where $i$ and $k$ denotes current UAV and other UAV index, respectively. This equation is the UWB residual-based optimization where we attempt to find the most fit pose according to current UWB measurement at the latest time $M$. Notice that we do not use any information regarding previous time and only use other robot's current pose and UWB data. The current triangulated pose is inserted to the sliding window and used in the subsequent error-based optimization to obtain the current estimate of drift correction as explained in Section III.*D*.

To prevent the insertion of wrong triangulation results, we propose a simple approach to discard inconsistent triangulations or lost UWB data based on two conditions. First, we discard triangulation results that differ more than a predefined threshold from the original VIO pose. Given that the triangulation is derived from the VIO pose, we expect the results to closely align with the original position estimated by the VIO process. Second, we perform a check using past triangulated poses and velocity data from VIO. If the velocity derived from past triangulated poses closely matches the velocity data from the VIO, we accept the current triangulated pose result. We consider the triangulation result valid if both conditions are satisfied.

### D. Error-based Optimization

Inspired by other error-based optimization technique [24], we propose to further extend the previous UWB triangulation results by estimating the drift $\Delta\hat{\mathbf{T}}_t^{w,i}$ of robot $i$ from the sliding window of UWB triangulated poses. For every robot, we do a second optimization with every new pose added to

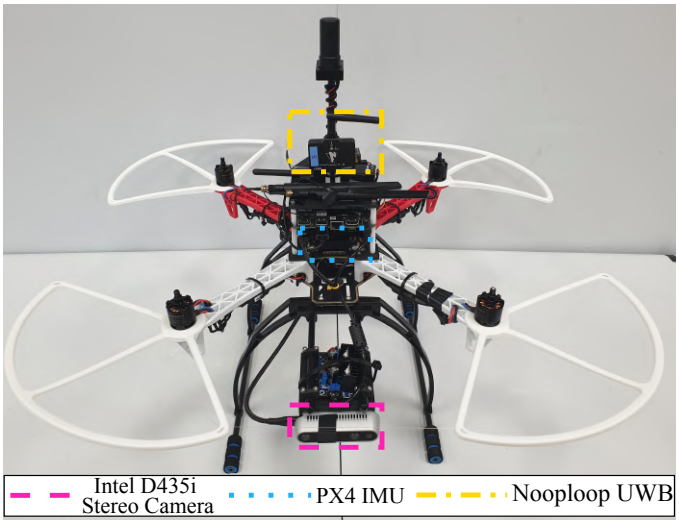


Fig. 3. UAV platform for our experiment equipped with camera, IMU, and UWB sensor for localization.

TABLE I. Processing time of each module on Intel(R) Core i7-1185G7.

| Process | Runtime (ms) |
|---|---|
| VIO preprocess | 2.0 |
| Pose triangulation | 5.0 |
| Error optimization | 2.0 |
| Postprocessing | 1.0 |
| Total time | **10.0** |

the sliding window as follows:

$$\min_{\Delta\hat{\mathbf{T}}_m^{w,i}} \sum_{m=1}^{M} \left\| \mathbf{T}_m^{w,i} \; \Delta\hat{\mathbf{T}}_m^{w,i} - \hat{\mathbf{T}}_m^{w,i} \right\|^2. \tag{8}$$

where $m$ denotes the time available in sliding window. Notice that this is a different expression from (7) because the objective is to minimize the drift based on recorded UWB triangulation poses. This approach effectively combines multiple triangulation errors into one optimization task. Moreover, it simplifies the underlying issue because optimizing the robot's pose directly may result in a significant deviation from its original pose. The final estimated drift correction will be used to correct the VIO trajectory resulting in the corrected trajectory output.

## IV. Experimental Evaluation

### A. Experimental Setup

We evaluated our algorithm's performance in the real world by utilizing five UAVs equipped with a stereo camera, IMU, and UWB sensors as depicted in Fig. 3. The UAVs operated in an open field with no infrastructure except Wi-Fi for data exchange. Each UAV also carries an Intel(R) i7-1185G7 CPU to simultaneously process the VIO and our algorithm. We utilized Nooploop UWB sensor that has a two-dimensional positioning accuracy around 10cm as specified by the manufacturer[1]. We selected VINS-Fusion [25] algorithm for VIO due to its minimal resource requirement. Additionally, we utilized RTK-GPS trajectory as ground truth for quantitative evaluation.

