



**Politecnico
di Torino**

ScuDo

Scuola di Dottorato - Doctoral School
WHAT YOU ARE, TAKES YOU FAR

Doctoral Dissertation

Doctoral Program in Aerospace Engineering (38th cycle)

UAV-UGV Autonomous Multi-Robotic Cooperation in GNSS-Denied Environments

By

Alessandro Minervini

Supervisor(s):

Prof. Giorgio Guglieri, Supervisor

Prof. Marcello Chiaberge, Co-Supervisor

Doctoral Examination Committee:

Prof. Giulio Avanzini, Università di Roma "La Sapienza"

Prof. Stefano Cacciola, Politecnico di Milano

Politecnico di Torino

2026

Declaration

I hereby declare that, the contents and organization of this dissertation constitute my own original work and does not compromise in any way the rights of third parties, including those relating to the security of personal data.

Alessandro Minervini

2026

* This dissertation is presented in partial fulfillment of the requirements for **Ph.D. degree** in the Graduate School of Politecnico di Torino (ScuDo).

A nonna Rosa e nonno Sergio

Acknowledgements

I would like to express my gratitude to my supervisor, Prof. Giorgio Guglieri, who allowed me to join his research group and investigate extremely interesting topics in the robotics field.

I am also deeply grateful to my co-supervisor, Prof. Marcello Chiaberge, for hosting me in his interdepartmental center, the PIC4SeR. Being part of such a stimulating environment and receiving the technical guidance of the entire staff has been fundamental to both my scientific and personal growth.

A special thanks goes to the members of the DRAFT student team, who supported me in many research activities with enthusiasm, availability, and high-level expertise.

I would also like to thank Prof. Adrian Carrio for hosting me at his company, Dronomy, during my research abroad period. The months I spent working with him and his team were extremely valuable for the development of this work.

Finally, I would like to express my deepest gratitude to my parents, my sister, and my girlfriend for their constant support and encouragement throughout this journey.

Abstract

Autonomous navigation in GNSS-denied environments remains a fundamental challenge in robotics. Heterogeneous multi-robot formations composed of UAVs and UGVs have gained considerable interest within the scientific community, as they are particularly advantageous for accomplishing missions in real-world scenarios. One of the most critical aspects to enable multi-robotic cooperation in GNSS-denied environments is robot localization. The presence of both ground and aerial platforms can improve the design of the platforms in the formation by equipping each robot with a sensor setup tailored to its physical and operational characteristics. Additionally, shared information among the robots can introduce additional constraints to improve localization accuracy.

A typical configuration in service robotics consists of UGVs cooperating with rotary-wing UAVs. These aerial platforms are significantly constrained by strict limitations in weight, size, and power consumption; thus, some sensor setups are inconvenient in applications requiring small dimensions and high flight endurance. For instance, 3D LiDAR-based localization algorithms can provide high accuracy and robustness, representing some of the most comprehensive solutions for navigation purposes. However, their applicability to UAVs is limited due to their weight, size, and power consumption.

On the other hand, vision-based approaches such as Visual-Inertial Odometry (VIO) offer optimal solutions for their installation simplicity, low weight, and power consumption. Additionally, in many scenarios, image data represent the final output of the mission, while LiDAR sensors are often limited to support navigation and obstacle avoidance. Therefore, exploiting cameras for both perception and navigation purposes reduces sensor redundancy and enables a more compact, lightweight, and energy-efficient sensor setup. However, vision-based localization algorithms suffer from drift accumulation and poor robustness under challenging conditions, such as

low-texture environments, high-speed motion, and brightness changes. These limitations effectively reduce their applicability in real-world scenarios, with substantial consequences on system design and overall efficiency.

This dissertation focuses on improving the localization performance of UAVs relying on VIO algorithms. The limitations of visual-based navigation are identified, and different methodologies are introduced to address the typical failures experienced by these algorithms. To improve robustness in challenging environments, state-of-the-art pipelines are analyzed, and the solutions are implemented within popular existing frameworks.

First, monocular setups are analyzed, and methodologies to mitigate common issues related to these setups are introduced. The inherent limitations of these configurations are addressed through a geometric initialization strategy, and adjustments to the optimization process are introduced to mitigate solution divergence during pure rotational motions.

Then, the focus is shifted to the problem of insufficient feature tracking, which substantially reduces the reliability of visual-based systems. To address this issue, an IMU-based prediction of the feature locations across consecutive frames is included within an open-source VIO pipeline to improve performance during high-speed motions. Subsequently, a generalized multi-camera setup is implemented within the same framework to enhance localization in challenging environments.

The drift accumulated by VIO algorithms is addressed with two sensor-fusion techniques. To avoid significant increases in the payload, lightweight sensors are utilized for this scope. In particular, VIO estimates are combined with the measurements from a downward-facing range finder to regularly correct the altitude drift accumulated during the navigation. In the context of multi-robotic cooperation, a sensor fusion method leveraging the UGV sensor setup is introduced. The approach effectively reduces the UAV drift by combining the UAV detections obtained by the UGV cameras with the 3D LiDAR point cloud acquired by the ground platform.

Experimental results demonstrate that the proposed methodologies significantly improve localization accuracy, robustness, and global consistency of VIO algorithms across a range of challenging scenarios. Moreover, the results confirm that these improvements can be achieved while preserving the compactness and limited computational requirements of small aerial platforms.

Contents

List of Figures	x
List of Tables	xiv
1 Introduction	1
1.1 Overview of UAVs and UGVs Operational Characteristics	2
1.2 UAVs and UGVs in GNSS-Denied Scenarios	3
1.2.1 Warehouse Logistics and Material Handling	3
1.2.2 Industrial and Subterranean Environments Inspections	4
1.2.3 Military and Surveillance Applications	5
1.2.4 Space Exploration	6
1.3 UAV-UGV Cooperation from a Localization Perspective	8
1.4 State-of-the-Art of Navigation Solutions in GNSS-Denied Environ- ments	9
1.4.1 Visual Inertial Odometry	10
1.4.2 LiDAR–Inertial Odometry	14
1.4.3 Infrastructure-Based Localization Systems	18
1.4.4 A Qualitative Comparison of the Presented Solutions	20
1.5 Focus and Scope of the Work	22
1.5.1 The Importance of Cameras for Mission Requirements	24

1.5.2	The Topics Addressed	25
1.6	Chapters Overview	26
2	Open Challenges of VIO	27
2.1	Mono-Camera Setups	28
2.2	Insufficient Feature Tracking	31
2.3	Global Drift Accumulation	33
2.4	Computational complexity	35
3	Improvements to State-of-the-Art VIO Pipelines	37
3.1	Proposed solutions to Monocular Setup Issues	37
3.1.1	Initialization Phase	39
3.1.2	Pure Rotational Motions	41
3.1.3	Experimental Evaluation	44
3.1.4	Results	46
3.1.5	Discussion	53
3.2	Basalt VIO: An Open-Source VIO Pipeline	55
3.3	Improving VIO Robustness During High-Speed Motions	56
3.3.1	IMU-based Feature Prediction	58
3.3.2	Experimental Evaluation	64
3.3.3	Results	65
3.3.4	Discussion	67
3.4	Multi-Camera Pipeline Implementation	67
3.4.1	Generalization of the Image Processing	68
3.4.2	Keyframe Generation	69
3.4.3	Experimental Evaluation	70
3.4.4	Results	74

3.4.5	Discussion	78
4	Sensor Fusion for VIO Drift Reduction	80
4.1	Overview of the Proposed Solutions	81
4.2	Range-Aided Altitude Estimation	82
4.2.1	Related Works	83
4.2.2	Mathematical Formulation of the Problem	83
4.2.3	Altitude Estimation	86
4.2.4	Mahalanobis Distance for Inconsistency Detection	88
4.2.5	Experimental Evaluation	89
4.2.6	Results	91
4.2.7	Discussion	91
4.3	Vision–LiDAR Cooperative Localization	94
4.3.1	Related Works	94
4.3.2	UAV Raycasting with 3D-LiDAR Point Cloud	96
4.3.3	Experimental Evaluation	99
4.3.4	Results	101
4.3.5	Discussion	105
5	Conclusions	107
	References	111

List of Figures

1.1	Artist's concept of the Perseverance rover and Ingenuity helicopter on Mars. Credit: NASA/JPL-Caltech (Public Domain).	8
1.2	Timeline evolution of Visual Odometry	11
1.3	Timeline evolution of LiDAR-based odometry	15
1.4	The diagram qualitatively compares the three localization solutions across key performance metrics. Scores range from 1 (least favorable) to 5 (most favorable).	22
2.1	The figure compares the dimensions (in centimeters) of the Intel RealSense T265 with and without its original protective cover. The device weighs 54 g with the cover and 13 g when the cover is removed.	36
3.1	Depth degeneration in the VINS-Mono implementation during a pure rotational motion: the pictures show the frames after a yaw rotation. In red and in violet, the points have depths respectively below and above 10 m.	42
3.2	The baseline framework logic (left) compared with our implementation (right)	44
3.3	The D455 camera from RealSense.	45

3.4	Visualization of the planar surface utilized for geometrical constraints during the initialization phase. The 3D visualization shows the plane and the position of the detected features whose back-projected rays intersect the plane. In the picture, the planar surface is bounded by yellow lines, and the features that lie on the surface are visualized in red. The numbers close to the features are their corresponding IDs.	48
3.5	Visualization of the trajectory for D1, D2, D3, and D4. The blue trajectory corresponds to the solution of the baseline algorithm. The red trajectory corresponds to our implementation. The green trajectory is the reconstruction of the ground truth according to the known reference position.	49
3.6	Trajectory visualization for D5 and D6. The pictures show the comparison of the baseline framework trajectory (blue curve) with our implementation (red curve).	51
3.7	The estimated trajectory evaluated over a subset of the EuRoC MAV datasets. The results show the baseline implementation (blue trajectory), proposed solution (red trajectory) and ground truth (green trajectory).	52
3.8	The OAK-D Pro Wide stereo camera from Luxonis	64
3.9	Impact of the IMU prediction: (a) The histogram for the <i>Easy</i> dataset counts the occurrences of features with prediction errors in pixel units. (b) The <i>Medium</i> dataset error distribution. (c) The <i>Difficult</i> dataset error distribution. The data is collected over 200 frames. . . .	66
3.10	The flow chart illustrating the optical flow generation algorithm generalized to a multi-camera system.	69
3.11	The stereo camera configurations: (a) <i>Stereo1</i> (cam0 and cam1, yellow field of view) and <i>Stereo2</i> (cam2 and cam3, red field of view) with a shared field of view between cam1 and cam2 (blue); (b) <i>Stereo1</i> and <i>Stereo2</i> with non-overlapping fields of view.	72
3.12	The two configurations of the multi-camera setup: (a) The two stereo cameras are tilted at 45 degrees; (b) The two stereo cameras point in opposite directions.	73

3.13	Comparison of the trajectory in the xy plane: (a) Trajectories for <i>config1</i> on D3; (b) Trajectories for <i>config2</i> on D4. The trajectories refer to the data collected after returning to the VICON room.	76
3.14	The multi-camera setup during robustness testing: <i>cam0</i> and <i>cam1</i> (Stereo 1); <i>cam2</i> and <i>cam3</i> (Stereo 2).	77
4.1	Schematic of the Adaptive Kalman Filter fusing range finders, FCUs, and T265's measurements.	89
4.2	Overview of the UAV's experimental setup.	90
4.3	Adaptive Kalman Filter managing range finder inconsistency.	92
4.4	Mahalanobis distance considerably increases when range finder inconsistencies occur.	93
4.5	Altitude estimated by the Adaptive Kalman Filter ensures minor drift compared to the T265 altitude.	94
4.6	Raycast of the UAV's bounding box. (a) illustrates the sensor configuration to detect the UAV bounding box, while (b) shows the pyramid obtained from back-projecting the bounding-box vertices.	96
4.7	The robots used for data collection: UAV (a) and UGC (b)	99
4.8	The Livox Mid-360 (a) and its field of view (b)	100
4.9	Detection of the UAV: the figure shows, for the three datasets, a frame where the UAV was successfully recognized. The bounding box obtained directly by the object detection module is in red, while the manually reduced bounding box is in blue. (a) refers to D1, (b) to D2 and (c) to D3.	103
4.10	3D visualization of the results: the figure shows, for the dataset D2, the LiDAR point cloud (gray points), the points inside the pyramid (red points), the cluster estimated to correspond to the UAV (blue points), the rays obtained from the back-projection of the bounding box vertices (green lines), and the distance vector from the camera to the first hit cluster (cyan line). (a) Illustrates the projection on the horizontal plane, while (b) presents the full 3D visualization of the scene.	104

5.1	The diagram qualitatively compares the three localization solutions across key performance metrics. Scores range from 1 (least favorable) to 5 (most favorable). A detailed explanation of the figure is provided in 1.4.4.	109
-----	---	-----

List of Tables

3.1	Comparison of the selected metrics across the four datasets for two distinct cases.	47
3.2	Errors in meters evaluated for both implementations across the two datasets. R1 indicates the error accumulated after the first rotation, while EOS stands for End of Sequence error. Errors below 0.1 m are in green, while errors above 0.5 m are in red.	51
3.3	Comparison of the selected metrics across EuRoC datasets. The RMSE is provided in meters.	51
3.4	Nomenclature of all used symbols.	58
3.5	Errors in prediction estimate: the table shows the average error computed over the sequences (Avg Error) and the maximum error obtained over all frames.	65
3.6	Scores assigned to the challenging visual conditions in the considered datasets. A score of zero indicates the absence of the condition, whereas a score of ten indicates its maximum presence.	73
3.7	RMSE ATE comparison (values in meters) for <i>config1</i> with average CPU load percentages. For each dataset, the total traveled distance is indicated in parentheses next to the dataset name.	75
3.8	RMSE ATE comparison (values in meters) for <i>config2</i> with average CPU load percentages. For each dataset, the total traveled distance is indicated in parentheses next to the dataset name.	75
3.9	Comparison of final estimated positions (meters) between Stereo 1 and Multicamera setups (dataset D5).	77

4.1	Comparison of the chosen metrics across the 3 datasets	102
-----	--	-----

Chapter 1

Introduction

Autonomous robots represent the most convenient solution in terms of costs and time saving for several applications. Typically, these robots are distinguished into three main categories: ground vehicles, aerial vehicles, and marine vehicles. Ground vehicles may have many configurations, and their names change accordingly. Aerial vehicles, instead, are commonly classified as fixed-wing and rotary-wing and are typically referred to as drones. Finally, marine vehicles can be divided into surface and underwater platforms.

When these robots are able to navigate an environment without continuous human control, they are named autonomous or unmanned robots. For this reason, researchers usually refer to these robots as Unmanned Ground Vehicles (UGVs), Unmanned Aerial Vehicles (UAVs), Uncrewed Surface Vessels (USVs), and Autonomous Underwater Vehicles (AUVs). While remaining of great interest for the scientific community, USVs and AUVs fall outside the scope of the proposed study. Since this dissertation examines UAV-UGV cooperation in GNSS-denied environments, this paragraph provides a general overview of these two platforms to outline their main characteristics, operational capabilities, and complementary roles within a cooperative robotic system.

1.1 Overview of UAVs and UGVs Operational Characteristics

Rotary-wing UAVs (RW-UAVs) are particularly suitable for tasks requiring agile three-dimensional motion, rapid deployment, and the observation of elevated or hardly accessible areas. For example, they can inspect the upper portions of storage racks, industrial plants, buildings, or infrastructure that would be difficult or unsafe to reach from the ground. Their ability to hover and to change viewpoint rapidly also makes them effective for visual inspection and environmental perception tasks. However, these capabilities are typically constrained by their limited flight endurance and payload capacity [1, 2].

Fixed-wing UAVs are particularly suitable for missions requiring long-range coverage, extended flight endurance, and efficient operation over large areas. Their aerodynamic efficiency enables them to carry out tasks such as large-scale environmental monitoring, agricultural surveying, infrastructure inspection, and wide-area surveillance more effectively than rotary-wing platforms. Moreover, their ability to maintain higher cruising speeds makes them advantageous for time-critical missions involving long trajectories. However, fixed-wing UAVs cannot normally hover and have limited capability to inspect confined spaces or structures from multiple close-range viewpoints. Their deployment may also require dedicated launch and recovery procedures, sufficient open space, or additional vertical-take-off-and-landing mechanisms [3, 4].

UGVs can generally operate for long periods, carry heavier sensing and computing equipment, and provide stable ground-level observations of the environment. In the considered configuration, the UGV can be equipped with sensors that would be difficult to integrate onboard a lightweight UAV, such as a three-dimensional LiDAR, high-capacity batteries, or more powerful onboard computing hardware. The ground platform can therefore support long-duration mapping, environmental monitoring, and high-accuracy localization tasks, while the UAV provides flexible aerial observations [5–7].

1.2 UAVs and UGVs in GNSS-Denied Scenarios

Typically, the environments where autonomous robots are employed are distinguished into GNSS-available and GNSS-unavailable environments. This distinction exists because of the simplifications relying on environments where the GNSS signal is assumed to be always reliable. In GNSS-denied or partially denied environments, the robots should be provided with another localization source to enable autonomous motion. Since a considerable number of real-world applications involve a GNSS-denied environment, scientific research is focusing on addressing the challenges of robot localization in such conditions. The following paragraphs review some of the most relevant application fields in which heterogeneous UAV-UGV systems can provide a significant operational advantage.

1.2.1 Warehouse Logistics and Material Handling

Warehouses represent one of the most relevant GNSS-denied environments for UAV-UGV cooperation. Their high operational throughput requires the continuous handling of incoming and outgoing goods, pallet movements, order picking, and storage operations. As a result, discrepancies may arise between the physical location of goods and the information recorded in the warehouse management system, reducing inventory traceability and potentially causing delays, misplaced items, or stock inaccuracies. Regular inventory checks are therefore essential to maintain an accurate representation of stored goods and to promptly identify inconsistencies. Due to the high repetitiveness of some procedures, such as pallet transportation, inventory checks, and goods identification, robotic automation is particularly attractive for reducing operation times, costs, and human error.

UGVs can be employed for goods transportation and storage operations, while RW-UAVs are particularly suitable for inspecting shelves and acquiring information about the inventory state [8]. In particular, aerial platforms can access elevated locations without requiring dedicated lifting systems or manual inspection procedures. They can acquire images of labels, barcodes, QR codes, or stored goods, and can rapidly inspect areas that are inaccessible or inefficient to reach using only ground robots. Research works have investigated heterogeneous UAV-UGV systems for warehouse inventory operations [9–11]. This division of roles can reduce the

time required for inventory operations and improve the accessibility of vertically distributed storage areas.

The two platforms can also cooperate more directly. A UGV may act as a mobile base for UAV charging and data transmission, thereby compensating for the limited flight endurance of rotary-wing UAVs. Moreover, the UGV can transport sensing and computing hardware that would be unsuitable for a lightweight aerial platform, including high-capacity batteries, high-performance processors, and 3D LiDAR sensors.

1.2.2 Industrial and Subterranean Environments Inspections

UAV-UGV cooperation is also particularly relevant for the inspection of industrial plants, infrastructures, and subterranean environments. Examples include caves, tunnels, underground infrastructures, mines, and building sites. Such environments are generally characterized by limited illumination, irregular terrain, narrow passages, and potentially hazardous conditions for human operators. Especially, the risk associated with human access to such environments motivates the employment of autonomous robots in these scenarios.

In particular, UGVs can provide long-endurance ground-level exploration and transport heavy sensing payloads, such as 3D LiDAR sensors, high-capacity batteries, or powerful computing platforms. RW-UAVs can instead access elevated areas, rapidly inspect vertical structures, and explore regions that are difficult or impossible for a ground vehicle to reach. By sharing maps, environmental observations, or relative-position measurements, the two platforms can improve the situational awareness of the overall robotic system [12–14].

The growing interest in these scenarios has motivated the Defense Advanced Research Projects Agency (DARPA) to establish the DARPA Subterranean (SubT) Challenge in 2017. The program was created to promote the development of innovative robotic technologies capable of autonomously mapping, navigating, searching, and operating in complex underground environments. The considered scenarios include human-made tunnel systems, underground urban infrastructures, and natural cave networks. Relevant use cases include search-and-rescue operations, infrastructure inspection, oil and gas exploration, archaeology, and scientific exploration [15]. To address these requirements, the DARPA SubT Challenge encouraged multidis-

disciplinary approaches combining advances in autonomous navigation, perception, communication networks, mobility, and multi-robot coordination.

The operational task consisted of autonomously exploring an initially unknown environment, constructing a spatial representation of the explored area, detecting predefined artifacts, and reporting their estimated locations to a central command post. The performance of the robotic teams was evaluated according to the number of correctly identified artifacts and the accuracy of their reported positions. The challenge included both a systems competition, based on physical robotic deployments, and a virtual competition, based on simulation environments.

The DARPA SubT Challenge did not focus exclusively on a single UAV-UGV pair. Instead, it promoted the use of heterogeneous teams composed of tracked, wheeled, legged, and aerial robots. This heterogeneity was considered fundamental to enable the exploitation of complementary capabilities. Several relevant works were produced as a result of the competition [16–18].

Overall, the DARPA Subterranean Challenge demonstrates that GNSS-denied subterranean inspection and search-and-rescue missions benefit substantially from heterogeneous robotic cooperation. In particular, the challenge highlights the importance of resilient onboard localization, multi-modal sensing, map sharing, communication-aware planning, and complementary UAV-UGV capabilities.

1.2.3 Military and Surveillance Applications

Military and surveillance operations represent a particularly relevant application domain for cooperative UAV-UGV systems. In military scenarios, GNSS signals cannot be assumed to be continuously available or reliable, as in some situations, this localization source is intentionally denied through jamming. Consequently, individual platforms may experience a progressive degradation of their navigation estimate, compromising the execution of reconnaissance, target observation, patrol, and search operations.

This problem is particularly critical for unmanned systems, since reliable localization is required not only to estimate the robot trajectory but also to maintain situational awareness, coordinate multiple assets, and safely execute autonomous tasks [19]. In these conditions, a cooperative architecture can reduce the dependence of each platform on GNSS by allowing robots to exchange navigation information

and exploit relative observations, such as inter-robot ranging or shared state estimates [20, 21]. UAVs can provide wide-area observations from elevated viewpoints, whereas UGVs can operate closer to the environment and can accommodate more substantial sensing and computing payloads. The complementary sensing and mobility characteristics of the two platforms can therefore provide additional navigation constraints and improve the resilience of the overall system in GNSS-contested environments [22]

In these scenarios, the ability to combine onboard sensing with inter-robot information exchange is particularly relevant. A UGV equipped with cameras, LiDAR sensors, or alternative positioning systems can provide an additional source of geometric information to support the navigation of a UAV. Conversely, aerial observations can improve the situational awareness and path-planning capabilities of the ground vehicle. For these reasons, enhanced multi-robot localization and cooperation are increasingly relevant research topics for the military field.

1.2.4 Space Exploration

Autonomous robots represent a valuable solution for space exploration missions. During these missions, latency in command transmissions does not allow for continuous human control from Earth; thus, robots must be provided with an autonomous navigation system. Due to the GNSS signal absence, localization must rely on alternative solutions that typically consist of fusing measurements coming from onboard sensors.

Several space-exploration missions have already demonstrated the value of autonomous robotic systems operating beyond the Earth. For instance, JAXA's Hayabusa2¹ mission deployed the two MINERVA-II1 rovers on the surface of asteroid Ryugu. After their release from the mother spacecraft, the rovers autonomously hopped across the asteroid surface and acquired images and scientific measurements in an environment where direct and continuous control from Earth was not feasible.

A particularly relevant future example of autonomous multi-robot cooperation is NASA's Cooperative Autonomous Distributed Robotic Exploration (CADRE)² mission. The project is designed to deploy three small rovers and a stationary base

¹<https://www.hayabusa2.jaxa.jp/en/>

²<https://www.jpl.nasa.gov/missions/cadre/>

station on the lunar surface, where the robots will autonomously coordinate exploration, mapping, and distributed ground-penetrating-radar measurements. CADRE represents a direct example of how cooperative autonomy can enable robotic teams to perform scientific operations under the communication delays, limited resources, and navigation constraints typical of planetary environments [23].

Probably, the most popular - and even most relevant - example of planetary exploration involving multi-robotic cooperation is the mission Mars 2020 by NASA³. In this mission, the rover Perseverance and the drone Ingenuity have been employed to explore the Martian surface. Both robots rely on onboard sensors for navigation, being equipped with cameras, inertial sensors, and wheel encoders. Particularly, Ingenuity is equipped with a mono-camera and IMU sensors, while Perseverance's navigation system is based on a stereo camera, IMU sensors, and wheel encoders.

The mission has the primary objectives of investigating past microbial life on Mars, characterizing the planet's geology and climate, and collecting rock samples. In this mission, Perseverance originally had the role of the main scientific platform, while Ingenuity was utilized as a technology demonstration. However, after the first successful flights, Ingenuity began to be directly involved in the primary mission with the role of operational scouting. During its operative life, Ingenuity was used to identify safe or scientifically interesting routes for Perseverance, enhancing the mission's effectiveness. On January 18 2024, Ingenuity performed its last flight after crashing from an altitude of 12 meters. The report published by NASA in [24] announced that the crash was due to a failure in the localization system and caused severe damage to the blades. However, at the time of writing, Perseverance remains operational and continues its scientific activities on the Martian surface.

This mission provides a valuable example of multi-robotic cooperation in a GNSS-denied environment. Although the cooperation was only task-based rather than coupling onboard sensor information, the mission demonstrated the practical advantages of combining the heterogeneous capabilities offered by ground and aerial robots.

³<https://www.nasa.gov/>

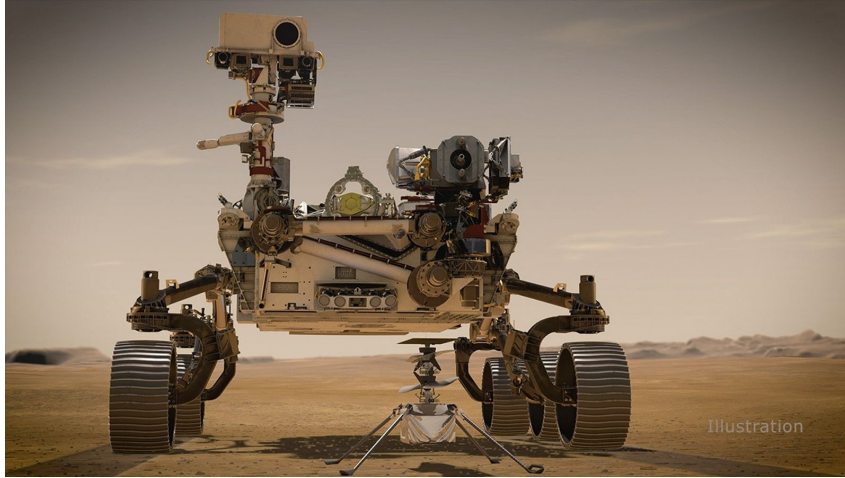


Fig. 1.1 Artist's concept of the Perseverance rover and Ingenuity helicopter on Mars. Credit: NASA/JPL-Caltech (Public Domain).

1.3 UAV-UGV Cooperation from a Localization Perspective

The previous section describes some of the most relevant applications in which the employment of a UAV-UGV robotic framework can introduce significant advantages. However, localization in GNSS-denied environments remains a fundamental challenge. Throughout the discussion, several studies that address this problem have been presented [8–14, 16–22], highlighting the importance of this topic within the scientific community.

Generally, from a localization perspective, the presence of heterogeneous robots operating in the same environment can improve the performance of the overall robotic team in two different ways:

1. by distributing complementary sensing capabilities across the different platforms;
2. by exploiting inter-robot observations and shared information to improve the localization performance of the overall system.

The first advantage arises from the possibility of equipping each robot with a sensor suite tailored to its physical and operational characteristics. Rather than integrating the same set of sensors on every platform, complementary sensors can be

distributed across the robotic team. For instance, a UGV can accommodate heavy and power-demanding equipment, such as a three-dimensional LiDAR, high-capacity batteries, and high-performance computing hardware. Conversely, a UAV can be equipped with a lightweight sensor configuration based on cameras and an IMU, preserving flight endurance, manoeuvrability, and payload capacity. This consideration is even more relevant for small RW-UAVs, which are typically constrained by limited payload capacity and battery energy. For these platforms, even modest increases in sensor mass or power consumption can significantly reduce their flight time and operational flexibility.

The second advantage is related to the possibility of using the measurements collected by one robot to support the localization of another robot. By exchanging relative observations, map information, or state estimates, the robots can introduce additional geometric constraints that are unavailable to an isolated platform. These techniques can be adopted to improve both localization robustness and accuracy, enhancing the performance of the whole system.

1.4 State-of-the-Art of Navigation Solutions in GNSS-Denied Environments

In the literature, several solutions to GNSS-denied navigation have been proposed over the last few decades. Three different surveys discussing the most advanced techniques are provided in [25–27].

GNSS-denied localization techniques can be classified according to the source of the reference information used to estimate the robot pose. A fundamental distinction can be drawn between relative and absolute localization methods. Relative localization estimates the motion of the robot with respect to an initial local reference frame, whereas absolute localization estimates the pose with respect to an externally defined reference frame, such as a pre-existing map or a localization infrastructure [25].

In this thesis, GNSS-denied localization methods are classified into three main categories:

1. **Relative localization and dead-reckoning methods.** These approaches estimate the robot motion by integrating onboard measurements over time. They

include inertial navigation systems, wheel odometry, visual odometry, visual–inertial odometry, LiDAR odometry, and local SLAM systems. Although visual or LiDAR measurements can reduce the drift generated by pure inertial integration, the estimated pose remains expressed in a local reference frame and may progressively diverge over long trajectories.

2. **Infrastructure-referenced localization methods.** These systems rely on artificial references installed in the environment and with known positions. Typical examples include visual markers, UWB anchors, RF beacons, RFID systems, motion-capture cameras, and other fixed external tracking systems. These approaches can provide accurate pose measurements with respect to the infrastructure reference frame, but require deployment, calibration, maintenance, and sufficient coverage of the operating environment.
3. **Map-referenced localization methods.** These approaches estimate the robot pose by matching current sensor observations against a pre-existing map of the environment. Examples include terrain-referenced navigation, terrain contour matching, visual map matching, semantic place recognition, and LiDAR-to-map registration. These methods can provide a pose estimate referenced to the map frame. However, their effectiveness depends on the availability, accuracy, and distinctiveness of the prior map.

The following discussion focuses on three representative families that are particularly relevant to the considered UAV-UGV scenario: Visual–Inertial Odometry, LiDAR–Inertial Odometry, and infrastructure-based localization systems.

1.4.1 Visual Inertial Odometry

Visual–Inertial Odometry (VIO) has emerged over the last two decades as one of the most effective solutions for localization in GNSS-denied environments. VIO algorithms fuse visual information from one or more cameras with inertial measurements provided by an IMU, enabling autonomous platforms to estimate their motion. Due to the low weight and power consumption associated with these solutions, VIO algorithms are extremely suitable for UAVs.

During the last two decades, VIO implementations evolved along two different directions. The first concerns the estimation architecture, which has progressively

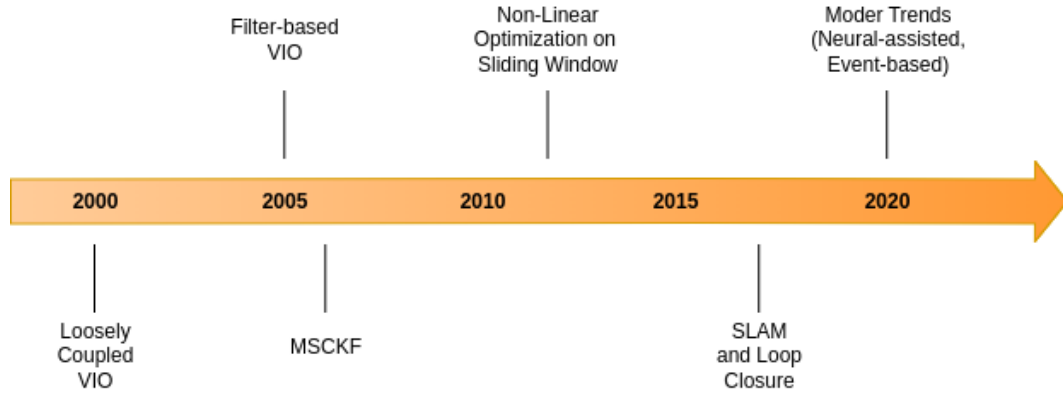


Fig. 1.2 Timeline evolution of Visual Odometry

evolved from loosely coupled filter-based methods to tightly coupled nonlinear optimization. The second direction concerns the visual measurement model, which can be either feature-based (indirect approaches) or photometric (direct approaches). These algorithms typically rely on monocular and stereo configurations, while multi-camera setups have been introduced to overcome typical failures of those systems.

Origins and Problem Formulation

The objective of VIO is to estimate the state of a moving platform equipped with a camera and an IMU. The state vector typically includes the pose of the body with respect to a world reference frame, its velocity, and the inertial sensor biases. While a rigorous derivation of the estimation problem is beyond the scope of this section, a description of the main elements of the formulation is provided.

In a VIO algorithm, the high-rate IMU measurements are employed for state propagation by integrating the rigid-body motion equations. Due to IMU biases and noises, the integration rapidly results in drift accumulation, producing an unreliable estimate. To constrain this drift, VIO algorithms impose geometric constraints between successive poses through the features observed from the camera across consecutive frames. In this context, VIO can be reduced to a sensor fusion problem in which IMU measurements provide short-term motion prediction, while camera images contain information to continuously correct the drift. This sensor fusion can be performed either at the state level (loosely coupled approaches) [28, 29] or directly at the measurement level (tightly coupled approaches) [30–36]

An important aspect concerning VIO algorithms involves the camera configuration. In monocular setups, the absolute metric scale of the environment is not observable from vision alone. The integration of inertial data enables the recovery of scale under sufficient motion excitation, but initialization may require specific maneuvers to ensure observability. Stereo configurations provide metric depth information from the known baseline between cameras, simplifying initialization and improving robustness. In contrast, these setups introduce an increasing computational complexity and hardware integration compared to monocular configurations.

Another relevant distinction concerns the visual front-end. Indirect (feature-based) methods rely on detecting relevant points in the image. These points typically correspond to 3D edges or color change in the scene, and are commonly referred to as *features*. These features are tracked across frames, and by minimizing their reprojection errors, the VIO pipeline is provided with the corrections introduced by IMU propagation. Direct methods, instead, operate on pixel intensities and minimize photometric errors under brightness constancy assumptions. While both formulations can be successfully implemented in a VIO pipeline, feature-based pipelines have historically dominated VIO implementations due to their robustness and structured residual formulation.

Early Loosely Coupled and Filter-Based Approaches

The first visual–inertial systems were often implemented with a loosely coupled approach [28, 29]. In such approaches, visual odometry (VO) and inertial navigation are computed independently, and their outputs are fused at a higher level utilizing a filtering framework. Although they save computational resources, this strategy does not fully exploit the cross-correlation between visual and inertial measurements, resulting in poor performance.

The development of tightly coupled filtering approaches represented a significant improvement in the visual-based navigation field. In these systems, visual feature observations are directly included in the filter update equations. The most common framework utilized in these approaches is the Extended Kalman Filter (EKF).

Multi-State Constraint Kalman Filter (MSCKF) [30] is certainly the most famous implementation based on an EKF. This implementation avoids explicit feature state estimation by imposing multi-view geometric constraints. This approach significantly

reduces computational complexity while maintaining real-time capability, making it particularly suitable for UAV platforms.

Despite their efficiency, EKF-based methods present intrinsic limitations. The reliance on first-order linearization produces inconsistencies, especially during aggressive maneuvers or long trajectories [37]. Moreover, removing the features from the state vector introduces the inability to properly estimate steady conditions.

Sliding-Window Optimization Approaches

To overcome the limitations of filtering frameworks, nonlinear optimization-based methods were introduced. These approaches utilize a sliding window of recent keyframes and optimize the states within the window by minimizing a cost function that includes both inertial and visual residuals. Compared to EKF-based methods, sliding-window optimization introduces an iterative re-linearization to handle nonlinearities. This allows for improved accuracy and consistency, especially during high-speed motion and long trajectories. While showing increased computational cost, the benchmark evaluation in [38] demonstrates that the CPU usage is still comparable to filter-based approaches, remaining convenient for the advantages introduced in the state estimation. Additionally, recently developed powerful platforms can easily afford the computational cost, ensuring real-time performance, especially for well-optimized implementation [30–34, 36, 39, 35].

Among the most popular implementations, OKVIS [31] and VINS-Mono [32] were the first open-source pipelines to achieve remarkable performance for monocular setups. Later, advanced versions of the same algorithms were proposed to include stereo configurations and loop-closure techniques [33] [40]. ROVIO [34] is another remarkable implementation, based on a direct method and achieving great robustness during high-speed motions. Further on, the authors of [36] propose Kimera. This implementation provides high modularity and an efficient SLAM framework, ensuring real-time performance on the CPU. ORB-SLAM [39] is another very popular open-source implementation that is widely used by the community. Basalt [35] is a stereo camera implementation that achieves remarkable performance on public datasets, while resulting in one of the most convenient solutions in terms of CPU usage. More recently, DM-VIO [41] gained high popularity within the community, thanks to the achieved performance over monocular setups.

Both indirect and direct formulations can be implemented in optimization frameworks. However, the majority of VIO systems are based on direct approaches because of the more natural integration of reprojection residuals with inertial constraints. Although direct methods remain potentially more informative in textureless environments, these approaches introduce results that are very sensitive to illumination changes.

From Odometry to SLAM and Loop Closure

Since classical VIO systems operate as odometry estimators, these algorithms tend to accumulate significant drift over the trajectories. This undesirable effect is emphasized by calibration inaccuracies and low-texture environments. To reduce this drift, VIO is typically joined with Simultaneous Localization and Mapping (SLAM) frameworks. When revisiting previously explored zones, SLAM frameworks introduce global constraints into the pose graph or bundle adjustment optimization. This process is typically referred to as loop-closure detection and can significantly reduce the drift accumulated over long trajectories [42, 36, 33]. However, SLAM-based approaches require additional computational overhead and increase system complexity. Therefore, especially on UAV platforms, pure VIO algorithms remain more convenient, and other strategies based on cooperative localization or additional sensor integrations are preferred to reduce the drift.

1.4.2 LiDAR–Inertial Odometry

LiDAR–Inertial Odometry (LIO) is a well-established navigation solution for navigation in GNSS-denied environments, particularly for ground vehicles and larger aerial platforms. By fusing accurate 3D data collected from LiDAR readings with high-frequency inertial measurements, LIO systems provide extremely robust and low-drift state estimation, even under challenging illumination and environmental conditions.

Origins: Scan Matching and Pure LiDAR Odometry

Early LiDAR-based navigation solutions relied primarily on geometric scan matching techniques to estimate motion between consecutive point clouds. One of the earliest

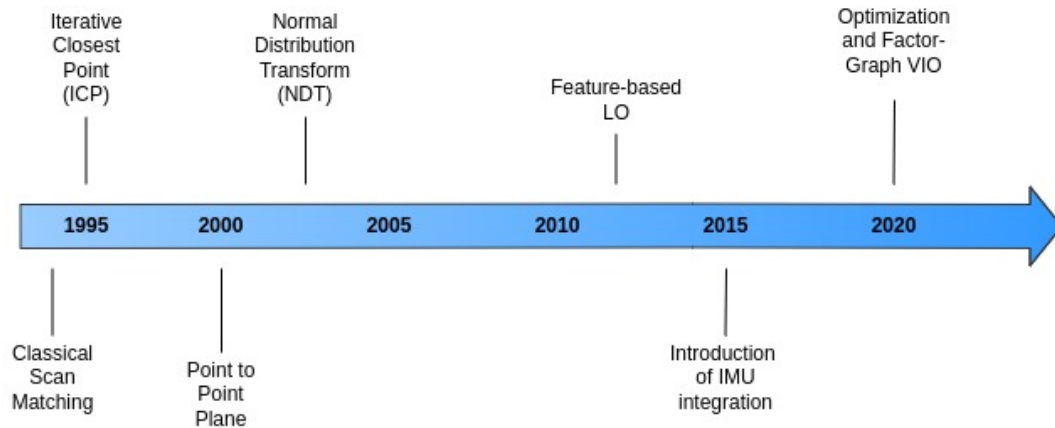


Fig. 1.3 Timeline evolution of LiDAR-based odometry

and most influential methods is the Iterative Closest Point (ICP) algorithm [43], which estimates the rigid-body transformation between two point sets by minimizing a point-to-point alignment error. Later, variants of this approach were proposed. For example, point-to-plane ICP [44] consists of a methodology to improve convergence properties and accuracy.

Another widely adopted formulation is the Normal Distributions Transform (NDT) [45]. This approach relies on representing the environment using local Gaussian distributions and performs a probabilistic scan alignment.

A remarkable implementation that acquired high popularity within the community was Lidar Odometry and Mapping (LOAM) [46]. The work introduces feature-based scan matching by extracting edge and planar features from each scan and performing separate high-rate odometry and low-rate mapping optimization. Although LOAM does not utilize inertial measurements, this implementation achieves real-time and high-accuracy performance, demonstrating the effectiveness of LiDAR-based navigation.

However, pure LiDAR odometry (LO) presents several intrinsic limitations that have been widely analyzed in the literature.

- **Motion distortion due to sequential point acquisition.** LiDAR sensors acquire points sequentially by mechanically spinning. Typically, the acquisition takes from 50 to 100 ms for each scan. When the platform experiences motion during this interval, the resulting point cloud is geometrically distorted, as each point corresponds to a different sensor pose. This inconsistency considerably

degrades performance, leading to high position errors and solution divergence. LIO systems such as FAST-LIO [47] and LINS [48] address this issue by compensating the position error utilizing IMU measurements.

- **Sensitivity to degenerate geometric environments.** Since LiDAR odometry relies on geometric constraints derived from planar or edge features, its performance is degraded in environments with insufficient structural diversity (long corridors, open fields, tunnels). This issue has been analyzed in scan matching literature [45] and in several LiDAR-SLAM systems, including LOAM [46] and LeGO-LOAM [49].
- **Limited robustness during aggressive motion.** Feature tracking in LiDAR odometry systems assumes relatively small inter-frame motion. This assumption fails under high-speed or highly dynamic motion, leading to the divergence of the solution. The integration of inertial measurements significantly improves robustness in such scenarios, as demonstrated by [50] [47].

Introduction of Inertial Fusion

The fusion of LiDAR and IMU measurements significantly improved the robustness and accuracy of LiDAR-based odometry solutions. As in the case of VIO, even for LIO, tightly coupled integration between LiDAR readings and IMU measurements rapidly became the most adopted solution.

One of the first tightly coupled formulations is presented in [50], where inertial measurements are used to compensate for motion distortion and improve 3D reconstruction. Further on, the authors in [51] explore a solution based on an error-state Kalman filter.

In LIO systems, the state typically includes pose, velocity, and IMU biases. The IMU provides high-frequency state propagation, while LiDAR measurements impose geometric constraints derived from scan-to-scan readings. Compared to visual systems, LiDAR directly provides metric depth, making scale inherently observable without requiring motion excitation.

As previously discussed, a central technical challenge concerns motion distortion. In the literature, this issue is typically referred to as scan skew. IMU-based *deskewing* refers to the technique that compensates for this distortion. During the scan, the

point cloud is transformed according to the IMU propagation, compensating for the errors introduced by the movements that occurred within two consecutive scans. The IMU-based deskewing becomes a standard component of modern LIO pipelines, such as [47].

Optimization-Based State-of-the-Art Methods

Modern LIO systems predominantly rely on tightly coupled nonlinear optimization or iterated filtering frameworks.

LIO-SAM [52] formulates the estimation problem as factor graph optimization, integrating IMU preintegration theory with LiDAR feature-based residuals.

FAST-LIO [47] introduced a computationally efficient iterated error-state Kalman filter (IESKF) that directly includes raw point cloud measurements. FAST-LIO2 [53] further improves performance by implementing direct scan-to-map registration without explicit feature extraction, reducing computational cost and increasing robustness.

Other representative LIO systems are:

- LeGO-LOAM [49], optimized for ground vehicles,
- LINS [48], a LiDAR–Inertial state estimator using smoothing,
- R2LIVE [54], which integrates LiDAR, vision, and inertial sensing.

Loop Closure and Global Mapping

As with visual odometry, LIO accumulates drift over long trajectories. To mitigate this problem, many systems are provided with loop closure detection and pose graph optimization. Among these implementations, LIO-SAM [52] gained great popularity within the community.

Since LiDAR-based loop closure often benefits from more stable geometric descriptors compared to visual methods, the SLAM approach for LIO systems provides high reliability. However, false-positive detection may still occur in geometrically repetitive scenes, such as long corridors or areas with limited structural variation.

1.4.3 Infrastructure-Based Localization Systems

Infrastructure-based localization systems provide absolute positioning by employing external reference sources deployed in the environment. Unlike odometry approaches such as VIO or LIO, infrastructure-based systems offer global position estimates. For this reason, they are particularly adopted in structured environments such as warehouses, industrial facilities, and underground sites.

The working principle of these localization systems is conceptually analogous to GNSS. A set of transmitters or devices (anchors) is placed at known positions within the environment, while the mobile platform is equipped with a second set of devices. By measuring quantities such as signal propagation time, signal strength, or relative bearing, an algorithm can estimate the position of the mobile platform with high accuracy. The estimation phase can be performed by trilateration, multilateration, triangulation, or optimization-based sensor fusion.

Infrastructure-based systems can be distinguished into motion capture systems, radio-frequency ranging systems (generally known as Ultra-Wideband), and WiFi/BLE fingerprinting approaches.

Motion Capture Systems

Motion capture systems represent the most accurate infrastructure-based solution currently available. Commercial systems such as Vicon and OptiTrack rely on multiple calibrated infrared cameras distributed around the environment. Thanks to a set of reflective markers attached to the robot, the position estimate is obtained by multi-view triangulation.

These systems achieve millimeter-level position accuracy at high rates (100–200 Hz). As a result, they are widely used in robotics research for benchmarking and ground-truth acquisition. Since their installation requires a controlled indoor environment, precise calibration, and high cost, these systems are rarely utilized in real applications. Additionally, in real-world scenarios, motion-capture systems would suffer from occlusions, drastically reducing their reliability and accuracy. In contrast, these systems provide a valuable solution for benchmark comparisons and performance evaluation in the research field.

Ultra-Wideband (UWB) Anchor-Based Systems

Ultra-Wideband (UWB) systems have emerged as one of the most practical infrastructure-based solutions for robotics applications. UWB relies on short-duration radio pulses with large bandwidth, enabling precise Time-of-Flight (ToF) estimation and decimeter-level ranging accuracy. In a UWB setup, anchor nodes are placed at known positions, while a mobile tag is attached to the robot. By analyzing the signals received from the tags, distance estimates are obtained through two-way ranging or time-difference-of-arrival strategies, and the robot position is computed, typically via trilateration or fused within a state estimator. A more detailed description of the UWB working principle and strategies for distance estimation is provided in [55, 56].

Compared to WiFi or Bluetooth-based systems, UWB is significantly more robust to multipath interference, providing superior accuracy when occlusions occur. Several papers propose solutions based on this technology for GNSS-denied environment localization [57–60].

Advantages of UWB systems include:

- Drift-free absolute positioning,
- Lightweight onboard hardware suitable for UAVs,
- Decimeter-level accuracy in indoor environments.

However, infrastructure deployment is required, and performance may degrade in metallic or cluttered environments. Range is typically limited to tens of meters indoors.

WiFi and BLE Fingerprinting Systems

WiFi and Bluetooth Low Energy (BLE) beacon-based localization systems exploit received signal strength (RSSI) measurements or channel state information to estimate position. For these solutions, two main approaches are used: model-based ranging and fingerprinting. In fingerprinting methods, a radio map is constructed offline by recording signal characteristics at known positions. During operation, online measurements are matched to the pre-built map to infer location.

These systems are attractive due to their low cost and the possibility of utilizing existing infrastructure. However, accuracy is generally around meter-level precision, and performance is highly dependent on environmental conditions. Additionally, signal propagation variability, occlusions, and environmental changes may significantly degrade reliability.

A comprehensive survey on WiFi localization techniques is provided in [61].

1.4.4 A Qualitative Comparison of the Presented Solutions

The localization solutions examined in the previous paragraphs have reached a high level of technological maturity, experiencing significant improvements over the last few years. While achieving promising performance, these localization systems show relevant limitations that currently reduce their applicability. VIO frameworks are extremely convenient due to their installation simplicity, low weight, and reduced power consumption, being particularly suitable for RW-UAV platforms. However, monocular setups show inherent limitations that considerably reduce their applicability in real-world scenarios [62, 63]. For this reason, stereo setups are the most common configurations, at the cost of increasing computational resources and installation complexity. Nevertheless, these configurations exhibit poor performance in low-texture scenarios and are very sensitive to brightness variations and motion blur. Additionally, VIO systems exhibit consistent drift, and image processing and solver implementation require appropriate optimizations to ensure real-time performance on onboard computers.

LIO algorithms show greater accuracy and robustness compared to VIO. Particularly, 3D LiDARs represent the most comprehensive solutions, ensuring real-time performance, reduced positioning errors, and high robustness in many scenarios. Additionally, these sensors provide the data for an accurate 3D reconstruction of the surrounding environment; thus, they can fulfill obstacle avoidance tasks. While being the most comprehensive solutions for UGV platforms, 3D LiDARs are sub-optimal for RW-UAVs. Due to their high weight and dimensions, these sensors are not suitable for UAV platforms that require low weight and a small size. Additionally, these sensors exhibit high power consumption, which would drastically reduce the flight endurance of drones.

Infrastructure-based localization is the most convenient solution to achieve extremely accurate positioning. In scenarios where centimeter-level accuracy is required, these systems offer a valuable solution, eliminating the drift accumulation that affects VIO and LIO algorithms. Moreover, the algorithms involved in pose estimation are typically lighter compared to VIO and LIO implementations. Since the installation of motion capture systems is unfeasible in real-world scenarios where occlusions frequently occur, UWB-based solutions are the most convenient in terms of accuracy and reliability. However, a full-system installation might be expensive for large environments. At the same time, even if the tags attached to the mobile platform are cheap and light, they require multiple installations to guarantee high precision in real-world scenarios.

A qualitative comparison of these localization solutions is provided in Figure 1.4. For this analysis, among the infrastructure-based solutions, the UWB system is considered, as this technology guarantees better performance in real-world scenarios compared to its alternatives. For each performance metric, a score is assigned: a score of 1 indicates the least favorable option, whereas a score of 5 shows the most favorable solution. The score is mostly assigned according to the knowledge of the author and the experimental evidence collected through this work. As for the accuracy, VIO shows the worst performance due to the large drift accumulated over long trajectories. In the literature, VIO accuracy is typically around 1–3% of the trajectory length, whereas LIO shows reduced errors around 0.5%. Although UWB systems do not accumulate drift, the accuracy heavily depends on signal trilateration quality and the number of occlusions in the signal path. The robustness is intended as the capability of the localization system to face adverse conditions (high-speed motions, brightness variations, occlusions). In these scenarios, LIO is certainly the best solution, thanks to the extremely accurate and extensive data that these sensors can collect from the environment. Even if VIO shows relevant limitations in such complex scenarios, UWB systems are considered worse, as multipath might cause a total absence of localization. In contrast, UWB-based methods are the best choice for computational and energy efficiency. The trilateration algorithm for pose estimation is extremely light compared to VIO and LIO pipelines. In particular, as LIO algorithms typically collect more data than VIO frameworks, LiDAR-based solutions are considered to be more demanding in terms of computational costs. Additionally, LiDAR sensors consume a considerable amount of energy, being the worst solution in terms of energy efficiency. Finally, deployment flexibility is another

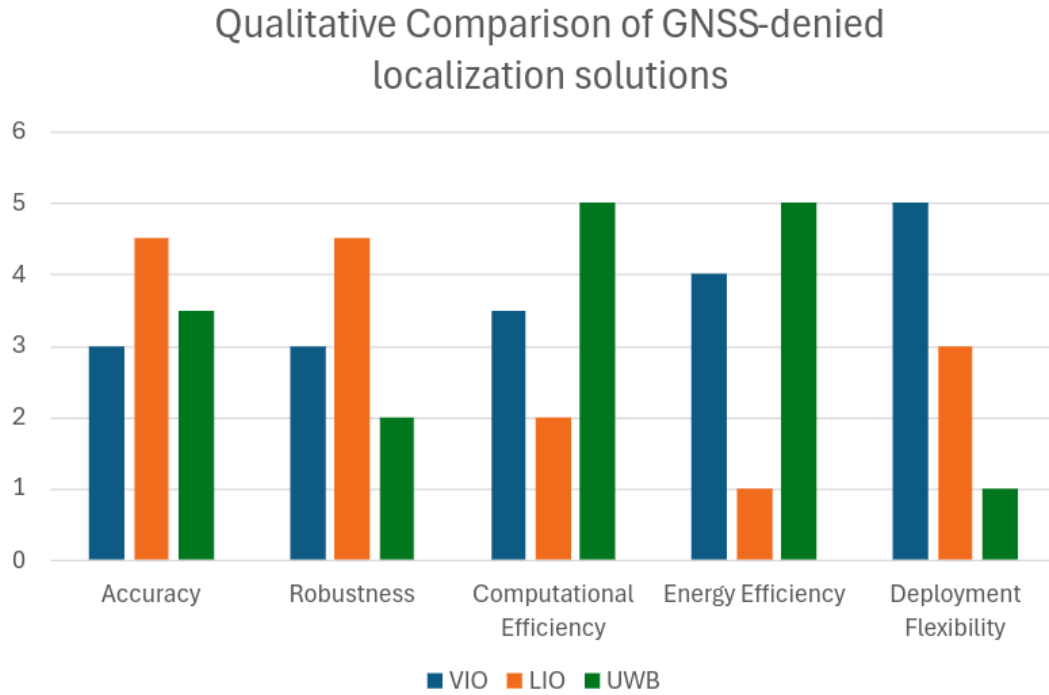


Fig. 1.4 The diagram qualitatively compares the three localization solutions across key performance metrics. Scores range from 1 (least favorable) to 5 (most favorable).

important aspect for a comprehensive evaluation, and VIO-based localization systems are the most convenient solutions in these terms. In contrast, 3D LiDAR sensors are typically heavy, and their dimensions constrain the installation, especially for RW-UAV platforms. Nevertheless, UWB systems exhibit the worst performance, as they require external installations.