[1]https://www.nooploop.com/en/linktrack/

TABLE II. Absolute trajectory error (APE) and relative pose error (RPE) on experiments, which are expressed as APE/RPE (unit: m). Bold values indicate the best performance in `Aligned` results. Underlined values indicate the best performance in `Global` results.

| Exp1 | | UAV1 | UAV2 | UAV3 | UAV4 | UAV5 |
|---|---|---|---|---|---|---|
| VINS [25] | Aligned | 0.17/0.11 | 0.11/0.04 | 0.70/0.04 | 0.29/0.09 | 0.52/0.05 |
| Co-VINS [26] | Global | 0.24/0.09 | 0.65/0.27 | 1.53/0.19 | 1.44/0.05 | 2.54/0.31 |
| Direct | Aligned | 0.20/0.04 | 0.14/0.04 | 0.69/0.04 | 0.30/0.04 | 0.50/0.04 |
| | Global | 0.20/0.04 | 0.65/0.04 | 1.73/0.04 | 1.52/0.04 | 2.36/0.04 |
| Ours | Aligned | **0.16/0.02** | **0.08/0.02** | **0.45/0.02** | **0.16/0.03** | **0.48/0.02** |
| | Global | 0.16/0.02 | 0.49/0.04 | 1.10/0.04 | 1.34/0.04 | 1.86/0.04 |

| Exp2 | | UAV1 | UAV2 | UAV3 | UAV4 | UAV5 |
|---|---|---|---|---|---|---|
| VINS [25] | Aligned | 0.60/0.13 | 0.54/0.15 | 2.08/0.15 | 0.20/0.13 | 2.61/0.14 |
| Co-VINS [26] | Global | 20.51/0.30 | 15.27/0.25 | 9.70/0.52 | 17.26/0.27 | 13.05/0.27 |
| Direct | Aligned | 0.55/0.13 | 0.62/0.18 | 1.61/0.15 | 0.21/0.16 | 1.77/0.16 |
| | Global | 0.55/0.13 | 1.08/0.19 | 8.64/0.16 | 2.59/0.16 | 4.75/0.19 |
| Ours | Aligned | **0.32/0.10** | **0.31/0.14** | **0.72/0.10** | **0.13/0.10** | **1.20/0.12** |
| | Global | 0.32/0.10 | 0.89/0.14 | 4.26/0.10 | 2.00/0.11 | 2.03/0.12 |

The experiment is done in the same environment with two different scenarios. In `Exp1` dataset, we deployed all UAVs and ran the VIO algorithm with our proposed method normally. The `Exp2` dataset is coupled with a compromised VIO system (wrong extrinsic except for UAV1) to test our algorithm's capability to correct the overall trajectory.

Our evaluation is comprised of `Aligned` and `Global` results. `Aligned` refers to the result where each UAV's estimated trajectory is aligned with its ground truth trajectory. On the other hand, `Global` results align the UAVs' estimated trajectories to the global frame, which is UAV1's initial position.

We compared our proposed method with two open-sourced algorithm, VINS-Fusion[2] and Co-VINS[3] [26], and direct method. The direct method replaced the UWB triangulation and Error-based optimization module with a direct UWB optimization residual as follows:

$$\min_{\check{\mathbf{T}}_m^{w,i}} \sum_{k=1,k\neq i}^{N} \left\| d_m^{i,k} - (\check{\mathbf{T}}_m^{w,i} - \mathbf{T}_m^{w,k}) \right\|^2. \tag{9}$$

where $\check{\mathbf{T}}_m^{w,i}$ is the directly optimized pose result at time $m$. Compared with our method, the direct method does not consider any information regarding temporal triangulation error that might also be accumulated from UWB triangulation and directly substitute the obtained VIO pose by $\check{\mathbf{T}}_m^{w,i}$.

We used both the absolute trajectory error (APE) and relative pose error (RPE) with reference to available ground truth. APE measures the absolute deviation of each pose from the ground truth with the nearest timestamp. RPE computes the displacement error between successive timestamps. We utilize the EVO python package to compute both errors [27].