1.5 Focus and Scope of the Work

The discussion over autonomous multi-robot cooperation in GNSS-denied environments provided in 1.2 identifies localization as one of the main factors determining the effectiveness and reliability of the overall robotic system. In cooperative UAV-UGV missions, each platform must estimate its pose accurately enough to safely execute its assigned task, share meaningful spatial information with the other agents, and maintain a coherent representation of the surrounding environment. Localization errors may compromise navigation safety, task execution, map consistency, and the effectiveness of inter-robot cooperation.

Due to their capabilities, RW-UAVs are widely adopted in the applications considered in this thesis. Their relevance in GNSS-denied environments, together with the need to preserve flight endurance and compact dimensions, motivates the research carried out in this dissertation. Without loss of generality, the methodologies proposed in this thesis have been developed with small RW-UAVs as the target platform, since these vehicles are particularly constrained in terms of payload, power consumption, and onboard computational resources. Nevertheless, the proposed approaches are not limited to this specific class of aerial robots and can be extended to other robotic platforms relying on visual-inertial navigation.

During the last decades, substantial progress has been achieved in GNSS-denied navigation. However, the comparison reported in Section 1.4.4 shows that no universally optimal solution is currently available, especially for RW-UAVs. LiDAR-based and LiDAR-inertial localization systems can provide accurate and robust three-dimensional navigation in a wide range of environments. Nevertheless, the integration of a LiDAR sensor onboard a small aerial platform generally increases payload weight, physical dimensions, power consumption, and system cost. These requirements may reduce flight endurance, limit manoeuvrability, or require the adoption of a larger and more expensive UAV platform.

Conversely, cameras and IMUs provide an attractive sensor configuration for lightweight aerial robots. Visual-inertial odometry can estimate the UAV motion using sensors with limited weight and power requirements, making it particularly suitable for small rotary-wing platforms. This aspect becomes even more relevant in applications where cameras are required independently of navigation. In such cases, exploiting the same visual sensor for both mission-related perception and state estimation can substantially reduce the overall sensing payload.

However, despite the advantages of visual sensing, current VIO systems still exhibit important limitations that reduce their reliability in real-world UAV missions. In particular, visual tracking can degrade during high-speed motions, motion blur, illumination changes, and observations of low-texture or repetitive environments. Moreover, VIO remains a relative localization method and can accumulate drift over long trajectories. These limitations are particularly critical for UAVs, whose flight safety and autonomous operation depend on the continuous availability of a reliable pose estimate.

The primary objective of this dissertation is therefore to improve the robustness and long-term accuracy of camera-based UAV localization in GNSS-denied environments, while preserving the low payload, power consumption, and hardware simplicity that make visual sensing attractive for aerial platforms. The work focuses on visual–inertial localization systems and investigates how their limitations can be mitigated through algorithmic improvements, lightweight sensor fusion, and cooperation with heterogeneous robotic platforms.

1.5.1 The Importance of Cameras for Mission Requirements

In many GNSS-denied applications involving aerial robots, perception sensors are required not only to support navigation but also to accomplish the primary mission objectives. While both cameras and LiDAR sensors can be employed to estimate the vehicle state and enable autonomous navigation, their contribution to the mission payload is often substantially different. In some scenarios, cameras frequently represent an indispensable sensing modality because the information required by the mission itself is inherently visual. Examples include infrastructure inspection, warehouse inventory, and all the applications in which object detection is mandatory. In these scenarios, the collected images constitute the final output of the mission and are therefore necessary regardless of the adopted navigation strategy.

On the contrary, the role of LiDAR sensors is often limited to supporting navigation and obstacle avoidance. Although LiDAR measurements can also be exploited for three-dimensional reconstruction and geometric mapping, such information is not always required by the mission objectives. As a consequence, for the applications in which camera payload is already mandatory, the addition of a LiDAR sensor introduces an extra sensing modality whose benefits may be confined to navigation-related tasks.

During system design, every additional sensor increases the overall weight and power requirements, potentially reducing mission duration or requiring a larger and more expensive platform. Consequently, whenever a camera is already available to satisfy mission requirements, it becomes attractive to exploit the same sensor for navigation purposes as well. Besides the localization purposes, cameras can also be employed to perform obstacle avoidance tasks, providing the depth of triangulated points across the frames. Although the quality of the environment reconstruction is

not comparable to a LiDAR-based 3D map, checks for free paths during navigation can still be performed with high reliability.

Additionally, in the context of multi-robotic frameworks, the heterogeneity of the robots involved leads to equipping each platform with a minimal sensor configuration, as a specific task required by the mission can still be accomplished by the robot with a dedicated sensor setup. For instance, whenever a 3D reconstruction is required, the formation may include LiDAR-equipped robots, while maintaining other agents with a camera-based setup.

1.5.2 The Topics Addressed

Although VIO algorithms have experienced significant improvements over the last decades (see the timeline in Figure 1.2), open challenges still exist, effectively reducing the applications of these systems. The primary objective of this dissertation is to improve the robustness and long-term accuracy of vision-based UAV localization in GNSS-denied environments while preserving the low payload and power requirements that make visual sensing particularly attractive for aerial platforms. From both the literature review and the experimental evidence collected throughout this work, the dissertation identifies the following aspects as the most crucial for VIO algorithms:

1. Lack of robustness in challenging scenarios (high-speed motion, low-texture environments, brightness changes)
2. Large drift accumulated over long trajectories
3. High computational complexity of best-performing solutions

This dissertation focuses primarily on the first two aspects. Although computational efficiency remains an important research topic, especially for small aerial platforms, it is not directly addressed as a primary contribution of this dissertation. Conversely, the proposed work aims to improve the robustness of visual-inertial localization during challenging operating conditions and to reduce the drift accumulated by UAVs over time.

The methodologies proposed in this thesis are implemented within selected state-of-the-art VIO pipelines. The theoretical assumptions, implementation choices, and

practical limitations of these pipelines are analyzed in order to identify conditions in which the localization performance degrades. Novel methodologies and sensor fusion strategies are then introduced to mitigate these limitations while preserving the basic advantages of camera-based UAV localization.

Moreover, sensor fusion represents a central element of the proposed approach. Under the assumption that substantial increases in payload weight and power consumption should be avoided, the developed solutions combine visual-inertial estimates with information provided by lightweight onboard sensors, such as downward-facing range sensors. Additionally, in a multi-robotic context, the work extends sensor fusion to sensors installed on cooperating robots. In particular, a methodology is introduced to reduce UAV localization drift by combining UAV detections from the UGV camera images with 3D LiDAR point cloud data acquired by the UGV.

1.6 Chapters Overview

Chapter 2 discusses in detail the open challenges concerning visual-based navigation presented in the previous section. The chapter introduces the state-of-the-art solutions to mitigate typical failures of VIO pipelines.

Chapter 3 contains three methodologies to improve the robustness of state-of-the-art VIO pipelines, according to the limitations identified in Chapter 2. In particular, typical failures of monocular setups are addressed, and the problem of insufficient feature tracking is mitigated through two different approaches.

Chapter 4 focuses on sensor fusion for VIO drift reduction. To provide visual-based localization with enhanced global consistency, two sensor fusion algorithms are proposed. The first solution is based on the elaboration of raw readings of a downward-facing range finder employed as an altitude sensor. The second algorithm combines UAV detections from the UGV camera images with 3D LiDAR point-cloud data acquired by a UGV.

Chapter 5 contains the conclusions of the work, and considerations about future works are provided.

Chapter 2

Open Challenges of VIO

In Chapter 1, an overview of state-of-the-art GNSS-denied localization systems in the robotic field has been provided. Since this chapter aims to discuss the current limitations of VIO systems, a more extensive overview focused on visual-based navigation systems is presented in this section.

The evolution from loosely coupled and filter-based approaches towards sliding-window optimization frameworks has significantly improved the accuracy, consistency, and real-time applicability of VIO algorithms. Nevertheless, despite the improved performance achieved by modern implementations, several fundamental limitations remain. These limitations are related both to the observability properties of monocular visual-inertial systems and to the robustness of visual feature tracking in challenging conditions.

As emerged from the previous section, modern VIO algorithms can achieve accurate state estimation under favourable visual and motion conditions. However, initialization requirements, visual degeneracy in low-texture environments, limited camera field of view, and the computational cost of sliding window algorithms remain open challenges. In particular, four topics emerge as the most relevant open challenges:

- The lack of robustness for monocular setups
- The failures of visual-based localization under insufficient feature tracking
- The large drift accumulation over long trajectories

- The high computational load required by advanced algorithms or multi-camera setups

This chapter examines each of these aspects and reviews the most relevant studies addressing the associated issues.

2.1 Mono-Camera Setups

Monocular setups represent the minimal configuration to run a VIO algorithm. The simplicity of this configuration is particularly convenient in applications where the UAV needs to be light and inexpensive. Thanks to their reduced computational cost compared to stereo or multi-camera setups, monocular VIO algorithms are suitable for several boards, offering the possibility to reduce the size and cost of a UAV. Although early visual-motion-estimation systems investigated both monocular and stereo camera configurations, the proposed architectures for state estimation were substantially identical. The simplest approach to implementing VIO algorithms consists of loosely coupled sensor fusion [28, 29]. In these frameworks, vision acquisitions and IMU measurements are treated independently and then coupled typically by an EKF. Nevertheless, the lack of robustness and the poor stability in scale estimation motivated researchers to investigate other solutions. SLAM implementations in [64–67] provide feasible solutions in terms of accuracy and robustness, with the primary objective of reducing the drift experienced by these systems. Since these algorithms constrain the camera position to the observed landmarks, they require the introduction of the 3D position of the observed features in the state vector, resulting in high computational cost. At the time these algorithms were proposed, many onboard computers could not process the pipeline in real time, motivating the development of less computationally demanding approaches. In [30], the authors propose the well-known Multi-State Constraint Kalman Filter (MSCKF). The main contribution of the work is the derivation of a measurement model that expresses geometric constraints across multiple camera poses without including the 3D position of the features in the state vector. By reducing the complexity of the problem, MSCKF is lightweight compared to SLAM approaches and ensures real-time processing for the boards of that period. An improved version of the algorithm has been proposed in [37]. This work proposes a Jacobian computation that

preserves the consistency of the filter, ensuring the correct observability properties and achieving improved performance compared to the first version of the algorithm.

Further on, sliding window optimization approaches were introduced. These approaches optimize the state vector over a bounded-size sliding window, marginalizing old states to keep the size of the problem fixed. Compared to SLAM approaches and MSCKF, these methods do not suffer from the linearization inconsistency of filter-based algorithms and optimize only the 3D landmarks observed in the current sliding window. OKVIS [31] is one of the first implementations of sliding window optimization. Another very popular open-source implementation is VINS-Mono [32]. The authors in [68] propose a new methodology consisting of IMU pre-integration. This approach has been released in the GTSAM 4.0 optimization toolbox [69]. ROVIO [34] is another VIO implementation that is based on a direct method approach. This implementation achieves remarkable performance under aggressive motion, thanks to the use of direct methods for optical flow generation.

The open-source algorithms above have been compared in [38]. The benchmark comparison confirms the superiority of optimization-based algorithms over filter-based approaches, evaluating their performance and computational load over different platforms. However, improvements in estimator formulation alone cannot solve the intrinsic degeneracies of the visual-inertial estimation problem.

Despite the numerous advantages in terms of installation simplicity, required computational resources, lightweight, and size, monocular setups pose several issues that reduce their applicability in real-world scenarios. From both literature analysis and experimental evaluations conducted through this work, two failure scenarios - strictly related to monocular setups - have been identified in this dissertation:

1. Initialization phase
2. Pure rotations

For clarity, the initialization phase refers to the time interval between the start of the pipeline and the first sufficiently informative motion, which provides the baseline required to triangulate image features. In contrast, pure rotations refer to motions in which the system experiences negligible or no translational displacement. In both scenarios, monocular algorithms exhibit poor performance due to the inherent limitation of these setups to determine the scale of the problem in the absence of

sufficient relative motion across frames. Although the two failure scenarios both refer to a common issue, the distinction is necessary as the two issues can be addressed by different methodologies. In particular, the problem arises from the structure of VIO pipelines, which require an estimate of the 3D position of image features to run the optimization process. Since for 3D feature positioning a baseline is required, a monocular setup necessarily needs to experience motion across frames to acquire this baseline. This issue strictly depends on the monocular setup, as stereo or multi-camera setups sharing portions of the field of view can recover the system scale by utilizing the fixed baseline provided by cameras placed at a known distance. Although IMU data can be included in the pipeline to provide information about the motion scale, situations in which the accelerometer does not provide useful data may occur, such as during pure rotation or constant velocity motions. Therefore, a monocular setup might experience serious failures during these motions, showing poor performance in real-world scenarios.

Some studies underline the limitations of monocular VIO algorithms from a mathematical perspective [62], [63]. These papers analytically demonstrate that yaw and global position are always unobservable for a monocular setup. During initialization and pure rotational motions, the problem is ill-posed due to the absence of parallax and translational excitation. This implies that even translation and scale become unobservable. To deal with the initialization phase, several algorithms leverage the contents of [70] to recover relative rotation and translation up to an unknown scale factor through a vision-only Structure from Motion (SfM). In [71], the authors propose a methodology to fuse inertial data with SfM to recover the scale and address the initialization problem. The authors of [72] introduce a solution based on SfM and inertial measurement fusion that outperforms similar solutions in computation time. The same procedure is utilized in the VINS-Mono open-source implementation. The main advantage of SfM and IMU fusion is that they operate without any assumptions on the initial state or the surrounding environment. However, these methods rely on a sufficient baseline between consecutive camera frames and sufficient excitation of the inertial sensors. Additionally, during initialization, the estimate may experience a delay while waiting for enough baseline or adequate IMU excitation. In the context of UAV applications, such requirements can be restrictive. UAV flight stability is highly sensitive to delays during the initialization phase, and even a short delay of a few seconds may compromise the flight. As a result, these initialization strategies are feasible only under specific motion conditions. Although

the authors of [32] and [72] report strong initialization performance, their evaluations on UAVs are conducted offline over public datasets, and the UAV does not rely on VIO estimates for real-time attitude and position control during the whole flight. These limitations concerning monocular setups explain why stereo cameras are the most popular solutions when robustness and reliability are required.

2.2 Insufficient Feature Tracking

Visual-inertial algorithms rely on visual measurements to constrain the drift generated by IMU propagation. In these implementations, such measurements are obtained by detecting and tracking image features across consecutive frames. Therefore, the availability, spatial distribution, and tracking quality of visual features strongly affect the accuracy and robustness of the state estimate.

Large uniform surfaces, poorly illuminated areas, and scenes affected by motion blur may provide an insufficient number of distinguishable features. Additionally, when a limited number of features are detected, their geometric configuration may be unsuitable to properly constrain the pose estimate. For example, features concentrated in a small image region or located at a similar depth may provide limited information about the platform motion. Under these conditions, the visual update becomes weak or unavailable, and the estimator increasingly relies on IMU propagation. Since inertial integration is affected by noise and bias accumulation, the position and attitude estimates rapidly degrade.

Several approaches have been proposed to improve robustness under challenging visual conditions. In [73], the authors propose a VIO algorithm based on line features. By merging the results coming from semantic segmentation, optical flow, and reprojection error, their approach reduces the influence of dynamic objects. To enhance the performance of a VIO algorithm encountering different brightness conditions, the authors of [74] propose a methodology that couples a model-based Left Invariant Extended Kalman Filter (LIEKF) with a statistical neural network, both driven by raw inertial measurements. While these approaches mitigate common failures of VIO algorithms, textureless environments are excluded from these treatments.

The most intuitive method to address featureless environments is to increase the field of view of the cameras involved in localization tasks. Covering a wider field

of view, multi-camera configurations can reduce the probability that all cameras simultaneously observe textureless regions.

One of the first papers that proposes a multi-camera-based VIO algorithm is [75]. In this approach, the two stereo cameras face opposite directions, and the experimental results indicate superior performance over state-of-the-art stereo camera solutions. However, the implementation relies on an Extended Kalman Filter (EKF) for pose estimation rather than modern optimization techniques such as bundle adjustment or factor graphs, which achieve finer performance. A generalized multi-camera implementation is introduced by the authors in [76]. The implementation supports an arbitrary number of cameras in various configurations, including overlapping and non-overlapping fields of view. Similarly, the authors of [77] propose an alternative approach and compare it with [76], demonstrating the superiority of their solution on custom datasets. Another multi-camera approach is presented in [78]. The work examines a setup consisting of a stereo camera with two additional monocular cameras mounted on the sides to increase the field of view of the overall system. The implementation optimizes computational performance by tracking the features detected from the stereo camera on the monocular cameras on the sides. The authors of [79] propose a methodology to specifically handle setups with non-overlapping fields of view, whereas in [80] the authors provide an implementation tailored for UAVs.

A key aspect concerning multi-camera implementations relates to computational load. Since these pipelines require feature extraction and tracking over more than two cameras, they result in being highly demanding in terms of computational resources. Among the works above, only [78] examines this aspect, proposing a solution that accounts for a reduction in computational load. The paper [81] analyzes the computational load for a multi-camera setup, comparing the proposed algorithm to ORB-SLAM 3. The authors report the CPU loads for different multi-camera configurations, showing that the computational load is never below 300% for setups involving more than two cameras. In [82], the authors provide an implementation of a multi-camera setup, assuming no overlapping regions among the cameras. Thus, the reported CPU loads do not include feature tracking across cameras, which significantly impacts the computational effort.

Although multi-camera setups cannot guarantee robust localization in environments that are globally poor in visual information, the wider field of view they

provide can significantly improve localization accuracy and reliability. Nevertheless, these setups introduce additional hardware and computational requirements, requiring further installations and powerful boards for real-time processing.

A further approach to improve robustness in featureless environments consists of learning relative motion directly from IMU measurements through neural networks. RNIN-VIO [83] integrates a learned inertial displacement constraint with conventional IMU propagation and visual feature measurements, improving robustness when visual tracking is weak or unavailable. The paper proposes a tightly coupled VIO implementation in which a neural network estimates relative displacement and its uncertainty from a temporal window of IMU measurements. This estimate is incorporated as an additional measurement constraint within an EKF-based filter, alongside IMU propagation and visual feature measurements extracted from the images. As a result, the neural constraint helps compensate for the drift accumulated by conventional inertial integration and improves robustness when visual measurements are weak or temporarily unavailable. However, the proposed network is trained and evaluated mainly on handheld and pedestrian motions. The authors explicitly identify its limited generalization to unseen motion patterns, including drone motion, as an open issue. A recent extension of this concept to aerial platforms is proposed in [84], where a neural network estimates relative displacement and uncertainty directly from raw IMU measurements. These predictions are integrated as additional constraints within the VINS-Fusion optimization framework, together with visual reprojection and IMU pre-integration residuals. The approach is evaluated on high-speed drone sequences and demonstrates improved robustness during aggressive maneuvers and simulated visual blackouts. Nevertheless, the network is trained on a specific UAV dataset, and its generalization across different aerial platforms, sensor configurations, and flight dynamics remains an open challenge.

2.3 Global Drift Accumulation

The term global drift accumulation refers to the progressive divergence of the estimated trajectory from the real trajectory over long missions. Although modern VIO algorithms can provide accurate local motion estimates, they operate primarily as odometry systems. As a consequence, small residual errors in IMU bias estimation,

camera–IMU calibration, feature tracking, and visual measurement processing are accumulated over time.

Sliding-window optimization methods reduce the computational complexity of the estimation problem by marginalizing old states and retaining only a bounded set of recent keyframes. While this strategy allows real-time operation, it also limits the ability of the estimator to directly correct errors that occurred far in the past. Therefore, even when the local trajectory remains smooth and internally coherent, its global position and orientation may progressively drift from the real values.

Map alignment methods [85–87] rely on loop-closure detection across the map built during navigation. These approaches effectively reduce the accumulated drift, enabling consistency over long trajectories introduced by dead-reckoning algorithms. Similar approaches are implemented in popular SLAM VIO pipelines to improve the global consistency of the reconstructed trajectory [42, 36, 39]. However, loop closure requires the vehicle to revisit visually recognizable locations and to maintain a sufficiently informative map of the environment. These requirements cannot be guaranteed during exploratory missions, long one-way trajectories, or operations in repetitive and dynamically changing environments. The approaches in [88] and [89] propose methods to relatively localize robots by fusing fiducial marker detections coming from multiple robots. While requiring low computational resources, these methods are very sensitive to marker visibility and visual conditions, since these factors may compromise the detection and cause false positives. These methods also require installations on each robot or, even worse, on the surrounding environment, reducing their operability and feasibility.

Approaches that utilize UWB technology are also popular to reduce the drift of vision-based algorithms. In particular, some solutions apply them to cooperative robot formations. In [90], the authors propose a methodology to enhance the localization of two UAVs by fusing the data coming from UWB sensors and VIO, whereas in [91], the authors show a similar approach but extended to UAV swarms.

2.4 Computational complexity

As concerns the aspect related to computational complexity, although playing a crucial role in the choice of navigation systems, several boards are currently capable of running best-performance VIO algorithms in real-time.

Although VIO pipelines, especially when involving multiple cameras, may require substantial computational resources, this does not prevent their real-time execution on current embedded platforms. Modern heterogeneous computing boards can execute state-of-the-art VIO algorithms in real time, making them suitable for robotic and UAV applications. However, general-purpose embedded platforms are commonly designed to provide a broad set of interfaces, memory resources, and processing capabilities. Consequently, their overall dimensions, power requirements, and thermal management needs may be unsuitable for small aerial platforms.

This limitation is therefore not exclusively related to the computational complexity of VIO but also to the availability of compact general-purpose hardware platforms. A custom hardware design, jointly optimized for the selected sensors and the target estimation pipeline, can significantly reduce the footprint of the overall localization system while preserving real-time operation. By removing unnecessary peripherals and integrating dedicated processing hardware, a compact visual-inertial sensing module can be obtained.

The Intel RealSense T265¹ represents a relevant example of this design approach. The device integrated two fisheye cameras, an IMU, and an Intel Movidius Myriad 2 visual processing unit within a compact module. The visual-inertial SLAM pipeline was executed directly onboard, while only the estimated six-degree-of-freedom pose was transmitted to the host platform. With dimensions of $108 \times 24.5 \times 12.5$ mm and a nominal power consumption of 1.5 W, the T265 demonstrated that real-time visual-inertial localization is compatible with compact and lightweight robotic platforms [92]. Although the system relied on a proprietary, hardware-specific implementation, it provides evidence that the size and power limitations of commercial general-purpose computing boards are not intrinsic to VIO itself. Figure 2.1 shows the T265 device with and without its protective cover. The figure demonstrates the feasibility of designing custom hardware that can reduce the footprint of visual localization

¹<https://www.intel.com/content/www/us/en/products/sku/192742/intel-realsense-tracking-camera-t265/specifications.html>

algorithms. While the dimensions are approximately the same, the board without the cover weighs only 13 g. Compared to 54 g of the covered product, it introduces a significant advantage in terms of weight reduction, which is crucial for small UAVs. For UAV deployment, the commercial enclosure is not necessarily required in its original form whenever equivalent mechanical protection and mounting conditions are ensured by the platform design. The enclosure mainly contributes to protecting the electronics, simplifying installation, and providing a robust product-level package. In a custom UAV integration, the same results could be obtained by placing the device in a protected location of the airframe, while preserving the required camera field of view, rigid mounting, cable retention, and thermal operating conditions. This example shows that visual localization hardware can be adapted to reduce weight without losing real-time performance.

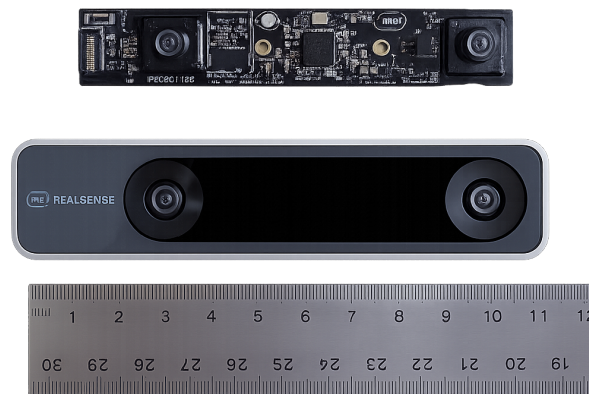


Fig. 2.1 The figure compares the dimensions (in centimeters) of the Intel RealSense T265 with and without its original protective cover. The device weighs 54 g with the cover and 13 g when the cover is removed.

Chapter 3

Improvements to State-of-the-Art VIO Pipelines

The scope of this chapter is to introduce novel methodologies to address the main limitations of state-of-the-art VIO pipelines. The considerations provided in Chapter 2 highlight a gap between current solutions and the performance required in real-world scenarios. In particular, monocular setups show limited robustness during specific phases of the pipeline or under certain motion conditions. Moreover, insufficient feature tracking degrades VIO performance, leading to inaccurate estimates and, in the worst cases, severe algorithm failures.

To analyze these problems accurately and better assess the required improvements, popular open-source state-of-the-art VIO pipelines are considered. The proposed solutions are integrated into these frameworks to mitigate the identified limitations of visual-inertial navigation. Finally, an experimental evaluation is conducted on different recorded datasets to demonstrate the effectiveness of the introduced methodologies.

3.1 Proposed solutions to Monocular Setup Issues

Monocular setups are an extremely convenient choice for UAVs requiring installation simplicity, low weight, and moderate computational load. On the other hand, in Chapter 2, the issues related to the employment of this setup have been provided,

highlighting the required improvements to state-of-the-art solutions for real-world scenarios' applicability. In particular, this dissertation identifies two aspects that currently compromise the application of monocular VIO in real-world scenarios:

- Position estimation during the initialization phase
- Failures of the pipeline when the system experiences pure rotational motions

While being aware of the intrinsic limitations of the monocular setup demonstrated in [62] and [63], methodologies to mitigate these limitations are introduced. In the proposed solution, the initialization phase is addressed by geometric considerations. Assuming the takeoff location of a UAV is known, a known planar surface is utilized to intersect back-projected feature rays, obtaining the 3D position of these features. This geometric constraint provides the solver with the scale of the problem, making translation observable as long as the intersected features are visible. Even if this approach strongly depends on assumptions about the surrounding environment, it is worth noting that in many applications these assumptions can be easily satisfied, since UAVs usually take off from known locations.

The second contribution consists of a methodology to deal with pure rotational motions. During pure rotations, sliding window algorithms continue optimizing the 3D position of the detected features. Since the scale becomes unobservable, the solver diverges while processing a solution that minimizes the residuals. To mitigate this behavior, non-initialized features are excluded from the optimization problem to avoid degenerate depth values. In addition, the features with the best baselines are treated as fixed parameters in the optimization problem. These features are named *anchors* and facilitate the optimizer in converging towards a stable solution.

The proposed solution is implemented within the VINS-Mono open-source code. The choice was determined by the following reasons:

1. The implementation provides satisfactory documentation and a Docker environment that facilitates result reproducibility.
2. Several video demonstrations prove the effectiveness of the solution.
3. The framework implements a standard sliding-window approach without over-complicating the problem.

4. The algorithm has great popularity within the community.

The experimental results indicate that the adopted methodologies improve VINS-Mono performance during both initialization and pure rotational maneuvers. In particular, the proposed initialization phase does not show any delay beyond the ordinary frame computation time. Instead, the methodology introduced to handle pure rotations drastically reduces position drift and prevents the solver from diverging during the optimization process.

3.1.1 Initialization Phase

Plane-Ray Intersection for Feature Depth Initialization

Let a feature be detected in an image acquired by a camera rigidly attached to the UAV. Assume that the camera center in the global reference frame is known and denoted by

$$\mathbf{o} \in \mathbb{R}^3.$$

Given a feature observation in the image, its back-projected ray direction expressed in the global frame is

$$\mathbf{d} \in \mathbb{R}^3, \quad \|\mathbf{d}\| = 1.$$

The ray associated with the feature can be written as

$$\mathbf{r}(t) = \mathbf{o} + t\mathbf{d}, \quad t > 0.$$

where t is a scalar value.

Known Plane Model

A known plane can be expressed with the following implicit equation:

$$\Pi: \quad \mathbf{n}^\top \mathbf{x} + d = 0,$$

where $\mathbf{n} \in \mathbb{R}^3$ is the plane normal and $d \in \mathbb{R}$ is the plane offset.

It is assumed that the known plane is bounded by a convex quadrilateral defined by the vertices.

$$\{\mathbf{v}_i\}_{i=1}^4, \quad \mathbf{v}_i \in \mathbb{R}^3.$$

Ray-Plane Intersection

The intersection between the ray $\mathbf{r}(t)$ and the plane Π is obtained by substituting the ray equation into the plane equation:

$$\mathbf{n}^\top (\mathbf{o} + t\mathbf{d}) + d = 0.$$

Solving for the scalar parameter t yields

$$t = -\frac{\mathbf{n}^\top \mathbf{o} + d}{\mathbf{n}^\top \mathbf{d}}.$$

The intersection is considered valid if

$$\mathbf{n}^\top \mathbf{d} \neq 0 \quad \text{and} \quad t > 0.$$

When these conditions are satisfied, the 3D position of the feature is computed as

$$\mathbf{p} = \mathbf{o} + t\mathbf{d}.$$

Bounded Plane Constraint

To ensure that the intersection point lies within the plane section defined by the four vertices, for each edge of the quadrilateral, the following condition has to be verified:

$$\mathbf{n}^\top ((\mathbf{v}_{i+1} - \mathbf{v}_i) \times (\mathbf{p} - \mathbf{v}_i)) \geq 0, \quad i = 1, \dots, 4,$$

If the inequality is satisfied for all edges, the point $\mathbf{p}$ lies inside the bounded planar region.

Use in Visual-Inertial Optimization

For features whose back-projected rays intersect the known plane, the corresponding 3D positions $\mathbf{p}$ can be estimated without relying on multi-view triangulation.

These features are treated in the visual-inertial optimization problem as fixed landmarks,

$$\mathbf{p} = \text{const},$$

by setting the corresponding parameter blocks as constant within the nonlinear least-squares optimization framework.

This geometrical approach addresses the initialization problem, arose from the inability of a monocular setup to recover the scale without a sufficient baseline between consecutive frames. This strategy prevents the solution of the solver from the divergence produced by IMU biases and integration errors.

3.1.2 Pure Rotational Motions

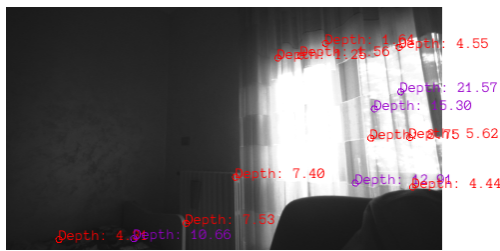
Pure rotational motions are critical for monocular VIO algorithms for the following reasons:

1. Absence of baseline between consecutive frames.
2. No accelerometer excitation.

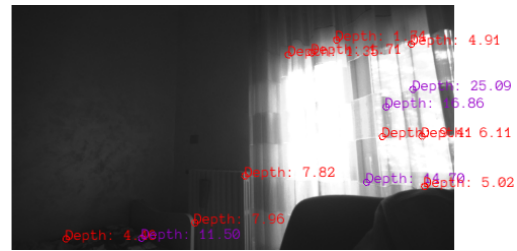
Before introducing the proposed solution, a concrete visualization of the behavior of the VINS-Mono implementation during a pure rotational motion is provided. The following assumptions must be considered:

1. Pure rotations require a specific setup to be reproduced; thus, in the considered datasets, pure rotations should be regarded only as an approximation of such motions.
2. Even if the effect is evaluated on the VINS-Mono implementation, it can be assumed that all sliding-window-based algorithms are affected by the same issue, as long as the implementation does not prevent these failures.

Figure 3.1 illustrates six frames of a dataset after a pure yaw rotation. Over the image, the detected features and the corresponding depth evaluated by the algorithm can be observed. The pictures show that the depth of the detected features rapidly grows across the frames up to degenerate values. This behavior is due to the inability of the optimizer to find a solution during pure rotational motions. While the feature depth grows, the position estimate rapidly drifts to reduce the corresponding growing residuals, producing drift and, in the worst cases, divergence of the solution. To facilitate the visualization, the points with a depth above 10 m are shown in violet.



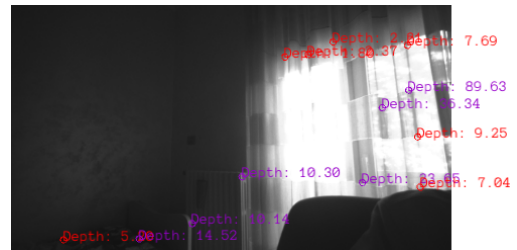
(a) Frame 199



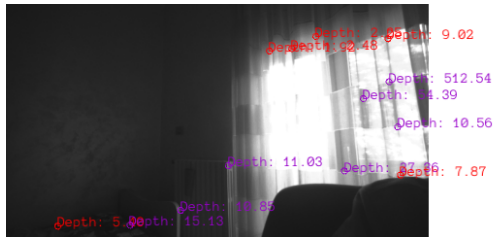
(b) Frame 201



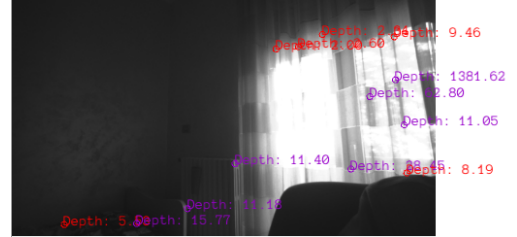
(c) Frame 203



(d) Frame 205



(e) Frame 207



(f) Frame 208

Fig. 3.1 Depth degeneration in the VINS-Mono implementation during a pure rotational motion: the pictures show the frames after a yaw rotation. In red and in violet, the points have depths respectively below and above 10 m.

Figure 3.2 shows the original logic of VINS-Mono compared to the one proposed in this chapter. VINS-Mono distinguishes four different types of features:

1. Not solved: default status for newly detected features.
2. Solved successfully: features whose triangulation produced a positive depth.
3. Solve failed: features whose triangulation produced a negative depth.
4. Initialized with default depth: features whose triangulation produced an unreliable depth (i.e., positive depth but smaller than a threshold).

The original algorithm includes all features in the optimization without any distinction, and the optimization problem is responsible for depth estimation. This approach works very well when there is enough baseline across frames and many features are properly triangulated. In this way, triangulation errors due to intrinsic calibration or inaccuracies in the position estimate are reduced by minimizing the residuals, and unreliable features initialized with a default depth tend to converge to the correct depth. However, this behavior does not occur during pure rotations. Since during this degenerate motion most of the features have negative or unreliable depths, the algorithm attempts to optimize them to reduce the residuals even if the states in the sliding window do not contain enough information about scale and translation. This produces divergence of the solution.

In the proposed approach, this problem is mitigated with two different adjustments to the original algorithm:

1. Only successfully triangulated features are included in the optimization process.
2. A small subset of the "best" triangulated features is promoted to anchors and is treated as constant parameters in the optimization problem. To select these anchors, the algorithm evaluates the number of observations and the baseline quality.

According to the methodology proposed, the tracked features are distinguished into anchors and non-anchors. The anchors are the features triangulated with a good baseline, whose 3D position is considered reasonably accurate. The remaining features are classified as non-anchors. Unlike non-anchors, these anchors are excluded from the optimization process. This strategy prevents the algorithm from forcing the

optimization of features that cannot be optimized during degenerate motions due to the lack of observability constraints.

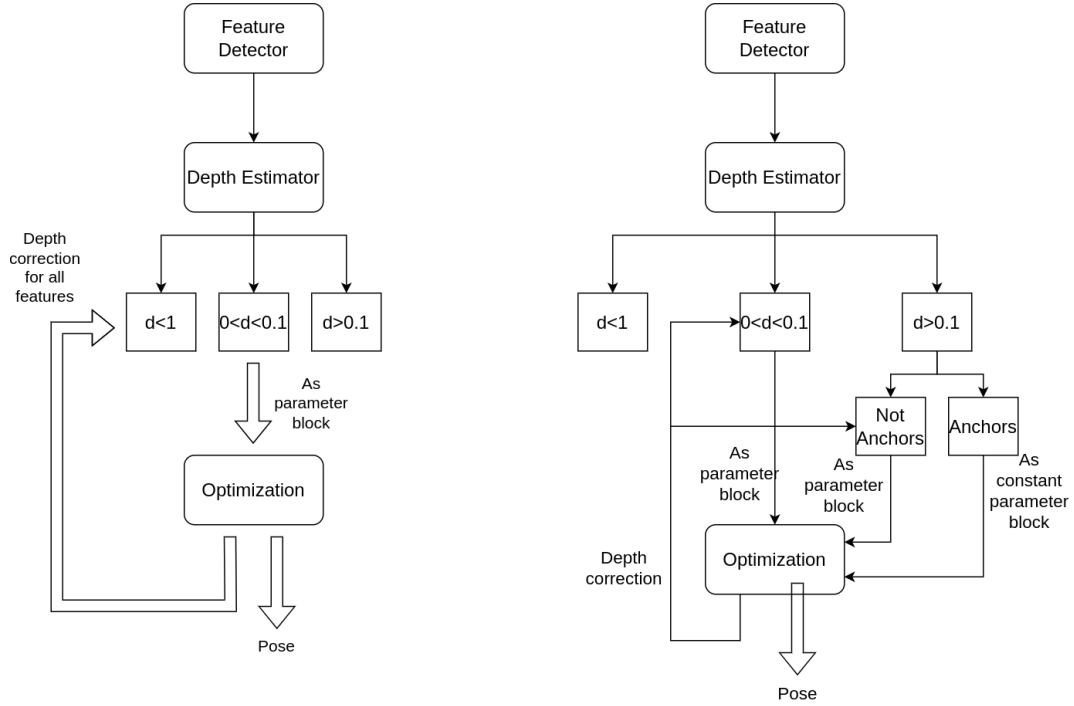


Fig. 3.2 The baseline framework logic (left) compared with our implementation (right)

3.1.3 Experimental Evaluation

Hardware Setup

To validate our solution, some datasets have been recorded with a RealSense D455 stereo camera (Figure 3.3). This device provides two stereo infrared cameras with many resolution options up to 1280x800. It features a variable frame rate up to 90 fps, a field of view of $87^\circ \times 58^\circ$ (HxV), and a stereo baseline equal to 95 mm. In addition, the device is equipped with a 6-axis IMU sensor. For our datasets, only the left camera stream has been recorded.

A 640x360 image resolution was utilized, with a frame rate of approximately 30 Hz. IMU data is provided at 200 Hz.



Fig. 3.3 The D455 camera from RealSense.

Data Collection

The experimental evaluation aims to demonstrate the effectiveness of our implementation both during the initialization phase and under pure rotational motions. To this end, the collected datasets emphasize these two aspects. Additionally, the presented implementation is evaluated over a subset of the popular public EuRoC MAV datasets [93], which were originally utilized by the VINS-Mono authors to evaluate the performance of their algorithm. This evaluation is conducted to ensure that our modifications do not compromise the accuracy and robustness of the original implementation. Since our adjustment concerning pure rotational motions excludes a subset of features from the optimization process, no accuracy improvements are expected on datasets where pure rotations do not occur. Nevertheless, to validate our solution, it is crucial to demonstrate that our modifications do not significantly degrade the localization accuracy of the baseline implementation.

For the initialization phase validation, four datasets (D1, D2, D3, D4) have been recorded with the D455 device. In all datasets, a horizontal plane, specifically an office desk, is inside the camera field of view. To simulate a UAV takeoff, the datasets are recorded by moving the camera upwards until a predetermined altitude is reached. Then, the camera is replaced at the starting position. Datasets D1, D2, and D3 are recorded by moving the camera at low, medium, and high velocities, respectively. D4 is recorded by slightly rotating the camera to provide the IMU with adequate excitation and facilitate initialization. This experimental evaluation aims to:

1. Verify that the features intersecting the plane are the ones that actually lie on the same plane.

2. Demonstrate that our method enhances localization accuracy during the initialization phase.
3. Demonstrate that the delay of pose estimation during the initialization phase is reduced to the regular frame processing time.

To verify the effectiveness of our solution during pure rotational motions, the datasets D5 and D6 have been recorded. In each dataset, five different yaw rotations are performed. After the last rotation, the camera is replaced at the starting position. For both the baseline and our implementation, the goal is to:

1. Evaluate the drift of the position at the end of each rotation.
2. Evaluate the overall drift of the position at the end of the dataset.

Finally, the RMSE of our implementation is evaluated over a subset of the EuRoC MAV datasets to compare the results with the baseline framework. This evaluation aims to:

1. Verify that our solution does not significantly degrade the localization accuracy of the baseline implementation.

3.1.4 Results

Initialization Phase

For datasets D1, D2, D3, and D4, the following results are provided:

1. A visualization of the features whose back-projected rays intersect the plane.
2. A qualitative comparison between the estimated trajectories obtained from the baseline algorithm and the proposed implementation.
3. The position error with respect to the ground truth is evaluated as the Euclidean distance between the estimated camera position and its known reference position.

4. The elapsed time from the beginning of the motion to the first non-zero value obtained from the position estimate. Any value greater than the measurement sensitivity, fixed at 0.001 m, is considered to be a non-zero value. This quantity estimates the delay introduced by the initialization phase.

For the sake of clarity, to evaluate the initialization delay, a custom user interface developed for offline analysis is utilized. The interface enables frame-by-frame analysis of the whole recorded sequence; thus, the frame where the motion starts can be easily recognized. Once the frame corresponding to the motion start and the one where the position estimate exceeds the measurement sensitivity are determined, the delay of the initialization can be computed by assuming that the frames are equally spaced in time.

Table 3.1 shows the results obtained from the four datasets.

Figure 3.4 provides a visualization of the points that lie on the selected planar surface. On the left, there is a frame of the dataset: the one on the top is related to our implementation, while the one on the bottom corresponds to the results obtained by the baseline algorithm. The features that lie on the plane are shown in red. The IDs associated with each feature are also provided in the visualization. The chosen plane in 2D image coordinates is represented in yellow, while its 3D representation is shown in blue.

Figure 3.5 provides a qualitative visualization of the trajectories. For each dataset, the figure shows the whole trajectory estimated by the original VINS-Mono framework in blue and our implementation in red. The ground-truth trajectory is reported in green, and it is drawn according to the measurements of the known reference position.

Table 3.1 Comparison of the selected metrics across the four datasets for two distinct cases.

Metric	Algorithm	D1	D2	D3	D4
Error [m]	Baseline	0.432	0.336	1.613	0.032
	Proposed	0.046	0.066	0.033	0.018
Delay [s]	Baseline	20.1	3.6	0.4	0.9
	Proposed	0.0	0.0	0.0	0.0

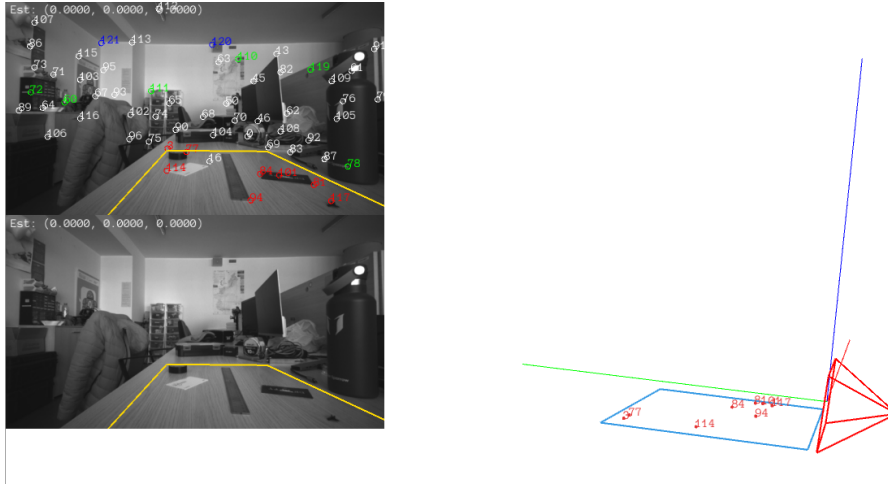


Fig. 3.4 Visualization of the planar surface utilized for geometrical constraints during the initialization phase. The 3D visualization shows the plane and the position of the detected features whose back-projected rays intersect the plane. In the picture, the planar surface is bounded by yellow lines, and the features that lie on the surface are visualized in red. The numbers close to the features are their corresponding IDs.

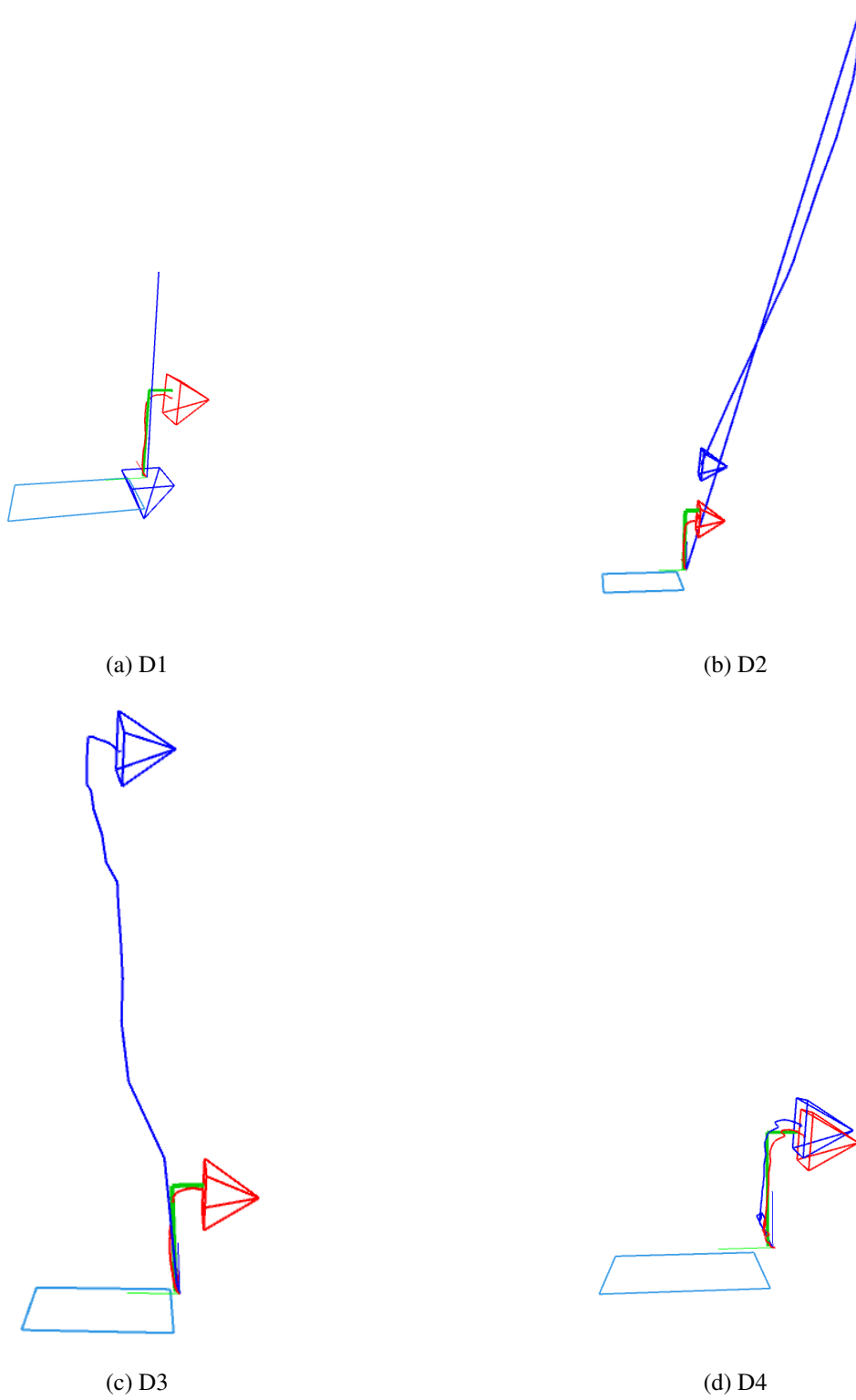


Fig. 3.5 Visualization of the trajectory for D1, D2, D3, and D4. The blue trajectory corresponds to the solution of the baseline algorithm. The red trajectory corresponds to our implementation. The green trajectory is the reconstruction of the ground truth according to the known reference position.

Pure rotational motions

For datasets D5 and D6, the following results for both the baseline framework and our implementation are shown:

1. The position error after each rotation. The error is computed by considering the position estimated immediately before one of the rotations ($\mathbf{p}_1$) and the position estimated immediately before the first rotation. The error is defined as the Euclidean norm of the difference between the two vectors:

$$\text{Error} = \|\mathbf{p}_1 - \mathbf{p}_0\|_2.$$

2. The position error evaluated at the end of the dataset. Since at the end of the dataset the camera is placed at the starting position, the ground truth position is known.
3. A visualization of the trajectory for the whole sequence.

For clarity, an explanation of the methodology utilized to compute the error is provided. In the recorded sequences, the camera only rotates along the yaw direction. As a consequence, under ideal conditions, the estimated position is expected to remain constant and only exhibit small oscillations around the position recorded before the first rotation. These oscillations arise from the practical difficulty of executing perfectly pure rotations without dedicated mechanical hardware or a testbed. Furthermore, as discussed in 3.1.4, the initialization phase may introduce substantial errors in the position estimate. To isolate and evaluate only the errors related to rotational motion, it is therefore more appropriate to use as a reference the position estimated at the end of the initialization phase, rather than the ground truth position.

Table 3.2 summarizes the results obtained from the two recorded sequences. Since a rotation is considered to be successfully estimated when the position error remains below 0.1 m, the corresponding values are highlighted in green. Instead, errors exceeding 0.5 m are marked in red, as such deviations are considered to be compromising during a real flight.

Figure 3.6 shows the trajectories of the baseline implementation (blue line) and the proposed method (red line). Two circles of 0.5 m diameter are drawn in green

Table 3.2 Errors in meters evaluated for both implementations across the two datasets. R1 indicates the error accumulated after the first rotation, while EOS stands for End of Sequence error. Errors below 0.1 m are in green, while errors above 0.5 m are in red.

Dataset	Algorithm	R1	R2	R3	R4	R5	EOS
D5	Baseline	0.042	0.108	0.174	0.502	1.353	0.949
	Proposed	0.314	0.177	0.183	0.053	0.126	0.022
D6	Baseline	1.833	5.330	7.677	8.350	11.275	14.483
	Proposed	0.114	0.091	0.068	0.072	0.192	0.198

around the first position estimated by the two methods once the target altitude is reached. Due to the missing ground truth, the circles are useful to visualize the error introduced by the rotations, assuming that the estimated trajectory is expected to remain inside the circle.



Fig. 3.6 Trajectory visualization for D5 and D6. The pictures show the comparison of the baseline framework trajectory (blue curve) with our implementation (red curve).