### B. Error-Correction Analysis

As shown in Table II, our algorithm can attain better results compared with the VIO algorithm itself. VINS-Fusion and Co-VINS have a loop closing module that can correct

[2]https://github.com/HKUST-Aerial-Robotics/VINS-Fusion
[3]https://github.com/qintonguav/Co-VINS

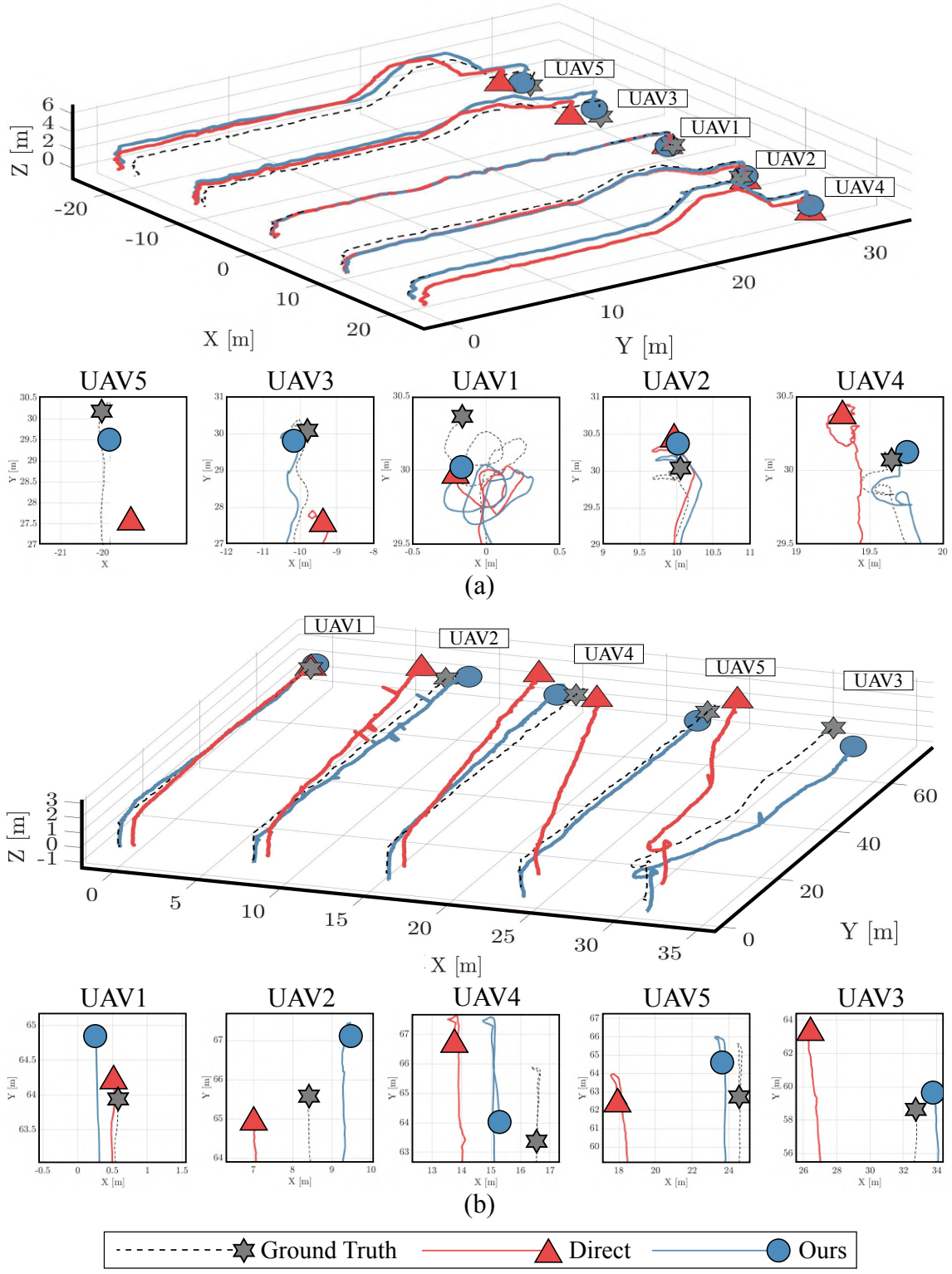


Fig. 4. Qualitative comparison of (a) `Exp1` and (b) `Exp2` performed in an open environment with no infrastructure. Our algorithm performs better and resembles the ground truth trajectory compared with the direct method. The symbol on each trajectory denotes the endpoint. All trajectories are shown in reference to UAV1's initial pose (0, 0, 0).