Evaluation on EuRoC MAV datasets

Table 3.3 shows the trajectory RMSE of the baseline implementation and the proposed solution evaluated over a subset of the EuRoC MAV datasets. As described in [93], the difficulty of the datasets is established according to the motion speed and the quality of illumination conditions. The Machine Hall (MH) datasets are recorded in the machine hall at ETH Zürich and simulate an industrial scenario. Instead, V2 datasets refer to sequences recorded in a Vicon room.

Table 3.3 Comparison of the selected metrics across EuRoC datasets. The RMSE is provided in meters.

Algorithm	MH_01_easy	MH_03_medium	MH_04_difficult	V2_02_medium	V2_03_difficult
Baseline	0.49	0.335	0.756	0.233	0.565
Proposed	0.316	0.272	0.612	0.344	0.69

Figure 3.7 provides a qualitative comparison of the 3D trajectories obtained from the two algorithms against the MAV ground truth. To facilitate the comparison, the whole trajectory is not provided for datasets where the drone visits the same area multiple times, making the visualization confusing.

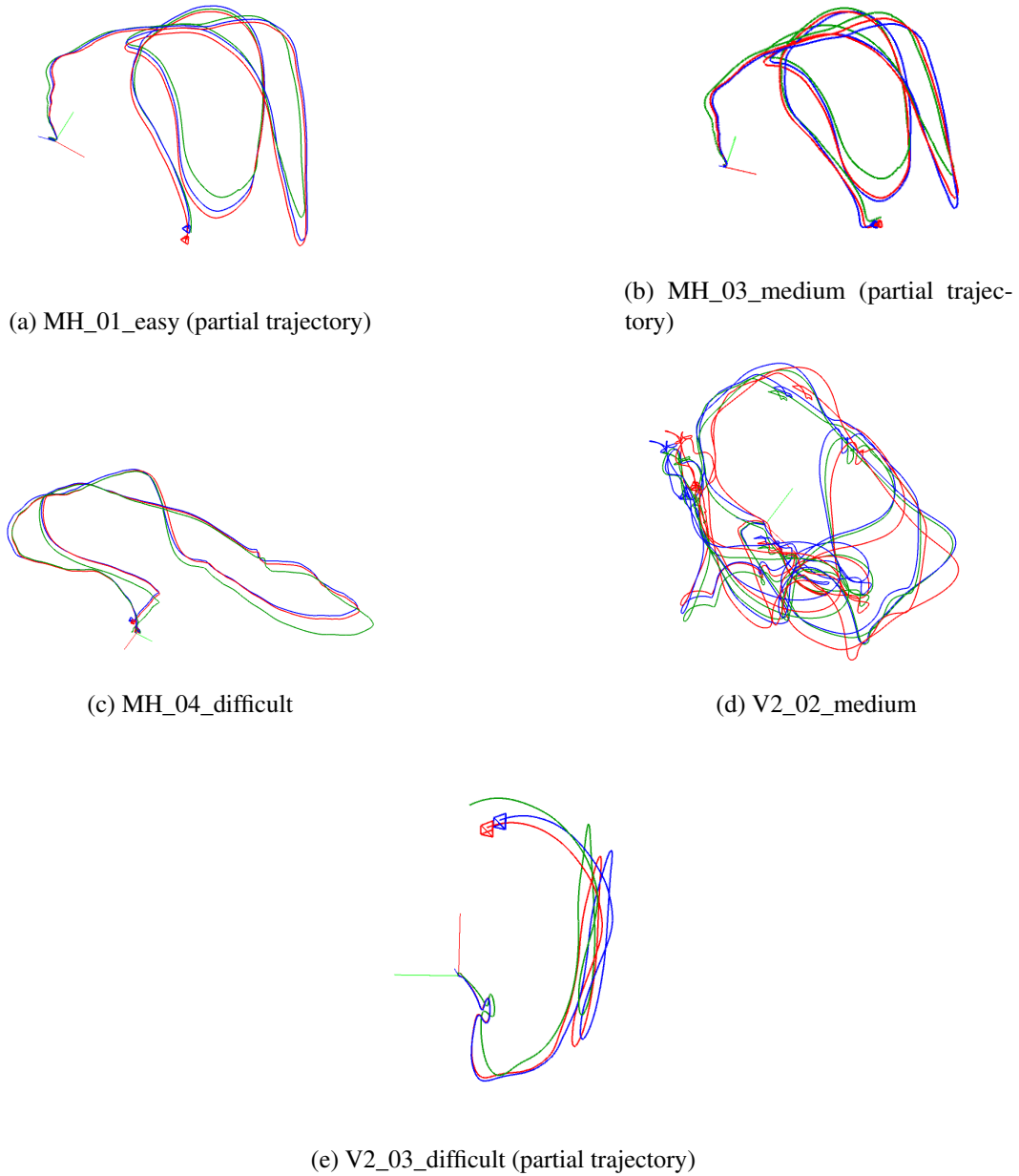


Fig. 3.7 The estimated trajectory evaluated over a subset of the EuRoC MAV datasets. The results show the baseline implementation (blue trajectory), proposed solution (red trajectory) and ground truth (green trajectory).

3.1.5 Discussion

Initialization Phase

Figure 3.4 demonstrates the capability of our implementation to properly detect the features that lie on the chosen plane. The red features detected in the top left image are inside the yellow plane, and their IDs correspond to those in the 3D visualization on the right side of the picture. Additionally, the 3D position of the features resulting from the intersection is consistent with the location of the points in the image. This can be observed by examining the feature IDs.

Table 3.1 and Figure 3.5 show that our implementation outperforms the baseline algorithm across all recorded datasets. From the trajectory visualization, it can be observed that the baseline algorithm exhibits a significant jump in the position estimate during the initialization phase. Even if the estimate partially recovers in a few frames, substantial errors remain in D1, D2, and D3. Since VINS-Mono requires adequate IMU excitation and a sufficient baseline to be properly initialized, the original baseline shows significant limitations during motions with poor inertial excitation. In D4, where the camera is intentionally moved at the beginning of the sequence to excite the IMU, the algorithm achieves significantly better performance. In contrast, our implementation exhibits lower errors across all datasets without experiencing any jump or discontinuity during initialization. In Table 3.1, the delay introduced by the initialization phase is reported for both implementations. Since VINS-Mono requires enough IMU excitation and a sufficient baseline to initialize properly, it experiences a considerable delay. This is particularly evident in dataset D1, in which the camera experiences the least excited motion. In this dataset, the baseline solution starts to estimate the motion after approximately 20 s, highlighting the limitations of the original implementation. Although this delay strongly depends on the motion profile, the collected datasets suggest a delay of approximately 1–2 seconds even for moderate IMU excitation. Instead, our implementation exhibits a delay of 0 s across all datasets. This indicates that as soon as the camera starts moving, the algorithm immediately produces a non-zero position estimate for the current frame. Naturally, this delay does not account for the delay introduced by frame processing, which, for VIO algorithms, is typically on the order of milliseconds. Delays in the order of seconds can be extremely high for real-time localization algorithms and can compromise the flight stability of a UAV. It is worth noting

that, probably, fine-tuning of the parameters could improve the performance of the baseline solution. Nevertheless, a strong dependence on parameter tuning is typically undesirable for these algorithms. On the other hand, the proposed solution requires that a known planar surface lie within the camera view during initialization. In scenarios where this does not occur, the provided approach cannot be applied. However, in many real-world applications, the takeoff location of a UAV is typically known, and the surrounding geometrical constraints may be helpful in improving localization robustness.

Pure Rotations

The results in Table 3.2 indicate that our implementation produces a more accurate position estimate during rotational motions. The baseline framework achieves better performance only after the first three rotations (R1, R2) of sequence D5, whereas the proposed method is superior in the remaining cases. Additionally, our implementation exhibits an error below the threshold of 0.1 m in 5 out of the 10 performed rotations, while errors exceeding 0.5 m never occur. On the other hand, the baseline implementation exhibits errors greater than 0.5 m in the majority of the examined scenarios. In particular, massive errors are experienced for dataset D6 due to a severe divergence of the solution.

The superiority of the proposed solution is further demonstrated by the 3D trajectories in Figure 3.6. The trajectory estimated by the baseline algorithm exhibits a large drift along all three axes. In contrast, our implementation shows a substantially reduced drift across both sequences. The baseline trajectory evidently falls outside the drawn circle, while the proposed solution remains within the bounds for the whole sequence.

Evaluation on EuRoC MAV datasets

The results presented in Table 3.3 demonstrate that the proposed solution does not significantly degrade the position estimate. In fact, across the Machine Hall sequences, our method shows slightly improved performance. On the contrary, the baseline implementation exhibits a lower RMSE for the Vicon-recorded datasets.

As shown in Figure 3.7, the estimated trajectories closely align, and the residual errors with respect to the ground truth are consistent with the quantitative results presented in Table 3.3.

Both qualitative and quantitative analyses have a key role in proving the effectiveness of the proposed solution. Since the proposed methods introduce several changes in a well-performing framework, it is essential to demonstrate that these changes do not introduce instability or degrade performance in other scenarios. The results provided in 3.1.4 address this requirement, covering sequences with different spatial scales, motion profiles, and illumination conditions.

3.2 Basalt VIO: An Open-Source VIO Pipeline

The development and experimental validation of the methods proposed in this thesis required the selection of a reliable open-source VIO pipeline as a reference implementation. Several publicly available systems were considered according to their reported accuracy and robustness on benchmark datasets, computational efficiency, software maturity, and suitability for modification. Among these alternatives, Basalt [35] was selected as the reference pipeline for the present work.

Basalt was identified as a particularly suitable solution because it combines accurate and robust visual-inertial state estimation with a contained computational load. This trade-off is particularly relevant for UAV applications, where the localization algorithm must operate in real time on embedded hardware with limited computational and power resources. In addition, the publicly available implementation provides a well-structured code, including modules for camera–IMU calibration, visual–inertial odometry, mapping, simulation, and evaluation. This modular organization, together with the available documentation and dataset support, made Basalt suitable for the integration and testing of the proposed extensions.

The main novelty of Basalt is its non-linear factor recovery approach for visual-inertial mapping. In conventional sliding-window VIO systems, old states are marginalized to maintain a bounded computational cost. Although marginalization preserves their information locally through a prior term, this term is linearized around a specific operating point and cannot be directly reused in a global bundle-adjustment problem. Basalt addresses this limitation by recovering a set of non-linear visual–

inertial factors that provide an approximation of the information accumulated by the local VIO estimator. These factors can then be combined with loop-closure constraints within a global bundle-adjustment framework, enabling globally consistent visual–inertial mapping while retaining relevant inertial information \cite{ref-basalt}.

Despite its strong performance, Basalt remains affected by the fundamental limitations of visual-inertial odometry discussed in the previous sections. In particular, its performance can degrade during visual degeneracy, low-texture observations, insufficient feature tracking, high-speed motions, and prolonged trajectories without reliable global constraints. For this reason, to address the common issues of state-of-the-art visual-based algorithms, this work starts from the Basalt pipeline and introduces additional modules aimed at mitigating these limitations. The following sections describe the proposed modifications and evaluate their contribution to the robustness of visual-inertial localization in challenging UAV scenarios.

3.3 Improving VIO Robustness During High-Speed Motions

Visual-Inertial Odometry algorithms rely on the ability to establish reliable correspondences between visual features observed in successive camera frames. These correspondences provide the geometric constraints required to correct the drift introduced by IMU propagation and to estimate the platform trajectory. Consequently, the robustness of the visual front-end is a fundamental requirement for the overall performance of a VIO pipeline.

High-speed motions represent a particularly challenging condition for visual tracking. Rapid translations, especially in the presence of nearby scene elements, and fast angular motions can produce large apparent displacements of image features between consecutive frames. Moreover, when the camera exposure time is not sufficiently short with respect to the platform motion, the acquired images are affected by motion blur. In these conditions, image gradients become less distinct, corners and other salient visual structures are degraded, and the appearance of individual features can change significantly between frames. As a consequence, both feature detection and feature association become less reliable.

Feature-based VIO pipelines are particularly affected by this problem. Common tracking techniques generally assume that the displacement of a feature between consecutive frames remains sufficiently limited to allow its association within a local search region. When the actual displacement exceeds this region, or when motion blur alters the local image appearance, the tracker may fail to associate the feature with its correct correspondence. The result can be a substantial reduction in the number of tracked features, the termination of long feature tracks, or the introduction of incorrect correspondences.

A temporary reduction in the number of tracked features may appear harmless, especially when it lasts for only a small number of frames. However, its effect can be critical for a VIO estimator. Visual constraints are not only required to correct the current pose estimate but also to maintain the consistency of the states included in the sliding window. If an insufficient number of correctly tracked features is available, the optimizer becomes increasingly dependent on the IMU propagation model. Since inertial integration is affected by sensor noise and bias accumulation, the predicted state may progressively diverge from the real trajectory.

More importantly, a limited set of visual measurements can result in a poorly conditioned optimization problem. The remaining features may not be sufficiently distributed across the image, may provide weak geometric constraints, or may include incorrect associations. In these situations, the nonlinear optimizer can converge towards an erroneous local solution. Once an incorrect pose estimate is introduced into the sliding window, the associated visual and inertial residuals may be linearized around an inaccurate operating point. The resulting error can then affect subsequent state estimates and, after marginalization, can remain embedded in the prior information used by the estimator. Therefore, even a short visual degradation caused by rapid motion can produce errors that persist after the image quality has recovered.

This problem is particularly relevant for UAV applications. Aerial platforms may experience rapid attitude changes during stabilization, trajectory tracking, obstacle avoidance, or aggressive manoeuvres. In these conditions, a temporary visual tracking failure can directly affect the position and attitude estimates used for autonomous navigation and control. While increasing the camera frame rate, reducing exposure time, or using cameras with improved low-light performance can mitigate the prob-

lem at the hardware level, these solutions may introduce additional requirements in terms of bandwidth, power consumption, illumination, or cost.

An alternative strategy consists of exploiting the high-rate inertial measurements already available within the VIO pipeline to support visual tracking. By integrating IMU measurements between consecutive camera frames, an estimate of the relative camera motion can be obtained before the visual update. This estimate can be used to predict the expected image location of each tracked feature in the following frame. Rather than initializing the tracker from the feature position observed in the previous image, the tracking process can start from an IMU-informed prediction that accounts for the expected camera motion.

Such a prediction does not eliminate the degradation introduced by severe motion blur, nor can it recover visual information that is absent from the image. Nevertheless, it can reduce the effective displacement that must be handled by the tracker, improve the initialization of the feature search process, and decrease the probability of losing correct correspondences during fast motions. Precisely in light of these considerations, this thesis introduces an IMU-aided feature prediction strategy aimed at improving the robustness of feature tracking during high-speed UAV motions. The methodology and its integration within the considered VIO pipeline are described in the following sections.

3.3.1 IMU-based Feature Prediction

Nomenclature

To simplify the interpretation of the mathematical formulation, a nomenclature table is provided with all the symbols utilized in this section.

Table 3.4 Nomenclature of all used symbols.

Symbol	Definition
Subindex W	Inertial reference system
Subindex C	Camera reference system

Continued on next page

Table 3.4 Nomenclature of all used symbols. (continued)

Symbol	Definition
ω_{WC}	Measured angular velocity of the camera frame C with respect to the inertial frame W
$\tilde{\omega}_{WC}$	True angular velocity of the camera frame C with respect to the inertial frame W
$\mathbf{a}_{WC}$	Measured linear acceleration of the camera frame C with respect to the inertial frame W
$\tilde{\mathbf{a}}_{WC}$	True linear acceleration of the camera frame C with respect to the inertial frame W
$\mathbf{b}_g$	Gyroscope bias
$\mathbf{b}_a$	Accelerometer bias
η_g	Gyroscope noise
η_a	Accelerometer noise
$\mathbf{R}_{WC}$	The 3×3 rotation matrix expressing the rotation from the camera frame C to the inertial frame W
$\mathbf{g}$	Gravity acceleration in the inertial reference frame W
$\times$	Skew-matrix operator ¹
$\exp$	Exponential map ²
$\mathbf{p}_{WC}$	Position of the camera C in the inertial frame W
$\mathbf{v}_{WC}$	Velocity of the camera C in the inertial frame W
Δt	Time interval between two consecutive IMU measurements
t_k	Time step at a discrete time k
$\mathbf{T}_{C_{k-1}C_k}$	The 4×4 transformation matrix expressing the transformation from the camera C frame at step k to step $k - 1$
$\mathbf{T}_{WC}$	The 4×4 transformation matrix expressing the transformation from the camera frame C to the inertial frame W
$\mathbf{p}_f$	3D position of one feature in the inertial frame W

Continued on next page

Table 3.4 Nomenclature of all used symbols. (continued)

Symbol	Definition
$\mathbf{p}_f^{hom}$	3D position of one feature in homogeneous coordinates
$\tilde{z}$	Average depth of the triangulated landmarks
¹ Given a vector $\omega = \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}$, its skew-symmetric matrix $\omega^\times \in \mathbb{R}^{3 \times 3}$ is:	
$\omega^\times = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}.$	
² The rigorous definition of this operator can be found in [68].	

Mathematical Formulation

To process images and track features across frames, Basalt utilizes the popular Lucas–Kanade methodology [94]. To reduce the computational effort, the algorithm typically searches for correspondences in a limited region of the image. The simplest approach consists of assuming that the features are in the same position as in the previous frame. Under this assumption, the features are searched only around that position, reducing the computation time and ensuring real-time performance. This approach works properly as long as the scene remains relatively stable and no motion blur occurs. However, during high-speed motions, the position of a feature considerably changes across frames, producing failures in the tracking process.

In its original implementation, Basalt utilizes this simplification, resulting in it being very sensitive to the motion blur introduced by high-speed movements. Since the quality of the position estimate heavily depends on feature tracking, rapid movements significantly degrade the localization accuracy.

To mitigate the impact of the motion blur introduced by rapid movements, a methodology that leverages IMU data to predict the feature positions is introduced in the original pipeline. The approach utilizes the last position of the camera evaluated by the VIO algorithm and the data coming from the IMU sensor to predict the camera position in the next frame. Before processing the next frame, the relative transformation between the previously estimated camera position and the actual

position is calculated. Once this transformation is computed, the 3D features can be back-projected into the image plane. The result of the back-projection represents an estimate of the feature location in the next frame.

Relying on IMU data for feature tracking produces a reliable optical flow even during fast movements. This allows for a more accurate and robust estimation of the trajectory even under challenging conditions such as rapid drone motion. Although IMU-based position estimates rapidly drift, they provide accurate information in the short term. Considering that VIO pipelines typically run at a 15–30 Hz image frequency, the IMU is utilized to predict the position within 0.066–0.033 s. In this range, the IMU should provide sufficiently accurate estimates to obtain a reliable prediction of the feature location.

As shown in [68], given the reference system of the camera (C), and an inertial reference system (W), the angular velocity $\tilde{\omega}_{WC}$ and the acceleration $\tilde{\mathbf{a}}_{WC}$ of the camera C with respect to the reference system W are

$$\tilde{\omega}_{WC}(t) = \omega_{WC}(t) + \mathbf{b}_g(t) + \eta_g(t) \quad (3.1)$$

$$\tilde{\mathbf{a}}_{WC}(t) = \mathbf{a}_{WC}(t) + \mathbf{g} + \mathbf{b}_a(t) + \eta_a(t) \quad (3.2)$$

where

- ω_{WC} is the true angular velocity of the camera estimated by the IMU
- $\mathbf{b}_g$ is the gyroscope bias
- $\mathbf{b}_a$ is the accelerometer bias
- η_g is the gyroscope noise
- η_a is the accelerometer noise
- $\mathbf{g}$ is the gravitational acceleration in the inertial reference system

The derivative of the rotation of the camera frame with respect to the inertial frame can be evaluated as:

$$\dot{\mathbf{R}}_{WC}(t) = \mathbf{R}_{WC}(t)[\tilde{\boldsymbol{\omega}}_{WC}(t) - \mathbf{b}_g(t) - \boldsymbol{\eta}_g(t)]^\times \quad (3.3)$$

where $^\times$ is the skew-matrix operator (see Table 3.4). Equation (3.3) represents the continuous-time rotational kinematic model on $SO(3)$.

Assuming that the angular velocity is constant within each IMU integration interval Δt_i , the rotation of the camera between two consecutive image frames can be obtained through the exponential map as:

$$\mathbf{R}_{WC}(t_{k+1}) = \mathbf{R}_{WC}(t_k) \prod_i \exp([\tilde{\boldsymbol{\omega}}_{WC}(t_i) - \mathbf{b}_g(t_i) - \boldsymbol{\eta}_g(t_i)]^\times \Delta t_i) \quad (3.4)$$

where the index i refers to the IMU samples collected between the two image frames acquired at t_k and t_{k+1} , and Δt_i is the time interval associated with the i -th IMU measurement.

Since the IMU is sampled at a higher frequency than the camera, multiple IMU measurements can be available between two consecutive image frames. Therefore, the rotation propagation between t_k and t_{k+1} is obtained by composing the incremental rotation matrices computed from the angular velocity measurements acquired within this interval. This formulation should not be interpreted as the independent numerical integration of the nine entries of the rotation matrix. Instead, each angular velocity measurement is integrated over the corresponding interval Δt_i to obtain an incremental rotation, which is mapped onto the rotation manifold ($SO(3)$) through the matrix exponential. Therefore, if $\mathbf{R}_{WC}(t_k)$ is a valid rotation matrix, the propagated rotation remains orthogonal up to numerical precision. The rotation matrix notation is used here because it simplifies the expression of coordinate transformations and feature projection.

Since multiple IMU measurements can be available between two consecutive image frames, the position and velocity can be propagated by integrating the acceleration over each IMU interval. The velocity and position are then propagated as

$$\mathbf{v}_{WC}(t_{i+1}) = \mathbf{v}_{WC}(t_i) + \mathbf{a}_{WC}\Delta t_i, \quad (3.5)$$

$$\mathbf{p}_{WC}(t_{i+1}) = \mathbf{p}_{WC}(t_i) + \mathbf{v}_{WC}(t_i)\Delta t_i + \frac{1}{2}\mathbf{a}_{WC}\Delta t_i^2. \quad (3.6)$$

Therefore, the camera position at t_{k+1} is obtained by recursively applying (3.5) and (3.6) over all IMU samples collected between t_k and t_{k+1} .

The transformation matrix $\mathbf{T}_{C_{k-1}C_k}$ between the camera in the current frame and the camera in the previous frame is:

$$\mathbf{T}_{C_{k-1}C_k} = \mathbf{T}_{WC}(t_k)^{-1}\mathbf{T}_{WC}(t_{k-1}) \quad (3.7)$$

where

$$\mathbf{T}_{WC}(t) = \begin{bmatrix} \mathbf{R}_{WC}(t) & \mathbf{p}_{WC}(t) \\ 0 & 1 \end{bmatrix} \quad (3.8)$$

Now each feature can be unprojected to obtain the homogeneous coordinates $\mathbf{p}_f^{hom}$ and rescale them by the average depth evaluated in the previous step to obtain the 3D position of the feature $\mathbf{p}_f$:

$$\mathbf{p}_f(t_{k-1}) = \mathbf{p}_f^{hom}(t_{k-1})\tilde{z} \quad (3.9)$$

where $\tilde{z}$ is the average depth of the landmarks collected in the previous steps. Note that the feature position refers to the previous time step t_{k-1} . Knowing the 3D position of each feature at the previous time step t_{k-1} , its 3D position in the current frame t_k can be computed as:

$$\mathbf{p}_f(t_k) = \mathbf{T}_{C_{k-1}C_k}^{-1}\mathbf{p}_f(t_{k-1}) \quad (3.10)$$

By projecting the 3D features onto the image plane, an estimate of the feature's position can be obtained.

3.3.2 Experimental Evaluation

Hardware Setup

One OAK-D Pro Wide stereo camera¹ from Luxonis (Figure 3.8) has been utilized to perform the experimental evaluation. The tests were conducted by streaming the images at 30 Hz with a 640x600 resolution. These cameras feature fisheye lenses with a field of view of 120°x95°x70° (DxHxV).



Fig. 3.8 The OAK-D Pro Wide stereo camera from Luxonis

Data collection

To test the proposed IMU-based feature prediction approach, three different datasets were recorded (D1_easy, D2_medium, D3_difficult). In these sequences, the camera rotates around the axis normal to the image plane to emulate fast maneuvers and produce motion blur. Across the three datasets, the speed of rotation increases according to the difficulty level associated with each dataset name. This experimental evaluation aims to verify the capability of the proposed methodology to improve feature tracking during rapid motions. For each dataset, the following aspects are evaluated:

1. The error committed by the proposed method and the baseline implementation.
2. The number of features with an error above a threshold.

For this experimental evaluation, the results are presented as follows:

¹<https://shop.luxonis.com/products/oak-d-pro-w?variant=43715946447071>

1. Table 3.5 contains the average error and the maximum error in pixels between the predicted position and the actual position of the feature in the current frame.
2. Figure 3.9 shows two histograms for each dataset to compare the baseline solution with the proposed approach. Each histogram indicates the number of features that exhibit an error within a certain range. For these results, the error is the Euclidean norm of the translation difference for each feature.

3.3.3 Results

For this experimental evaluation, the results are presented as follows:

1. Table 3.5 contains the average error and the maximum error in pixels between the predicted position and the actual position of the feature in the current frame.
2. Figure 3.9 shows two histograms for each dataset to compare the baseline solution with the proposed approach. Each histogram indicates the number of features that exhibit an error within a certain range. For these results, the error is the Euclidean norm of the translation difference for each feature.

The experimental evaluation is performed over approximately 200 frames for each dataset.

Table 3.5 Errors in prediction estimate: the table shows the average error computed over the sequences (Avg Error) and the maximum error obtained over all frames.

Dataset	Configuration	Avg Error (pixel)	Max Error (pixel)
D1_Easy	No IMU	7.04	43.08
	With IMU	0.25	8.72
D2_Medium	No IMU	11.80	79.02
	With IMU	1.00	17.31
D3_Difficult	No IMU	27.98	230.93
	With IMU	2.53	75.38

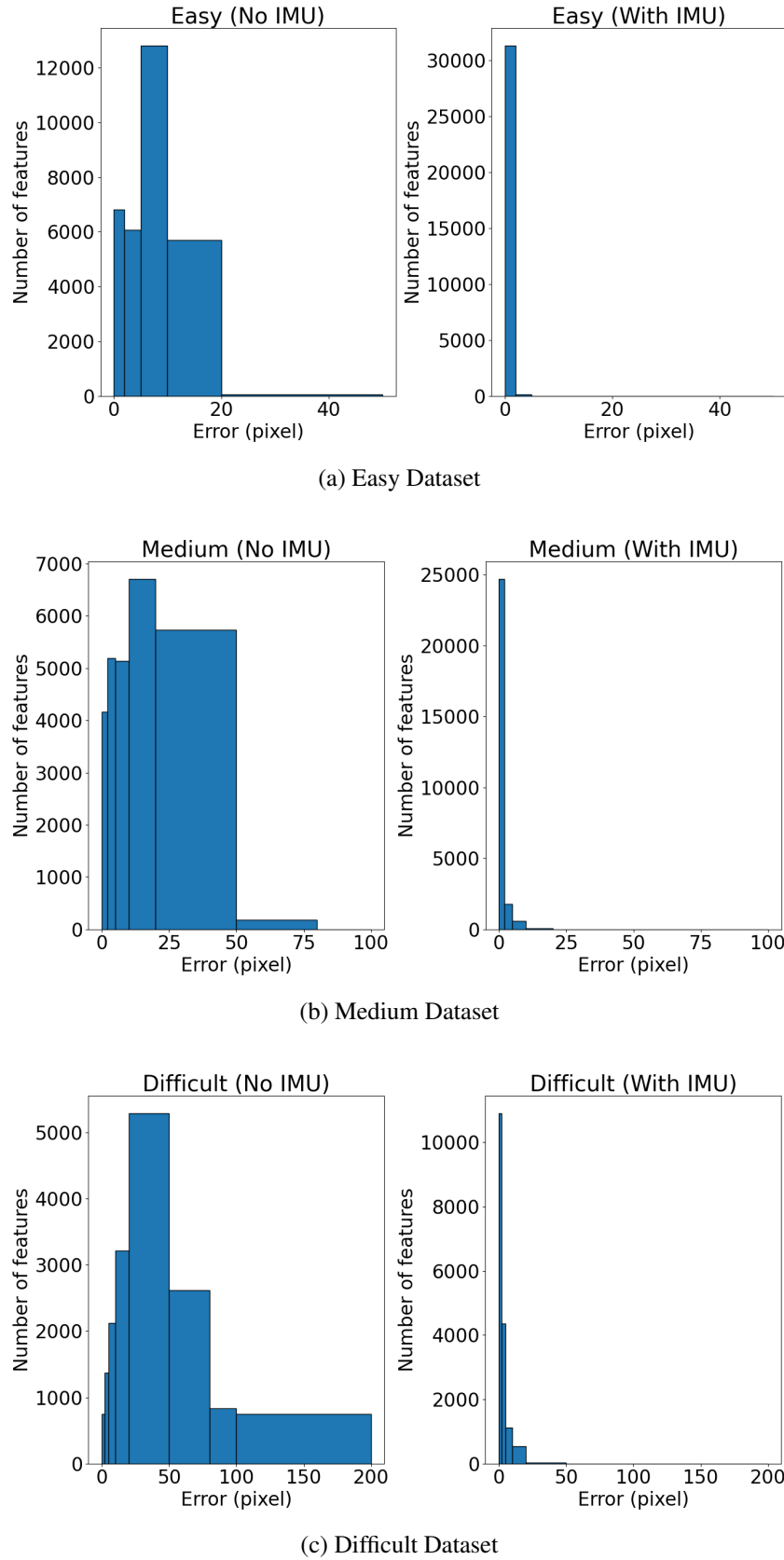


Fig. 3.9 Impact of the IMU prediction: **(a)** The histogram for the *Easy* dataset counts the occurrences of features with prediction errors in pixel units. **(b)** The *Medium* dataset error distribution. **(c)** The *Difficult* dataset error distribution. The data is collected over 200 frames.

3.3.4 Discussion

From the experimental results collected for the IMU-based feature prediction, the proposed methodology outperforms the baseline implementation. On dataset D1_easy, Table 3.5 shows that both methods are reliable, with low average errors. On the contrary, when rotation speed increases in datasets D2_medium and D3_difficult, the proposed approach demonstrates its superiority, showing reduced error in the position estimate of the tracked features. Once the assumption of relative stability no longer holds, an accurate prediction of the feature position is required to guarantee reliable tracking. For a further analysis, the histograms illustrated in Figure 3.9 indicate that the error dispersion is considerably reduced with the proposed methodology. This trend is evident in all datasets and proves the effectiveness of our methodology in reducing the number of tracking failures when motion blur occurs. The experimental results indicate that IMU-based feature prediction considerably improves feature tracking, especially during fast motion in which motion blur occurs.

3.4 Multi-Camera Pipeline Implementation

In this dissertation, a multi-camera setup refers to a configuration in which more than two cameras are utilized for VIO-based localization. Typically, in these configurations, the cameras are mounted to cover an extended field of view, enabling the VIO framework to extract features over a wider scene. This is particularly useful in texture-less scenarios, where few and low-quality features are extracted. In these scenarios, feature tracking degrades and, as a consequence, the position estimate accumulates drift. Therefore, a multi-camera setup represents a valuable solution in these challenging environments, enhancing the position estimate and improving localization robustness.

Additionally, when the configuration involves overlapping fields of view among cameras, the same feature can be observed by more than two cameras. These multiple observations provide the solver with more constraints, facilitating the convergence of the solution and improving position accuracy.

As a consequence, a multi-camera setup provides the following advantages for VIO-based localization:

- Extended field of view to cover a wider scene
- Increase in the number of observations for features tracked over more than two cameras

3.4.1 Generalization of the Image Processing

As concerns the image processing pipeline, Basalt only supports a stereo setup. The image is discretized into a square grid, and in each region, a fixed number of features can be detected. These detected features are tracked on the right camera using the Lucas–Kanade method. In the next frame, only newly detected features in the left image are tracked across the right camera, whereas the features resulting from the previous frame are tracked independently over the two images.

In our implementation, the optical flow generation is generalized to a multi-camera setup. In the shared region of the image, the features are tracked from one camera to the other. Then, new features are extracted in the remaining part of the image and tracked on cameras whose fields of view overlap with the current camera.

To achieve this generalization, a parent-child structure for camera relationships is implemented. In this structure, each of the N cameras may have one parent and from zero to M child cameras. The parent of camera i specifies the camera j with which camera i shares part of the field of view. For the parent of camera i , a bounding box has to be defined for the shared region of camera i with camera j . The children of camera i indicate which cameras share part of the field of view with i . The shared region of each child with camera i has to be defined for each child.

The process works as follows: first, features are extracted from the image captured by camera i , excluding the bounding box specified by its parent. Then, these features are tracked across the bounding boxes of each child camera. If a camera has no parent, the features are extracted from the entire image. This implementation is illustrated by the flow chart in Figure 3.10. With this methodology, any multi-camera setup can be accommodated. The relationships between the parent and child cameras, as well as the bounding boxes for the shared fields of view, must be manually provided before running the algorithm. Assuming that the cameras have the same lens, the shared bounding box can be experimentally identified once and remain consistent over time.

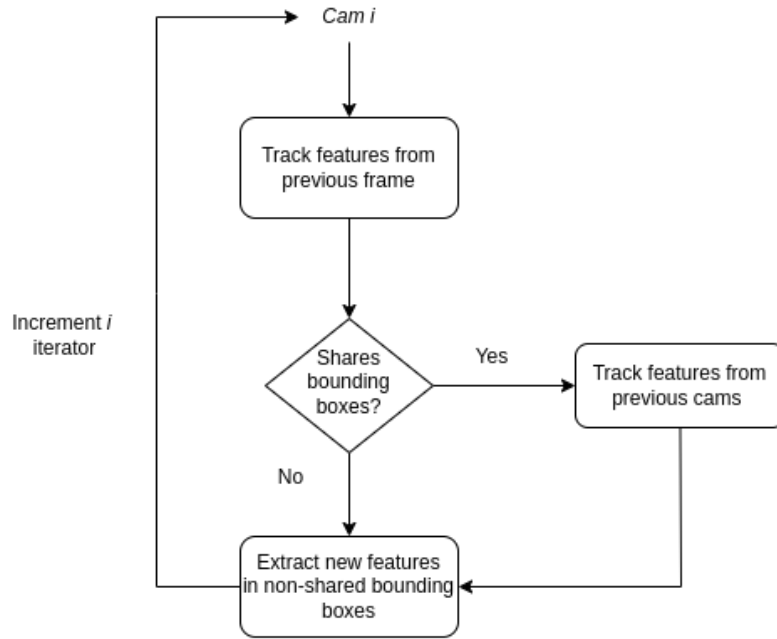


Fig. 3.10 The flow chart illustrating the optical flow generation algorithm generalized to a multi-camera system.

3.4.2 Keyframe Generation

In the original implementation of the algorithm, new landmarks are added by triangulating the unconnected features of the left camera of the stereo setup. When the ratio between the unconnected features of the left camera and the total number of features tracked in the current frame is below a threshold, the features are triangulated across the previous frame. Then, a new observation is added to the landmark database. Since this approach is only suitable for a stereo camera setup, an implementation to accomplish the landmark database update for a multi-camera setup is proposed. For each frame, the unconnected points are evaluated for each camera. The cameras that share the field of view with another one are considered to evaluate the ratio between the unconnected and the total number of features. If at least one of the ratios is below a threshold, the algorithm triangulates the unconnected points across the previous frame. In this way, the algorithm is enabled to generate keyframes for each camera and not only for the left camera of the original stereo configuration.

Algorithm 1 Original Stereo Camera Keyframe Selection

```

1: Initialize  $connected0 \leftarrow 0$ 
2: Initialize  $unconnected\_obs0 \leftarrow 0$ 
3: for each feature  $kpt\_id$  in left camera observations do
4:   if  $lmdb.landmarkExists(kpt\_id)$  then
5:      $tcid\_host \leftarrow lmdb.getLandmark(kpt\_id).host\_kf\_id$ 
6:     Add observation to landmark database
7:     Increment  $connected0$ 
8:   else
9:     Add  $kpt\_id$  to  $unconnected\_obs0$ 
10:  end if
11: end for
12: Compute ratio:  $r \leftarrow \frac{connected0}{connected0 + |unconnected\_obs0|}$ 
13: if  $r < threshold$  and  $frames\_after\_kf > min\_frames$  then
14:   Triangulate unconnected points across the previous frame
15:   Add new keyframe for the left camera
16: end if

```

3.4.3 Experimental Evaluation

Hardware Setup

The cameras utilized for the experiments are the OAK-D Pro Wide stereo cameras² from Luxonis. The tests were conducted by streaming the images at 30 Hz with a 640x600 resolution. These cameras feature fisheye lenses with a field of view of 120°x95°x70° (DxHxV). The ground truth data was obtained using a VICON motion capture system. The recording was performed in a 5x6 m room equipped with 10 VICON Bonita 10 cameras mounted on the walls at approximately 2 meters above the floor. Vicon Tracker 3.10 was used as the tracking software.

To evaluate the computational cost, the VIO pipeline is executed on an NVIDIA Jetson Orin Nano, which represents a standard for this type of application.

²<https://shop.luxonis.com/products/oak-d-pro-w?variant=43715946447071>

Algorithm 2 Multi-camera Keyframe Selection (**Proposed Solution**)

```

1: Initialize  $connected[i] \leftarrow 0$  for each camera  $i$ 
2: Initialize  $unconnected\_obs[i] \leftarrow \emptyset$  for each camera  $i$ 
3: for each camera  $i$  do
4:   for each feature  $kpt\_id$  in camera  $i$  observations do
5:     if  $lmdb.landmarkExists(kpt\_id)$  then
6:        $tcid\_host \leftarrow lmdb.getLandmark(kpt\_id).host\_kf\_id$ 
7:       Add observation to landmark database
8:       Increment  $connected[i]$ 
9:     else
10:      Add  $kpt\_id$  to  $unconnected\_obs[i]$ 
11:    end if
12:  end for
13: end for
14: for each camera  $i$  sharing FOV with another camera do
15:   Compute ratio:  $r_i \leftarrow \frac{connected[i]}{connected[i] + |unconnected\_obs[i]|}$ 
16:   if  $r_i < threshold$  then
17:      $take\_kf = true$ 
18:   end if
19: end for
20: if  $take\_kf$  then
21:   for each camera  $i$  do
22:     Triangulate unconnected points across the previous frame
23:     Add new keyframe for camera  $i$ 
24:   end for
25: end if

```

Data Collection

The multi-camera implementation is first evaluated on four different datasets. The first two datasets (D1, D2) are recorded with the two stereo cameras tilted at 45° , while datasets D3 and D4 contain sequences recorded with the two stereo cameras facing in opposite directions. Both configurations are shown in Figure 3.12. Figure 3.11 outlines the shared field of view for each camera. The difference between the two configurations is that *config2* has a wider field of view compared to *config1*. Instead, in *config1*, the algorithm can track features on more than one camera, increasing the observations for the same feature. The datasets are recorded starting from the VICON room and then moving around the corridors. After returning to the VICON room, the accumulated error with respect to the ground truth is evaluated. This approach was chosen because the corridors in the datasets present challenging conditions for VIO algorithms. Once returned to the VICON room, a comparison between the ground truth and the Basalt trajectory is provided for each dataset. This experimental evaluation aims to:

1. Provide a comparison of the position accuracy obtained with stereo cameras and a multi-camera setup
2. Evaluate the computational cost of each configuration

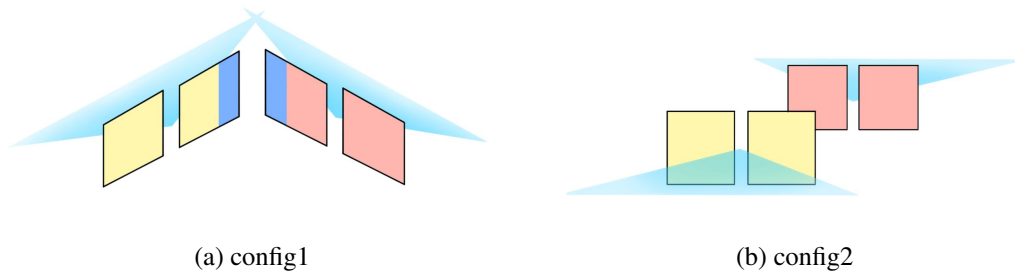


Fig. 3.11 The stereo camera configurations: (a) *Stereo1* (cam0 and cam1, yellow field of view) and *Stereo2* (cam2 and cam3, red field of view) with a shared field of view between cam1 and cam2 (blue); (b) *Stereo1* and *Stereo2* with non-overlapping fields of view.

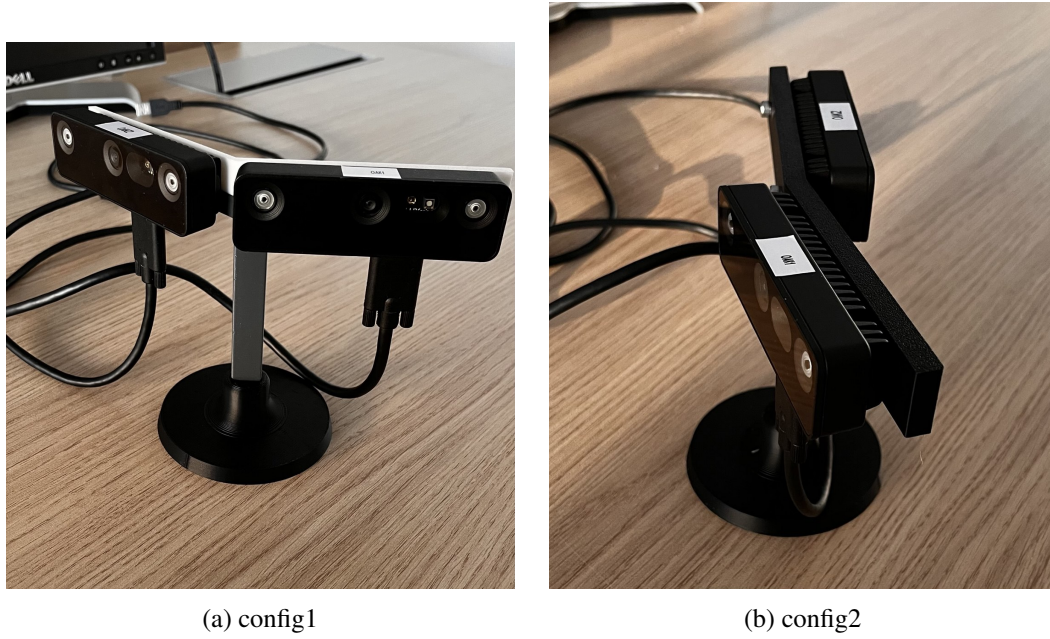


Fig. 3.12 The two configurations of the multi-camera setup: (a) The two stereo cameras are tilted at 45 degrees; (b) The two stereo cameras point in opposite directions.

For each challenging condition, a score between zero and ten is assigned. A score of zero indicates that the condition is absent from the dataset, whereas a score of ten corresponds to its maximum presence. The evaluation is reported in Table 3.6. The score is not evaluated with a rigorous method, but it should be intended as a pure qualitative evaluation of the challenging condition.

Table 3.6 Scores assigned to the challenging visual conditions in the considered datasets. A score of zero indicates the absence of the condition, whereas a score of ten indicates its maximum presence.

Dataset	Featureless Walls	Reflective Glass	Poor Lighting
D1	6	4	6
D2	6	4	5
D3	5	3	6
D4	5	3	5

Additionally, to evaluate the robustness of a multi-camera setup, the implementation is evaluated on dataset D5. This dataset is recorded only for the *config1* setup, as this configuration is particularly suitable for covering a wide field of view. In this

sequence, one stereo camera is intentionally placed towards a white wall, where few or no features can be extracted by the algorithm. This experimental evaluation aims to verify that, even if one of the cameras in the setup is not able to track features, the overall position estimate remains stable thanks to the optical flow provided by the other camera.

3.4.4 Results

To compare the accuracy of the estimated trajectory with the ground truth, the Root Mean Square Error of the Absolute Trajectory Error (RMSE ATE) is utilized. In Tables 3.7 and 3.8, the ground truth trajectory is compared with:

1. The trajectory estimated by the first stereo camera (*Stereo 1*)
2. The trajectory estimated by the second stereo camera (*Stereo 2*)
3. The trajectory estimated by the multi-camera setup (*Multicamera*)

To reduce the impact of stochastic data, the reported values correspond to the arithmetic average of the RMSE ATE obtained from five trials. In the same tables, the average percentage of CPU usage is reported. Additionally, Figure 3.13 illustrates the x - y plane trajectories for a qualitative comparison.

Table 3.9 contains the position estimated by the *Stereo1* and *Multicamera* setups after 3 s, 5 s, and 10 s from the beginning of the sequence. Since the cameras remain stationary, the ground truth translation is 0 m along all axes. Additionally, Figure 3.14 illustrates the scene recorded in dataset D5.

Table 3.7 RMSE ATE comparison (values in meters) for *config1* with average CPU load percentages. For each dataset, the total traveled distance is indicated in parentheses next to the dataset name.

Dataset	Algorithm	AVG Error (m)	CPU
D1 (148 m)	Stereo 1	1.34	148%
	Stereo 2	1.81	151%
	Multicam	1.22	275%
D2 (171 m)	Stereo 1	0.57	148%
	Stereo 2	1.24	151%
	Multicam	0.29	275%

1 Results from 5 trials per configuration. Best average errors highlighted in bold. CPU load measured at 30 Hz with 640×600 resolution.

Table 3.8 RMSE ATE comparison (values in meters) for *config2* with average CPU load percentages. For each dataset, the total traveled distance is indicated in parentheses next to the dataset name.

Dataset	Algorithm	AVG Error (m)	CPU
D3 (142 m)	Stereo 1	0.34	109%
	Stereo 2	0.92	122%
	Multicam	0.30	210%
D4 (164 m)	Stereo 1	0.98	109%
	Stereo 2	1.31	122%
	Multicam	0.49	210%

* Results from 5 trials per configuration. Best average errors highlighted in bold. CPU load measured at 30 Hz with 640×600 resolution.

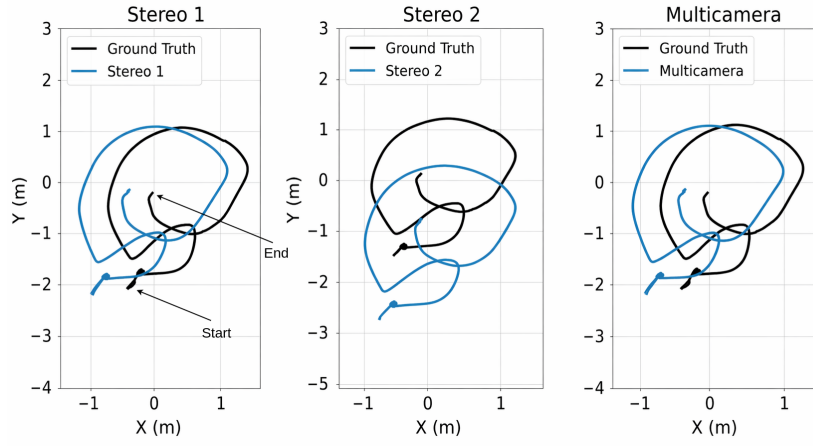
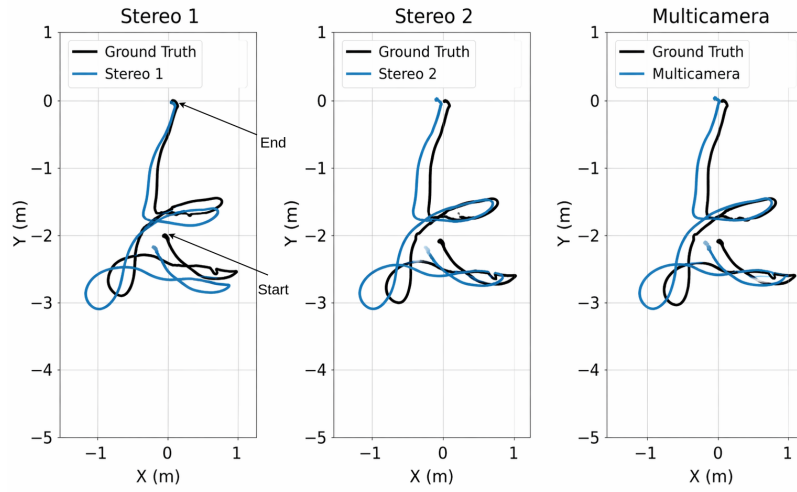
(a) *config1* on D3(b) *config2* on D4

Fig. 3.13 Comparison of the trajectory in the xy plane: (a) Trajectories for *config1* on D3; (b) Trajectories for *config2* on D4. The trajectories refer to the data collected after returning to the VICON room.

Table 3.9 Comparison of final estimated positions (meters) between Stereo 1 and Multicamera setups (dataset D5).

Time	Configuration	X	Y	Z
3 s	Stereo 1	-1.21	-0.28	4.64
	Multicamera	-0.04	0.00	0.08
5 s	Stereo 1	-2.79	-1.02	13.14
	Multicamera	-0.04	-0.08	-0.00
10 s	Stereo 1	-13.99	-33.86	53.94
	Multicamera	0.00	0.01	0.05



Fig. 3.14 The multi-camera setup during robustness testing: *cam0* and *cam1* (Stereo 1); *cam2* and *cam3* (Stereo 2).

3.4.5 Discussion

The multi-camera implementation presented in this work supports arbitrary configurations. Its performance is evaluated in terms of accuracy and robustness, utilizing two different configurations. The multi-camera setup offers improved localization accuracy compared to a traditional stereo configuration in challenging environments. From Tables 3.7 and 3.8, the multi-camera setup shows reduced errors compared to the stereo configuration for all recorded datasets. The *Stereo2* camera exhibits the worst results in terms of position accuracy. This trend can be explained by considering that for *Stereo2*, the VIO pipeline runs over the images of the second stereo camera and the IMU data recorded from the *Stereo1* inertial sensor. Since the two stereo cameras are not hardware synchronized, a latency is introduced between the visual data and inertial measurements. This latency is responsible for the degradation of the position accuracy, and these delays also propagate to the accuracy of the multi-camera setup. Thus, the RMSE ATE values reported in the two tables may be further improved by hardware-synchronizing the two cameras. Unfortunately, hardware synchronization is not available for OAK-D Pro Wide cameras; thus, this consideration cannot be supported by an experimental evaluation. In the same tables, the average percentage of CPU usage is reported. As expected, the multi-camera setups produce a considerable increase in the computational load. However, a CPU usage below 300% remains affordable for powerful onboard computers such as NVIDIA Jetson platforms.

While the accuracy of the position estimate heavily depends on camera synchronization, the results in Table 3.9 indicate that the multi-camera setup considerably increases the robustness of the localization without being affected by the latency within the devices. As expected, the position estimated by *Stereo1* rapidly drifts due to the absence of any tracked features. Thanks to the redundancy introduced by *Stereo2*, the position estimated by this device is stable around 0 m. The conducted robustness tests highlight the superior performance of the multi-camera configuration in handling feature-poor environments, while also revealing how quickly the localization accuracy of a stereo setup deteriorates under those conditions. Since these tests have been conducted to prove the redundancy of our multi-camera implementation, only *config2* has been considered.

Considering the growth of computational load and the additional installations required, the reader may ask whether a multi-camera configuration is justified given

the presented results. The answer is strictly related to the applicative scenario. For instance, a multi-camera setup results particularly convenient in low-textured environments. At the same time, in many real-world scenarios, autonomous robots are equipped with more cameras to accomplish more efficient data collection. Therefore, once additional cameras are already included in the sensor setup, utilizing them even for localization purposes represents a cost-effective way to improve the robustness and reliability of the navigation system without introducing further sensor redundancy.

Chapter 4

Sensor Fusion for VIO Drift Reduction

In Chapter 2, the problem of positioning drift accumulated by VIO –and, more generally, dead-reckoning– algorithms has been discussed. SLAM-based methods can effectively improve global consistency over long trajectories. However, their reliability may decrease in repetitive or visually degraded environments. Additionally, these methods require the platform to revisit previously observed areas and therefore cannot guarantee a continuous correction of the pose estimate.

Alternative solutions include marker-based localization and UWB systems. Although these approaches can provide robust absolute or relative position measurements under favourable operating conditions, their accuracy strongly depends on the visibility and placement of the markers or on the geometry and line-of-sight conditions of the UWB anchors. In particular, occlusions, non-line-of-sight propagation, multipath effects, and suboptimal anchor configurations can significantly degrade the quality of the resulting position estimates. Moreover, both approaches require the installation, calibration, and maintenance of dedicated infrastructure in the operating environment.