the trajectory when recognizing similar images. However, it relies on matching images and keypoints which are not always available in an open environment. In `Exp2`, Co-VINS suffered significant divergence due to incorrect relative pose from the loop closing module. In contrast, our method can correct the trajectory without matching images or keypoints. This result is obtained from our module's constant correction and optimization that tracks the drift error. Our method can better capture the drift's evolution by combining UWB triangulation and error-based optimization modules. This analysis confirms our claim that the proposed method can effectively correct long-term VIO drift.

### C. Comparison with Direct Approach

Compared with direct relative pose estimation, our optimization approach can correct the trajectory better and obtain lower APE on `Aligned` and `Global` error. In the direct optimization scheme, the algorithm tends to amplify VIO error as the disparity between ground truth and estimation widens. Direct optimization often struggles to identify the optimal pose due to big differences in residuals. In `Exp1` dataset, the direct algorithm can closely follow the ground truth trajectory but our algorithm performs better as depicted in Fig. 4(a). This problem is more apparent in `Exp2` dataset where the direct optimization result drifts from its supposed trajectory as illustrated in Fig. 4(b). However, our method does not encounter this issue and can freely estimate the drift. This comparison supports the claim that our algorithm can reliably estimate relative pose even in an anchorless environment.

### D. Runtime

The computational time of the algorithm is presented in Table I. VIO poses are obtained at around 30 Hz. Our algorithm can give odometry results at a frequency of 25-30 Hz. We also believe that our algorithm can efficiently generate reliable odometry results for multiple robots, as it operates independently of the VIO process and is designed to be lightweight. This result supports our method's real-time capability with low-cost sensors.

## V. Conclusion

In this paper, we have proposed a novel algorithm that seamlessly combines VIO results with UWB measurement to improve trajectory estimation. Our dual optimization framework utilizes UWB triangulation alongside error-based correction. UWB triangulation seeks the most accurate pose based on range measurements. Error-based optimization differs from other UWB-assisted relative pose localization by estimating the VIO drift error, which is unconstrained by the original VIO pose.

Future works could enhance the algorithm by using more robust triangulation methods that consider UWB's high-rate information. Furthermore, a deep learning framework that estimates the error can also be incorporated for better optimization.

## References


[1] S. Chen, M. J. O'Brien, F. Talbot, J. Williams, B. Tidd, A. Pitt, and R. C. Arkin, "Multi-modal user interface for multi-robot control in underground environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, 2022, pp. 9995–10 002.

[2] A. Ribeiro and J. Conesa-Muñoz, "Multi-robot systems for precision agriculture," *Innovation in Agricultural Robotics for Precision Agriculture*, pp. 151–175, 2021.

[3] M. Rodrguez, A. Al-Kaff, . Madridano, D. Martn, and A. de la Escalera, "Wilderness search and rescue with heterogeneous multi-robot systems," in *Proc. IEEE Int. Conf. Unmanned Aircr. Syst.*, 2020, pp. 110–116.

[4] P.-Y. Lajoie, B. Ramtoula, F. Wu, and G. Beltrame, "Towards collaborative simultaneous localization and mapping: a survey of the current research landscape," *arXiv preprint arXiv:2108.08325*, 2021.

[5] J. P. Queralta, J. Taipalmaa, B. Can Pullinen, V. K. Sarker, T. Nguyen Gia, H. Tenhunen, M. Gabbouj, J. Raitoharju, and T. Westerlund, "Collaborative multi-robot search and rescue: Planning, coordination, perception, and active vision," *Ieee Access*, vol. 8, pp. 191 617–191 643, 2020.

[6] A. Martinelli, F. Pont, and R. Siegwart, "Multi-robot localization using relative observations," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2005, pp. 2797–2802.

[7] P. Schmuck, T. Ziegler, M. Karrer, J. Perraudin, and M. Chli, "COVINS: Visual-inertial SLAM for centralized collaboration," *arXiv preprint arXiv:2108.05756*, 2021.