For these reasons, complementary sensing strategies that exploit sensors already available onboard the robotic platforms, or information acquired by cooperating robots, represent an attractive alternative. Such approaches can provide additional metric constraints to bound the drift of the VIO estimate while reducing the dependence on fixed external infrastructure.

4.1 Overview of the Proposed Solutions

This chapter presents two methodologies designed to reduce the drift affecting UAVs relying on VIO-based localization. The first method focuses on the vertical component of the state estimate. It fuses the altitude estimated by the VIO pipeline with measurements provided by a downward-facing range finder. While range measurements can provide an accurate local estimate of the distance between the UAV and the ground, their direct integration is complicated by the presence of obstacles, uneven terrain, and sloped surfaces. These conditions may generate measurements that do not correspond to the actual UAV altitude and can therefore introduce significant estimation errors. To address this issue, the proposed method identifies inconsistent range measurements through a Mahalanobis-distance-based detection mechanism. Two parallel Kalman filters are employed: the first estimates the altitude state, whereas the second is dedicated to detecting inconsistencies between the expected and measured range values. When an inconsistency is detected, the range measurements are excluded from the state update, and the corresponding offset is estimated. This approach enables the system to exploit range measurements when they are reliable while preventing corrupted observations from degrading the altitude estimate.

The second methodology addresses the reduction of the global positional drift through cooperative localization. In this framework, a UAV and a UGV operate in the same environment, with the UGV acting as an external source of geometric information. The UGV is equipped with a three-dimensional LiDAR and at least one camera. A neural network processes the camera images to identify the UAV and estimate its bounding box. The bounding box is then projected into three-dimensional space, defining a pyramid originating from the camera center. The LiDAR points contained within this pyramid are analysed, and the first visible cluster is associated with the UAV. The distance between the UGV and the identified cluster provides a metric observation of the UAV position relative to the UGV. Assuming that the UGV pose is accurately estimated in a common reference frame, this observation can be used as an external constraint to correct the positional drift accumulated by the UAV VIO system.

The two methodologies address different but complementary aspects of VIO drift. The range-finder fusion approach provides a local and lightweight correction of the vertical state, relying exclusively on sensors installed onboard the UAV. On

the other hand, the cooperative localization approach introduces an external metric constraint that can limit the accumulated positional error over larger portions of the trajectory. Together, these methods show how complementary sensing modalities and inter-robot cooperation can extend the reliability of VIO-based navigation without requiring the availability of GNSS signals.

The remainder of this chapter is organized as follows. First, the range-finder-based altitude estimation method and its inconsistency-detection strategy are presented. Subsequently, the cooperative UGV–UAV localization framework and the LiDAR-assisted UAV distance-estimation algorithm are described. For both methodologies, the experimental results are discussed, and their contribution to VIO drift reduction is evaluated.

4.2 Range-Aided Altitude Estimation

Although VIO algorithms accumulate drift along the 3D directions without any significant distinction, generally, the altitude drift can be compensated by altitude sensors. Moreover, an accurate altitude estimate in autonomous drones operating in indoor environments becomes even more crucial compared to outdoor applications. In outdoor scenarios, errors in meter units do not necessarily compromise a mission. Instead, in indoor environments, imprecise altitude estimation can significantly increase the risk of collisions with ceilings or compromise the mission's success.

Range finders are sensors that measure the distance from a surface using a laser beam. Compared to cameras, range finder measurements are more reliable in the long-term period as they do not accumulate drift and can achieve centimeter-level precision under optimal conditions. Range finders are widely used as a source for altitude estimation in outdoor missions, especially in situations when GNSS signals can become unreliable [95]. When inconsistencies in heights relative to the ground occur (i.e., in the presence of obstacles or slopes), the range finder altitude provides useless raw data for localization purposes. In both outdoor applications and indoor environments, the altitude variations experienced when the drone flies over obstacles or encounters ground slopes significantly affect altitude estimation. Therefore, employing a range finder for altitude estimation establishes the issue of detecting ground inconsistency.

4.2.1 Related Works

In [95], measurements from a range finder pointing to the ground and one pointing to the ceiling are fused in an adaptive Kalman filter, achieving more accurate altitude estimation. Even with a single range finder, reliable results can be obtained as in [96], where the range finder is combined with inertial sensors and a barometer for altitude estimation. In [97], altitude estimation involves cooperation between a range finder, a LiDAR, and inertial sensors, with valuable considerations to efficiently detect inconsistencies in altitude from range finder measurements.

The presented approach is based on the one proposed in [97], which combines range finders and inertial measurements with a 3D SLAM LiDAR-based algorithm. In contrast to this work, the proposed solution implements the filter, replacing the 3D LiDAR with a stereo camera whose data are utilized to run a VIO algorithm. Similar to the methodology outlined in [97], even in the proposed approach, Mahalanobis distance is utilized to identify inconsistencies in range finder measurements. However, to improve the robustness of inconsistency detection, two separate Kalman filter pipelines are utilized: the first one is employed for state estimation, whereas the other one is exclusively dedicated to the inconsistency detection. This approach allows for a clearer distinction of the inconsistencies without affecting the accuracy of the estimate. The implementation has proven to be both reliable and robust through rigorous experimental testing, showcasing promising performance.

4.2.2 Mathematical Formulation of the Problem

As this treatment aims to propose a sensor fusion methodology to estimate the altitude of a rotary wing UAV, only the translational dynamics along the z-axis of a world reference frame are modeled. The UAV is a quadrotor modeled as a point mass, and its state vector can be defined by the discrete-time linearized system as suggested by [97]. Defining t_i as the time at the i -th iteration of the Kalman Filter, the following equation can be written:

$$x_{[t_i]} = Ax_{[t_{i-1}]} + Bu_{[t_i]} \quad (4.1)$$

where

$$x = [z, \dot{z}, \ddot{z}, o_{RF}, o_C, o_V]^T, \quad (4.2)$$

$$A = \begin{bmatrix} 1 & \Delta t & \frac{\Delta t^2}{2} & 0 & 0 & 0 \\ 0 & 1 & \Delta t & 0 & 0 & 0 \\ 0 & 0 & e^{-\frac{\Delta t}{\tau}} & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad (4.3)$$

$$B = \begin{bmatrix} 0 \\ 0 \\ 1 - e^{-\frac{\Delta t}{\tau}} \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad u = a_d. \quad (4.4)$$

o_{RF} and o_C represent the offset of the estimated altitude from the range finder and T265 camera altitude measurements, respectively. While o_V is the bias of the velocity estimated by the flight control unit. They can be respectively defined as:

$$o_{RF} = z - z_{RF}, \quad (4.5)$$

$$o_C = z - z_C, \quad (4.6)$$

$$o_V = V_{FCU} - \dot{z}, \quad (4.7)$$

where z_{RF} is the measurement of the range finder, z_C is the altitude estimated by the VIO algorithm, and V_{FCU} is the velocity measured by the Flight Control Unit (FCU). In previous equations, Δt is the time difference between two subsequent steps of the Adaptive Kalman Filter, and τ is a delay measurement of the system's reaction to its inputs. The control output of the model is the desired acceleration a_d along the z axis, commanded to the UAV by the onboard computer and derived from

autonomous planning considerations. The measurement model of the system can be described by the equation:

$$z_{[t_i]} = Hx_{[t_i]} \quad (4.8)$$

where

$$H = \begin{bmatrix} 1 & 0 & 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad (4.9)$$

$$z = [z_{RF}, z_C, V_{FCU}]^T. \quad (4.10)$$

Considering a zero-mean white noise on both state and measurement prediction models, (4.1) and (4.8) can be rewritten as

$$x_{[t_i]} = Ax_{[t_{i-1}]} + Bu_{[t_i]} + v_{[t_i]}, \quad (4.11)$$

$$z_{[t_i]} = Hx_{[t_i]} + w_{[t_i]}, \quad (4.12)$$

where $v_{[t_i]}$ and $w_{[t_i]}$ are defined by the diagonal elements of process covariance matrices Q and R . Specifically, Q and R are defined as follows

$$Q_{[t_i]} = \text{diag}(q_1, q_2, q_3, q_4, q_5, q_6), \quad (4.13)$$

$$R_{[t_i]} = \text{diag}(r_1, r_2, r_3). \quad (4.14)$$

The diagonal elements of the Q matrix are the process noise variance of the corresponding element of the state vector, while the diagonal element of the R matrix represents the noise variance of the corresponding measurements.

4.2.3 Altitude Estimation

The proposed state estimation algorithm consists of a Kalman Filter where the prediction of the state vector x and the measurement vector z are performed with the modeled linear system previously introduced. Since the proposed methodology involves runtime adjustments of the filter equations, this implementation is referred to as the Adaptive Kalman Filter (AKF). Although in literature this terminology usually refers to online adaptation of filter parameters, the term *adaptive* well reflects the intentions of the proposed algorithm. The algorithm consists of two steps:

1. State propagation conducted via a conventional Linear Kalman Filter
2. Adjustment of matrices to update the range finder's offset without influencing the other states when inconsistencies are detected.

State propagation

The prediction of the state vector x^- and its covariance matrix P^- are defined as:

$$\hat{x}_{[t_i]}^- = A\hat{x}_{[t_{i-1}]} + Bu_{[t_i]}, \quad (4.15)$$

$$P_{[t_i]}^- = AP_{[t_{i-1}]}A^T + \Delta t Q_{[t_i]}. \quad (4.16)$$

Then, the measurement innovation can be calculated as:

$$y_{t_i} = z_{t_i} - H\hat{x}_{[t_i]}^- \quad (4.17)$$

and the covariance of the predicted measurement is:

$$S_{[t_i]} = HP_{[t_i]}^-H^T + R_{[t_i]}. \quad (4.18)$$

Hence, the optimal gain K can be computed as

$$K_{[t_i]} = P_{[t_i]}^-H^TS_{[t_i]}^{-1} \quad (4.19)$$

and the estimated state and its covariance matrix can be finally updated:

$$\hat{x}_{[t_i]} = \hat{x}_{[t_i]}^- + K_{[t_i]} y_{[t_i]}, \quad (4.20)$$

$$P_{[t_i]} = (I - K_{[t_i]} H) P_{[t_i]}^-. \quad (4.21)$$

Range Finder Offset Update

To prevent the propagation of erroneous measurements, the range finder's offset has to be updated without influencing the other states. Once the offset is updated, the range finder contribution can be reintegrated into the Kalman Filter. As recommended by [97], this task is accomplished by a reconfiguration of the observation matrix H :

$$H_{off} = \begin{bmatrix} 0 & 0 & 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}. \quad (4.22)$$

Consequently, the Kalman gain matrix can be calculated as:

$$K_{off[t_i]} = H_{off}^T (H_{off} H_{off}^T)^{-1} \quad (4.23)$$

and it is employed to update the state estimate and its covariance:

$$\hat{x}_{[t_i]} = \hat{x}_{[t_i]}^- + K_{off[t_i]} y_{[t_i]}, \quad (4.24)$$

$$P_{[t_i]} = (I - K_{off[t_i]} H_{off}) P_{[t_i]}^-. \quad (4.25)$$

All the other equations of the Kalman filter propagation remain the same and are calculated with the original observation matrix H . The reconfiguration of the Kalman gain contemporaneously updates the range finder offset while removing its contribution from the filter propagation.

4.2.4 Mahalanobis Distance for Inconsistency Detection

The altitude obtained from the fusion of VIO estimate and range finder measurements may become inconsistent when the height from the ground is altered by obstacles or slopes. The Kalman Filter alone is not able to detect these inconsistencies, and its solution tends to converge towards wrong altitude estimates. Therefore, detecting when the range finder measurement inconsistency occurs becomes crucial in these applications.

A straightforward approach to detecting these inconsistencies is by comparing two consecutive range finder measurements. However, the range finder outputs intermediate values when jumps in the measurements occur, and these values depend on the rate of the transition from one measurement to another, which is unpredictable during a mission. Therefore, establishing a predefined threshold to detect these transitions becomes impractical. An alternative approach consists of comparing the current measurement with the arithmetic mean or the median value of the previous n measurements, where n represents the predetermined buffer length. While this approach is more reliable than the initially proposed one, false-positive inconsistencies detection may occur during rapid altitude variations.

The evaluation of the Mahalanobis distance [98] typically addresses the detection of such inconsistencies. The Mahalanobis distance measures the deviation of a measurement from its normal distribution. According to the Kalman Filter state propagation, the squared Mahalanobis distance can be computed as:

$$D_{[t_i]}^2 = y_{[t_i]}^T S_{[t_i]}^{-1} y_{[t_i]}. \quad (4.26)$$

When no inconsistency occurs, the estimated state of the filter is represented by a Gaussian distribution, and the squared Mahalanobis distance is substantially smaller than 1 (often on the order of 10^{-6} and 10^{-7}). This result is expected as, in the absence of nonlinearities, the Kalman Filter produces a Gaussian distribution of the state. When an inconsistency occurs, the hypothesis of linearization fails, and the estimate deviates from the normal distribution. Once linear conditions are restored, the filter estimate gradually re-aligns with a Gaussian distribution. Although the transition is rapid, the Mahalanobis distance typically exhibits high variations, providing valuable information for inconsistency detection. In [97], the authors detect inconsistencies by evaluating the Mahalanobis distance during filter

propagation. However, due to the adaptive nature of the filter, the variation of the Mahalanobis distance is lower compared to the Linear Kalman filter. In the proposed approach, the Mahalanobis distance is evaluated within the Linear Kalman filter, as it ensures greater reliability in detecting inconsistencies. For this purpose, a parallel instance of a Linear Kalman filter operates within the estimation pipeline, exclusively for evaluating the Mahalanobis distance and detecting non-linear behaviors. Even though this approach requires a tuning of the distance threshold, the values during inconsistency events significantly differ from those recorded when inconsistencies are absent. Therefore, the tuning of the distance threshold can be performed with great generality to cover many motion scenarios.

An overview of the algorithm is provided by the schematic in Figure 4.1.

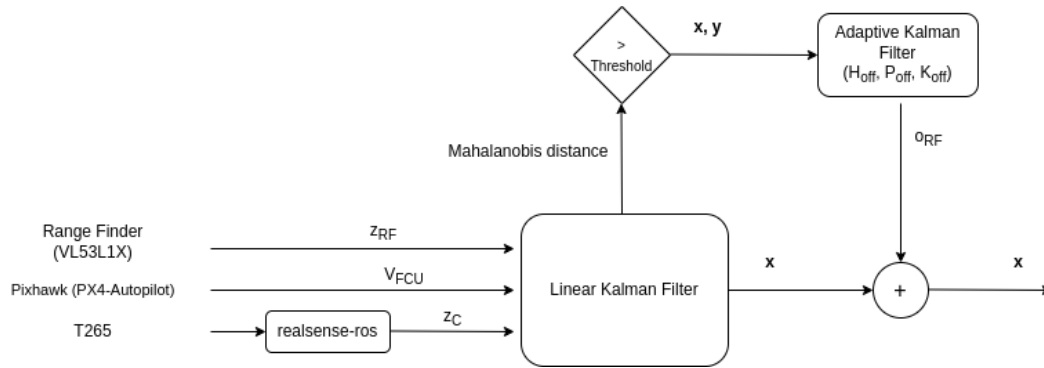


Fig. 4.1 Schematic of the Adaptive Kalman Filter fusing range finders, FCUs, and T265's measurements.

4.2.5 Experimental Evaluation

Hardware Setup

For the experimental evaluation, an Intel RealSense T265 stereo tracking camera is utilized. This device was released by Intel RealSense in 2019 and features two fisheye cameras and an integrated IMU to perform visual-inertial odometry directly onboard, without imposing any computational load on the host platform. Thanks to its reduced weight and embedded processor running the localization algorithm, the T265 has been one of the most widely adopted solutions for implementing VIO algorithms on UAV platforms, before being discontinued in 2021. Although the details of the onboard algorithm have not been publicly released by the manufacturer,

useful documentation regarding the device can be found in [92]. The T265 has been employed in several scientific works, such as [99–101].

The range finder utilized for the experimental evaluation is a VL53L1X developed by STMicroelectronics. By emitting a 940 nm infrared laser, this sensor is capable of measuring its absolute distance from a surface. It exhibits a maximum range of approximately 4 m for highly reflective targets, a minimum range of around 4 cm, and a resolution of 1 mm. Due to its small size and low power consumption, it is typically utilized on UAV platforms for both altitude sensing and obstacle avoidance tasks.

The experimental tests have been conducted within an indoor environment. The algorithm operates on the measurements coming from the sensors mounted on the drone, as shown in Figure 4.2. The distance sensor is connected to a Pixhawk v4 platform flashed with the PX4 flight controller firmware. The range finder raw measurements are read directly from the PX4 firmware and accessed from the NVIDIA Jetson onboard computer using ROS2.



(a) Range finder pointing downward.



(b) The UAV used for experimental data collection.

Fig. 4.2 Overview of the UAV's experimental setup.

Data Collection

Two different datasets (D1 and D2) have been recorded for the experimental evaluation. In D1, the UAV is first raised to a fixed height above the floor and is then manually moved to point the downward-facing rangefinder at different surfaces. Specifically, the UAV is moved above a desk, returned above an unobstructed portion of the floor, moved above the desk again, and subsequently moved above a chair

while passing over the floor. Finally, it is returned above an unobstructed floor area. This experimental evaluation aims to verify the sensitivity of the filter to ground inconsistencies in altitude estimation.

In D2, the T265 camera has been partially blinded for a while. This condition emulates a failure of the algorithm that might occur in low-texture environments or unfavorable brightness conditions. Additionally, this condition easily allows for drift accumulation that might occur over long trajectories. The experimental evaluation aims to show the filter's ability to accurately estimate the altitude when the visual-inertial odometry algorithm accumulates drift, thanks to the presence of the range finder measurements.

Although ground truth is not provided, the range finder measurements can be considered highly accurate altitude information whenever no obstacles fall within the sensor field of view. This assumption is supported by the millimeter-level precision of the employed rangefinder for distances above 3 meters.

4.2.6 Results

The results obtained from the dataset D1 are shown in Figure 4.3. The picture shows the altitude estimated by the VIO algorithm (blue curve), the range finder measurements (orange curve), and the altitude estimated by the proposed filter (green curve). Figure 4.4 shows the Mahalanobis distance (orange curve) and the offsets estimated by the filter during the same dataset (blue curve).

Figure 4.5 summarizes the results obtained from dataset D2. Once again, the pictures compare the altitude estimated by the VIO algorithm (blue curve), the range finder measurements (orange curve), and the altitude estimated by the proposed filter (green curve).

4.2.7 Discussion

The results provided by Figure 4.3 demonstrate the limited sensitivity of the altitude estimated by the filter to the measurement inconsistency of the range finder. The altitude estimate is well aligned with the range finder measurements until an inconsistency is detected. Once the inconsistency is recognized, the offset between the range finder measurements and the Kalman filter estimate is updated, and the filter

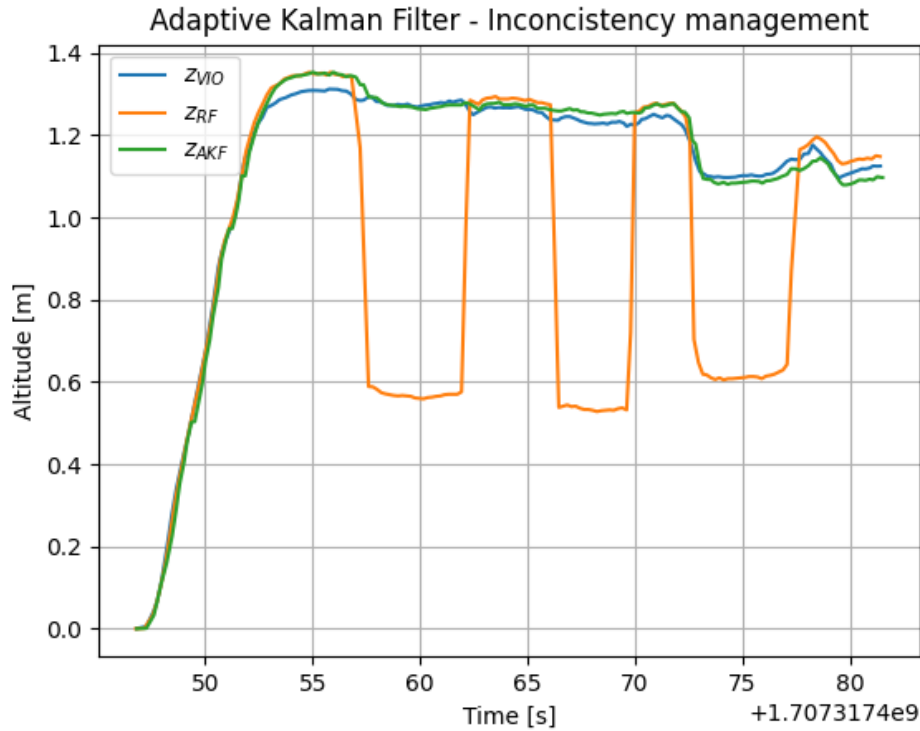


Fig. 4.3 Adaptive Kalman Filter managing range finder inconsistency.

integrates the range finder's contribution once again. The detection of subsequent inconsistencies leads to a gradual accumulation of a minor drift due to the filter's capability to estimate the offset. In Figure 4.4, the estimated offset and the squared Mahalanobis distance are shown. The jumps of the range finder offset correspond to inconsistencies in the range finder measurements. As shown in the picture, the Mahalanobis distance significantly increases with range finder offsets. Although the first two offsets are of equal magnitude, the estimate differs by a few centimeters. This discrepancy is attributed to the inherent limitations of the linear Kalman filter propagation, while more advanced filters may provide better performance.

The results shown in Figure 4.5 demonstrate the effectiveness of the proposed algorithm in reducing the drift accumulated by a VIO algorithm. In the sequence, at approximately 50 s, the T265 has been intentionally blinded to simulate conditions characterized by a lack of features or low brightness. As a consequence, the T265 altitude estimate deviates from the ground truth, while the estimate follows the range finder measurements. When the drone returns to the ground, the drift of the T265 is

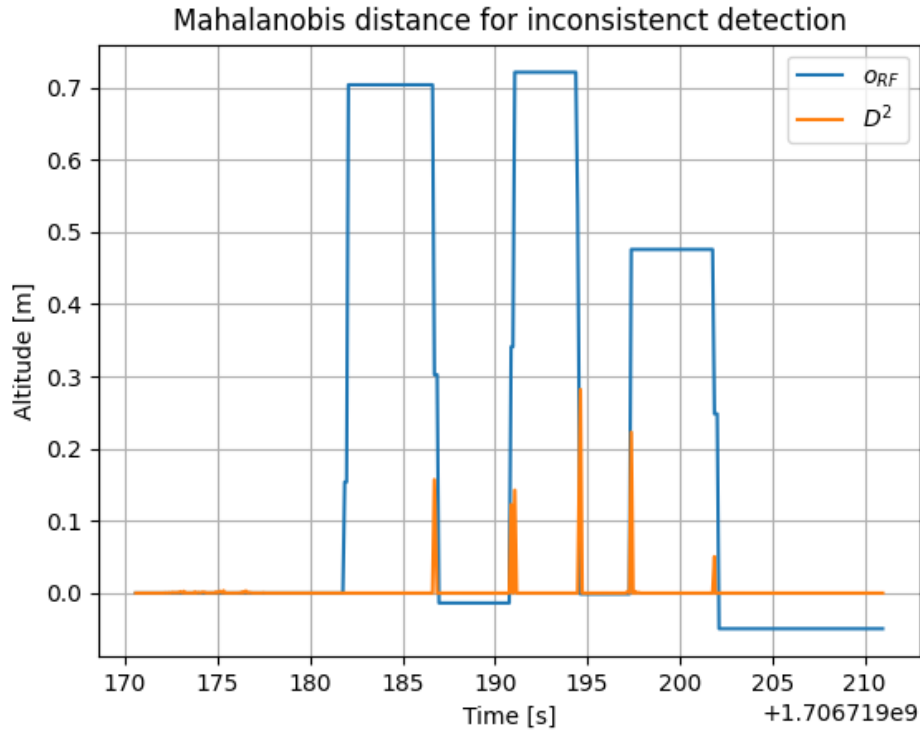


Fig. 4.4 Mahalanobis distance considerably increases when range finder inconsistencies occur.

considerably larger compared to the drift accumulated by the Adaptive Kalman filter estimate.

The experimental evaluation demonstrates that range measurements significantly improve altitude estimation compared to a purely vision-based algorithm experiencing performance degradation. Although further performance could be achieved by improving the estimation of the range finder offset, the presented methodology aims at demonstrating the effectiveness of the approach.

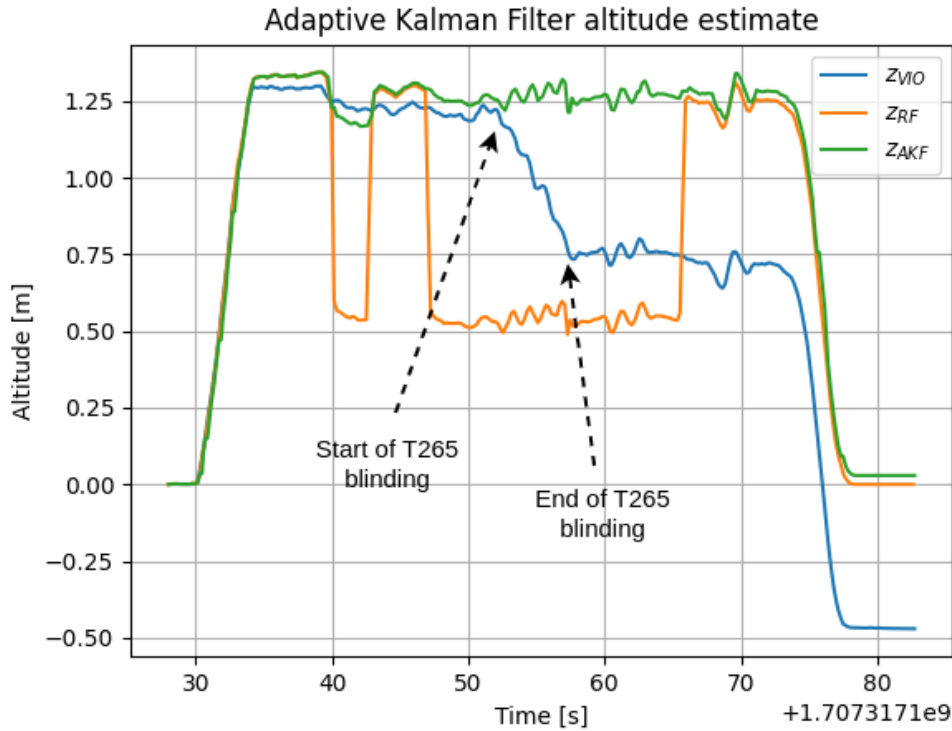


Fig. 4.5 Altitude estimated by the Adaptive Kalman Filter ensures minor drift compared to the T265 altitude.