[8] Y. Xie, Y. Zhang, L. Chen, H. Cheng, W. Tu, D. Cao, and Q. Li, "Rdc-slam: A real-time distributed cooperative slam system based on 3d lidar," *IEEE Trans. Intell. Transport. Syst.*, vol. 23, no. 9, pp. 14 721–14 730, 2022.

[9] J. Li, K. Koreitem, D. Meger, and G. Dudek, "View-invariant loop closure with oriented semantic landmarks," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2020, pp. 7943–7949.

[10] Y. Cao, C. Yang, R. Li, A. Knoll, and G. Beltrame, "Accurate position tracking with a single uwb anchor," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2020, pp. 2344–2350.

[11] B. Yang, J. Li, and H. Zhang, "UVIP: Robust UWB aided visual-inertial positioning system for complex indoor environments," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 5454–5460.

[12] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[13] C. Jennings, D. Murray, and J. J. Little, "Cooperative robot localization with vision-based mapping," in *Proc. IEEE Int. Conf. Robot. Automat.*, 1999, pp. 2659–2665.

[14] P. Zhu, Y. Yang, W. Ren, and G. Huang, "Cooperative visual-inertial odometry," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 13 135–13 141.

[15] M. Patel, M. Karrer, P. Bnninger, and M. Chli, "COVINS-G: A generic back-end for collaborative visual-inertial SLAM," *arXiv:2301.07147*, 2023.

[16] G. Loianno, J. Thomas, and V. Kumar, "Cooperative localization and mapping of MAVs using RGB-D sensors," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2015, pp. 4021–4028.

[17] W. Hess, D. Kohler, H. Rapp, and D. Andor, "Real-time loop closure in 2d LiDAR SLAM," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2016, pp. 1271–1278.

[18] S. Fang, H. Li, and M. Yang, "LiDAR SLAM based multivehicle cooperative localization using iterated split CIF," *IEEE Trans. Intell. Transport. Syst.*, vol. 23, no. 11, pp. 21 137–21 147, 2022.

[19] J. Choi, E. Lee, M. Jeong, D. Choi, and H. Myung, "Range-only relative position estimation for multi-UAV," in *Proc. Int. Conf. Control, Automat. Syst.*, 2022, pp. 1984–1989.

[20] S. Shin, E. Lee, J. Choi, and H. Myung, "MIR-VIO:mutual information residual-based visual inertial odometry with UWB fusion for robust localization," in *Proc. Int. Conf. Control, Automat. Syst.*, 2021, pp. 91–96.

[21] H. Zhou, Z. Yao, and M. Lu, "Lidar/UWB fusion based SLAM with anti-degeneration capability," *IEEE Trans. Veh. Technol.*, vol. 70, no. 1, pp. 820–830, 2021.

[22] S. Zheng, Z. Li, Y. Liu, H. Zhang, P. Zheng, X. Liang, Y. Li, X. Bu, and X. Zou, "UWB-VIO fusion for accurate and robust relative localization of round robotic teams," *IEEE Robot. Automat. Lett.*, vol. 7, no. 4, pp. 11 950–11 957, 2022.

[23] Y. Song, M. Guan, W. P. Tay, C. L. Law, and C. Wen, "UWB/LiDAR fusion for cooperative range-only SLAM," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2019, pp. 6568–6574.

[24] W. Xu and F. Zhang, "Fast-lio: A fast, robust lidar-inertial odometry package by tightly-coupled iterated kalman filter," *IEEE Robot. Automat. Lett.*, vol. 6, no. 2, pp. 3317–3324, 2021.

[25] T. Qin, J. Pan, S. Cao, and S. Shen, "A general optimization-based framework for local odometry estimation with multiple sensors," *arXiv preprint arXiv:1901.03638*, 2019.

[26] T. Qin, W. Wu, and S. Shen, "Collaborative localization for multiple monocular visual-inertial systems," Accessed on: Dec. 20, 2023. [Online], Available: https://github.com/qintonguav/Co-VINS.

[27] M. Grupp, "EVO: Python package for the evaluation of odometry and SLAM." Accessed on: Nov. 21, 2023. [Online], Available: https://github.com/MichaelGrupp/evo.