4.3 Vision–LiDAR Cooperative Localization

4.3.1 Related Works

Cooperative localization is a strategy that aims to improve the localization of one robot in a formation by leveraging the data coming from other robots in the same formation. To accomplish this goal, many different approaches have been proposed. To obtain relative positioning, fiducial markers are a common solution in GPS-denied environments. The approaches in [88] and [89] propose methods to relatively localize robots by fusing fiducial marker detections coming from multiple robots. While requiring low computational resources, these methods are very sensitive to marker visibility and visual conditions, since these factors may compromise the detection and cause false positives. These methods also require installation on each robot or, even worse, on the surrounding environment, reducing their operability and feasibility.

Map alignment methods [85], [86], [87] rely on loop-closure detection across the maps built by different robots. These approaches effectively reduce the accumulated drift for the whole formation. Nevertheless, they require high computational complexity and strongly depend on the SLAM quality of each robot. Additionally, these methods are not suitable for heterogeneous robots, since they usually map different parts of the environment.

Approaches that utilize UWB technology are also popular in cooperative localization solutions. In [90], the authors propose a methodology to enhance the localization of two UAVs by fusing the data coming from UWB sensors and VIO. In [91], the authors show a similar approach but extended to UAV swarms. Other works employ LiDAR data to enhance UAV localization in cooperative formations. The authors in [102] propose an approach to combine the VIO-based localization of Micro-Aerial Vehicles (MAVs) with the LiDAR detections obtained by a LiDAR-equipped UAV. Their methodology considerably reduces the VIO drift, but detection failures may occur when the MAV is not well detected by the LiDAR. Since the proposed approach also relies on cameras, UAV detection can be performed even when only a few points of the drone are detected by the LiDAR. The authors of [103] show an approach based on UWB data and LiDAR point clouds for a UAV–UGV configuration. In their work, the UGV is equipped with a LiDAR utilized to detect the UAV. The UWB data is fused with the point-cloud position for an accurate detection and localization estimate. In [104], the authors propose a methodology to reduce UAV localization drift by leveraging the accurate localization of LiDAR-equipped UGVs. In their work, the relative observations between the aerial and ground platforms are fused within a joint estimation framework to bound the UAV drift. Different from the approach proposed in this chapter, the correction relies on implicit relative pose estimation rather than direct metric distance measurements.

Contribution

In this chapter, an approach to reduce the localization drift affecting UAVs relying on VIO-based algorithms is introduced. To the best of the author’s knowledge, the proposed approach is the first solution to handle UAV drift mitigation by fusing UAV detections across images and LiDAR point clouds. Since the methodology is based on geometric considerations, it results in being simpler and less demanding in terms of computational resources than map-based algorithms. Compared to UWB-based

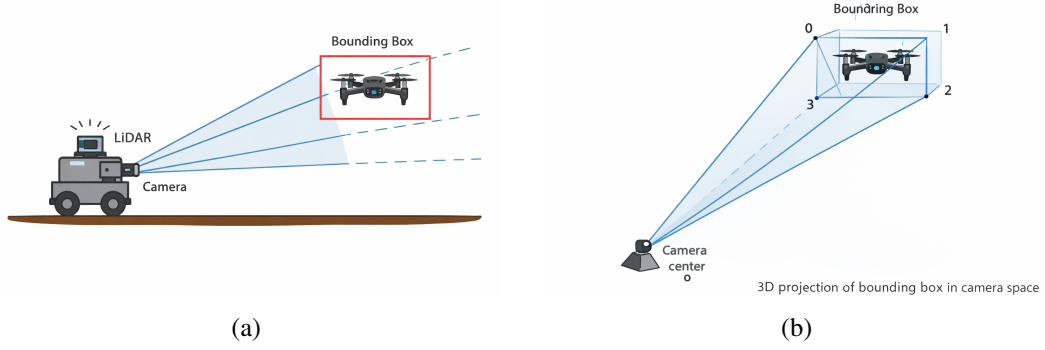


Fig. 4.6 Raycast of the UAV's bounding box. (a) illustrates the sensor configuration to detect the UAV bounding box, while (b) shows the pyramid obtained from back-projecting the bounding-box vertices.

methods, the proposed approach is minimal in terms of sensor configuration since it employs only LiDAR and cameras.

The presented methodology shows promising results in real-environment experiments. By estimating the relative distance between a UAV and a UGV, the approach addresses the issues related to UAV position drift in GPS-denied environments. Experimental results indicate that the error of the estimated distance remains below 10% when the UAV is detected at distances shorter than 3 meters.

4.3.2 UAV Raycasting with 3D-LiDAR Point Cloud

Let us consider a setup consisting of a UGV equipped with a 3D LiDAR and a camera and an UAV (Figure 4.6a).

Let $\mathcal{F}$ denote the reference frame in which both the camera center and the LiDAR point cloud are expressed.

Let $\mathbf{o} \in \mathbb{R}^3$ be the camera optical center in this frame. At a given time instant, an object detector provides a 2D bounding box around the UAV in the image. Let the bounding-box vertices be enumerated in clockwise order as

$$\{\mathbf{u}_i\}_{i=0}^3, \quad \mathbf{u}_0 = \text{top-left}, \mathbf{u}_1 = \text{top-right}, \mathbf{u}_2 = \text{bottom-right}, \mathbf{u}_3 = \text{bottom-left}.$$

Back-projection and Ray Construction

Each image vertex $\mathbf{u}_i = [u_i, v_i, 1]^\top$ is back-projected into a 3D ray direction using the camera intrinsic matrix $\mathbf{K}$:

$$\tilde{\mathbf{d}}_i = \mathbf{K}^{-1} \mathbf{u}_i, \quad \mathbf{d}_i = \frac{\tilde{\mathbf{d}}_i}{\|\tilde{\mathbf{d}}_i\|},$$

where $\mathbf{d}_i \in \mathbb{R}^3$ is a unit vector representing the direction of the ray associated with vertex i and $\tilde{\mathbf{d}}_i$ is its unnormalized quantity.

The ray corresponding to vertex i is defined as

$$\mathcal{R}_i = \{\mathbf{o} + s \mathbf{d}_i \mid s \geq 0\}.$$

where s denotes a scalar parameter representing the distance from the camera center along the ray direction.

Viewing Pyramid Side Planes

The four side faces of the viewing pyramid (Figure 4.6b) are defined by consecutive ray pairs

$$(0, 1), (1, 2), (2, 3), (3, 0).$$

For each pair (i, j) , the normal vector of the corresponding plane is computed as

$$\tilde{\mathbf{n}}_{ij} = \mathbf{d}_i \times \mathbf{d}_j, \quad \mathbf{n}_{ij} = \frac{\tilde{\mathbf{n}}_{ij}}{\|\tilde{\mathbf{n}}_{ij}\|}.$$

Each plane passes through the camera center $\mathbf{o}$ and is represented in implicit form as

$$f_{ij}(\mathbf{x}) = \mathbf{n}_{ij}^\top \mathbf{x} + d_{ij},$$

with offset

$$d_{ij} = -\mathbf{n}_{ij}^\top \mathbf{o}.$$

Normal Orientation Convention

The plane normals are required to point *inward* toward the interior of the viewing pyramid. Let $\mathbf{d}_c$ be the unit ray corresponding to the center of the bounding box. Each normal is oriented according to the rule.

$$\text{if } \mathbf{n}_{ij}^\top \mathbf{d}_c < 0, \quad \text{then } \mathbf{n}_{ij} \leftarrow -\mathbf{n}_{ij}.$$

After this operation, a positive value of the plane equation indicates that a point lies inside the pyramid half-space.

Point-in-Pyramid Test

Let $\mathbf{p} \in \mathbb{R}^3$ be a point from the LiDAR point cloud. The plane equation evaluated at $\mathbf{p}$ becomes

$$f_{ij}(\mathbf{p}) = \mathbf{n}_{ij}^\top \mathbf{p} + d_{ij} = \mathbf{n}_{ij}^\top (\mathbf{p} - \mathbf{o}).$$

According to the adopted sign convention, a point is considered *inside* the viewing pyramid if and only if

$$\mathbf{n}_{ij}^\top (\mathbf{p} - \mathbf{o}) > 0 \quad \forall (i, j) \in \{(0, 1), (1, 2), (2, 3), (3, 0)\}.$$

If for at least one plane the condition

$$\mathbf{n}_{ij}^\top (\mathbf{p} - \mathbf{o}) \leq 0$$

is satisfied, the point is classified as outside the pyramid. The points behind the camera can be rejected by enforcing:

$$\mathbf{d}_c^\top (\mathbf{p} - \mathbf{o}) > 0.$$

Distance Estimation

All LiDAR points satisfying the above constraints are considered to be inside the field of view of the camera. These points are then divided into equally spaced classes according to their Euclidean distance from the camera. All the contiguous classes

with non-zero points compose a cluster. The UAV-UGV distance is then estimated by evaluating the weighted mean of the classes' centers composing the cluster.

4.3.3 Experimental Evaluation

Hardware setup

The experimental data is collected using the UAV and UGV shown in Figure 4.7. In order to improve UAV detection, the drone was equipped with some red structural parts. The LiDAR employed is a Livox Mid-360, which has the field of view shown in Figure 4.8. The camera position was accurately measured with respect to the LiDAR to express each quantity in the same reference system. The camera used is an OV9712 with a field of view of $106^\circ/89^\circ/49^\circ$ (diagonal/horizontal/vertical) and has been employed with a resolution of 1080×720 . The camera was calibrated assuming the pinhole model; thus, its intrinsic values are available.



(a)



(b)

Fig. 4.7 The robots used for data collection: UAV (a) and UGC (b)



Fig. 4.8 The Livox Mid-360 (a) and its field of view (b)

Data collection

The approach proposed in 4.3.2 has been validated over three different datasets. In dataset 1 (D1), the UAV is stationary and placed at a known distance from the UGV. The UAV is fully visible within the camera's field of view. This dataset is used to verify the following aspects of the proposed method:

1. Correct detection of the UAV by the object detection module.
2. Correct identification of the LiDAR points lying inside the pyramid.
3. Verification that the points belonging to the first LiDAR hit cluster correspond to the UAV.
4. Comparison between the estimated UAV–UGV distance and the known ground-truth distance to evaluate the accuracy of the drift compensation.

It is worth noting that the third point cannot be rigorously verified, as the LiDAR provides only the spatial positions of the detected points and does not include semantic information about the observed objects. However, by visualizing the 3D point cloud, the main objects present in the scene can be clearly distinguished. This qualitative analysis allows us to visually confirm that the identified point cluster corresponds to the UAV, thereby providing supporting evidence of correct identification.

In dataset 2 (D2), the UAV has been manually moved to simulate small movements around a known distance from the UGV. The motion is intentionally limited to

both preserve the visibility of the UAV from the UGV-mounted camera and to keep the UAV approximately at the known distance, since the ground-truth measurements of the varying distance are not available. This dataset is used to verify the following aspects of the proposed method:

1. Correct detection of the UAV by the object detection module, even when the UAV moves.
2. Correct identification of the LiDAR points lying inside the pyramid.
3. Verification that the points belonging to the first LiDAR hit cluster correspond to the UAV, even when the UAV moves.
4. Verification that the estimated distance is around the ground-truth value measured when the UAV is stationary.

In dataset 3 (D3), the UAV has been manually moved as in D2, but the UAV was closer to the UGV. This dataset has been recorded to investigate how the detection is affected by the distance and how that could affect the measured distance.

4.3.4 Results

For all datasets, the following results are provided:

1. The ratio between the frames in which the UAV is successfully recognized over the total number of frames in which the UAV was visible
2. The error percentage of the estimated distance
3. The number of times the algorithm failed to recognize the cluster corresponding to the UAV was recorded

Additionally, for one of the three datasets, the following results are provided:

1. A visualization of the points evaluated by the algorithm to be inside the pyramid
2. A visualization of the points that the algorithm considers to belong to the UAV

Table 4.1 Comparison of the chosen metrics across the 3 datasets

Metric	Dataset		
	D1 (2.72 m)	D2 (2.72 m)	D3 (1.21 m)
Detections/Total	35/115	36/101	101/101
Error %	6%	-	-
Failures/Detections	1/35	2/36	0/101

Table 4.1 summarizes the results for all the datasets:

- Detection/Total quantifies how many frames the UAV has been successfully recognized by the object detection module over all the frames where the UAV was visible.
- Error% is defined as:

$$\text{Error\%} = \frac{|d_{\text{est}} - d_{\text{gt}}|}{d_{\text{gt}}} \times 100$$

where d_{est} and d_{gt} are the estimated distance and the ground-truth distance respectively. This quantity represents an average value over the frames in which the detection was successful, and the absolute error of the estimated distance was lower than 0.5 m

- Failure/Detection quantifies how many frames the estimated distance has an error greater than 0.5 m. This metric is very helpful because it highlights the cases in which the algorithm fails to correctly associate the LiDAR point cluster corresponding to the UAV.

The ground truth distance is reported in parentheses beside the dataset name.

Since in D2 and D3 the UAV is manually moved, and the ground-truth distance is not available, the Error% metric is no longer meaningful. For this reason, that metric has been omitted for those datasets. Instead, the Failure/Detection metric remains valid.

In Figure 4.10, a visualization of the results is provided. The LiDAR point cloud is represented by gray points. The red points correspond to those identified by the algorithm as being inside the pyramid, while the blue points belong to the first-hit cluster from which the distance is evaluated. The rays resulting from the back-

projected vertices of the bounding box are in green, while the estimated distance vector is represented in cyan.

Then, Figure 4.9 illustrates one image frame for each dataset with the detected bounding box. The red bounding box corresponds to the original detection output, while the blue bounding box represents a manually reduced region. The latter is defined to avoid including points within the field of view that do not belong to the UAV.

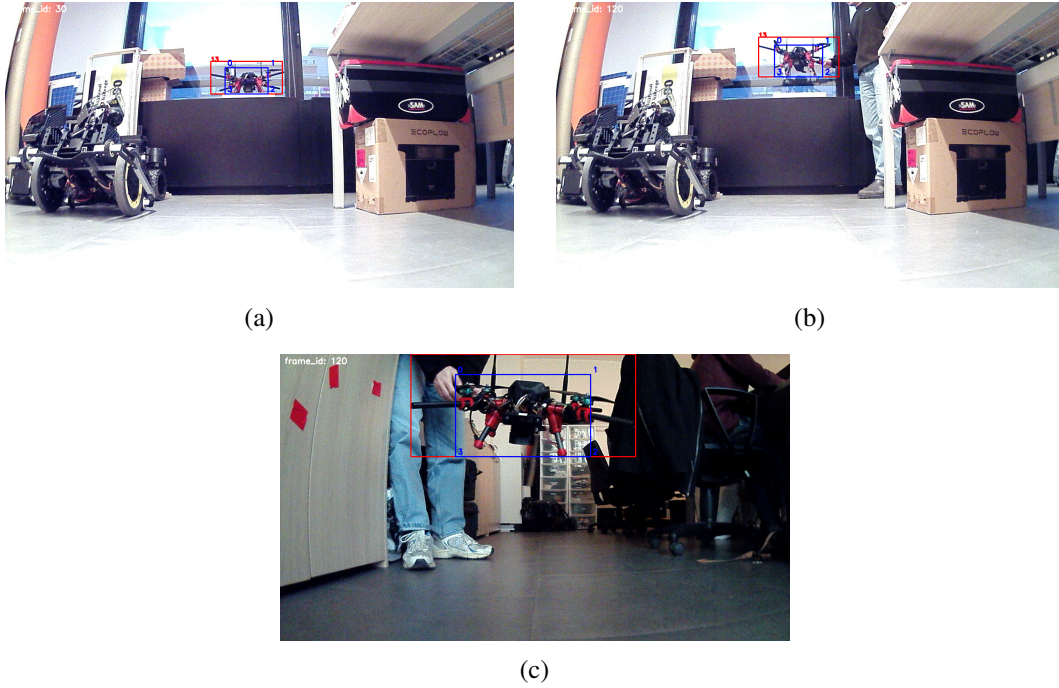
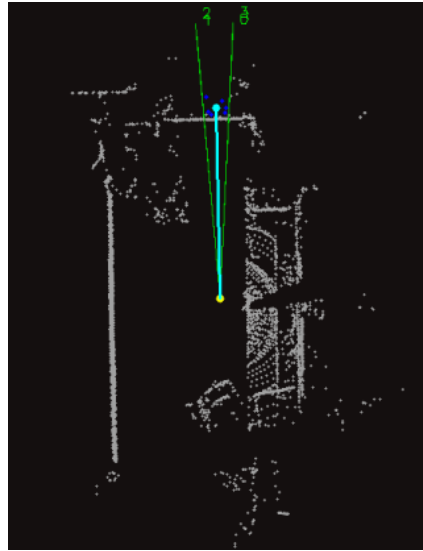
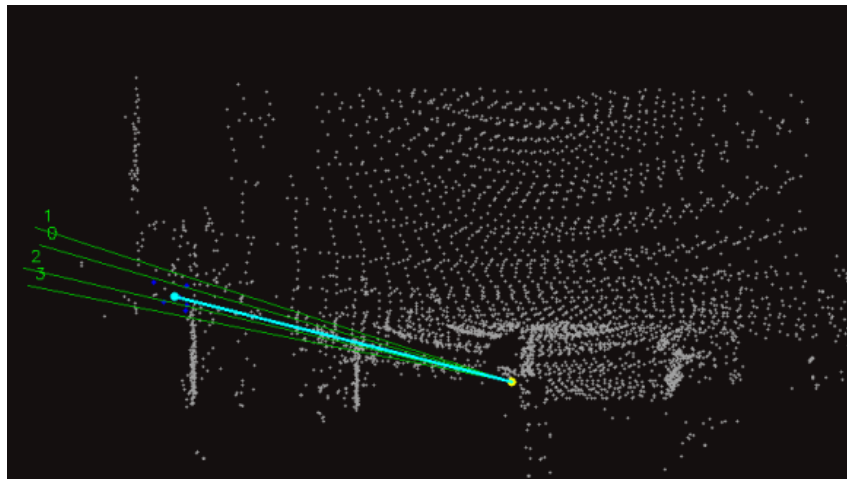


Fig. 4.9 Detection of the UAV: the figure shows, for the three datasets, a frame where the UAV was successfully recognized. The bounding box obtained directly by the object detection module is in red, while the manually reduced bounding box is in blue. (a) refers to D1, (b) to D2 and (c) to D3.



(a)



(b)

Fig. 4.10 3D visualization of the results: the figure shows, for the dataset D2, the LiDAR point cloud (gray points), the points inside the pyramid (red points), the cluster estimated to correspond to the UAV (blue points), the rays obtained from the back-projection of the bounding box vertices (green lines), and the distance vector from the camera to the first hit cluster (cyan line). (a) Illustrates the projection on the horizontal plane, while (b) presents the full 3D visualization of the scene.

4.3.5 Discussion

From Table 4.1, the UAV detection demonstrates to be very reliable on the D3 dataset with 100% successful recognition. Even if this dataset has been recorded at a lower distance (1.2 m), the drone has been moved in that dataset. Therefore, the obtained result is quite remarkable. On the other hand, the detection success of approximately 35% across D1 and D2 leaves room for improvement. The reason why the detection failed for those datasets is probably imputable to the lack of training data of UAV photos similar to the images collected in those datasets. For real-world scenarios, in which better performance might be desired, the training dataset should be collected more rigorously. In particular, it should include numerous images of the UAV acquired at different distances and from different viewpoints. Furthermore, the recognition process could be facilitated by equipping the UAV with a visual marker or another distinctive feature, such as a high-contrast colour pattern. However, the successful frames are enough to demonstrate the effectiveness of the algorithm.

The error of the estimated distance is around 6% for D1. Considering that the ground-truth distance for this dataset is 2.72 m, it means that the algorithm has committed an error of approximately 16 cm in the distance estimate. Since the Livox Mid-360 should have a precision of around 3 cm according to the specifications, the remaining error is mainly attributed to the cluster identification. In particular, the following aspects have to be taken into consideration:

- The identified cluster may include a few points not belonging to the UAV.
- The center of the cluster, computed by a weighted mean of adjacent bins, is only an approximation of the position of the UAV's center of gravity.
- The calibration of the camera and its relative position with respect to the LiDAR is inevitably affected by small errors.

A fine aspect of the proposed approach is the number of failure detection occurrences. The metric Failures/Detections basically indicates whether the algorithm identifies another cluster as corresponding to the UAV. Across the three datasets, this situation occurs only 3 times and proves the algorithm's effectiveness and reliability. However, it is expected that in some circumstances, the failure occurrences may increase. That might happen, for example, when the drone is close to surfaces that may be wrongly identified as the first cluster. However, knowing approximately the

UAV position from the VIO algorithm running onboard the drone, wrongly estimated distances can be easily filtered.

Since the drift accumulated by a VIO algorithm may easily exceed 1 m of error, even assuming a 10% error for the estimated distance would result in a considerable improvement in terms of position accuracy.

Chapter 5

Conclusions

GNSS-denied environments represent challenging scenarios for multi-robot cooperation. Several works identify localization as the most critical aspect and propose navigation solutions to overcome current limitations [8–14, 16–22]. Although research has developed promising localization systems over the last two decades, the identification of a comprehensive and reliable solution remains an open challenge.

Having analyzed UAV–UGV multi-robot formations and their most relevant applications in GNSS-denied environments, the choice of the sensor setup for RW-UAVs was identified as a critical aspect in meeting the numerous requirements related to weight, size, power consumption, and computational resources. The detailed discussion shows that vision-based navigation systems represent the most convenient solution for satisfying these requirements (see Figure 5.1). Beyond localization, visual data is necessary in all the application scenarios considered. Therefore, whenever a camera must be included in the sensor setup to satisfy mission requirements, it becomes attractive to exploit the same sensor for navigation purposes. The detailed analysis of vision-based navigation provided in Chapter 2 shows the current limitations of VIO algorithms and highlights the need for further improvements.

This dissertation introduces novel methodologies to enhance the localization performance of UAVs relying on vision-based navigation systems. The proposed approaches address three different limitations of state-of-the-art VIO frameworks:

- The lack of robustness for monocular setups
- The failures of visual-based localization under insufficient feature tracking

- The large drift accumulation over long trajectories

The first two aspects are addressed by analyzing state-of-the-art pipelines to identify the conditions under which localization performance degrades. The proposed solutions are integrated into these pipelines, and an experimental evaluation is performed to validate their implementation.

The issue of drift accumulation over long trajectories is addressed through sensor-fusion techniques. To avoid significant weight increases, the sensors included in the setup are selected according to their size, weight, and power consumption. For these reasons, the combination of VIO pipelines with rangefinder measurements represents an effective solution. Although the introduction of this sensor only addresses altitude-drift reduction, the proposed approach still improves position accuracy along the vertical direction.

In the context of UAV-UGV multi-robot cooperation, a sensor fusion technique leveraging the UGV sensor measurements is particularly attractive because it avoids introducing additional sensors on the aerial platform. The experimental results show promising performance in reducing the drift accumulated by vision-based algorithms over long trajectories. Although failures may still occur in real-world scenarios, outliers in distance estimates can be filtered out using simple and widely adopted methodologies.

Except for the multi-camera implementation, the solutions proposed in this dissertation introduce negligible computational overhead. Although the proposed multi-camera setup substantially increases the processing requirements, the experimental results demonstrate its applicability on the embedded computing platforms typically employed in these applications. Therefore, the proposed solutions can be replicated without requiring specialized hardware or incurring a significant increase in system cost or complexity.

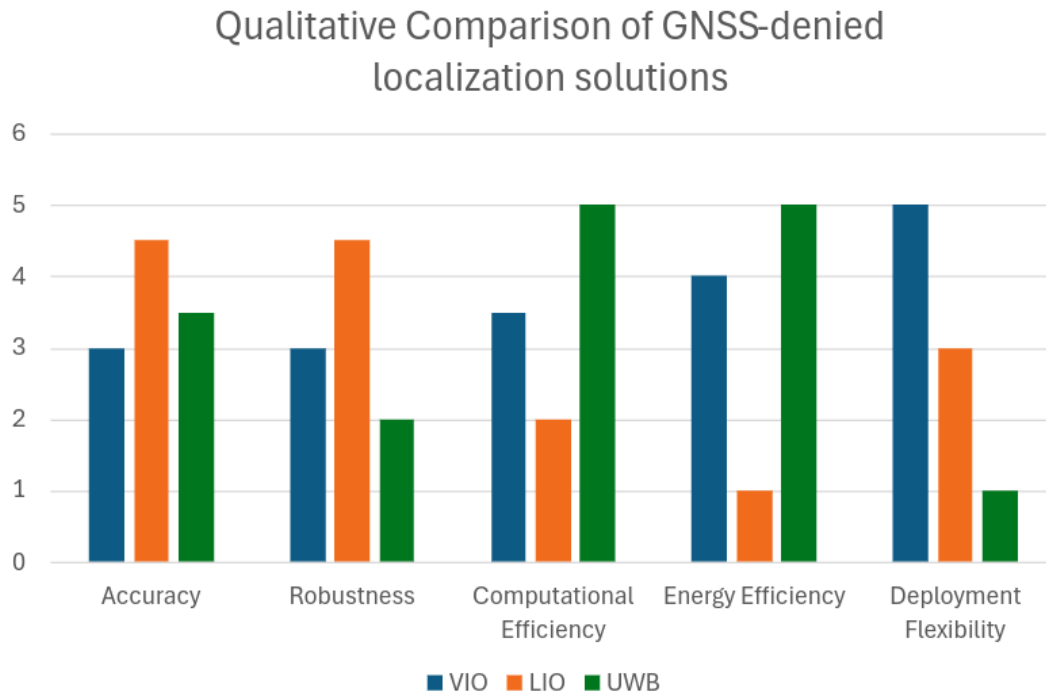


Fig. 5.1 The diagram qualitatively compares the three localization solutions across key performance metrics. Scores range from 1 (least favorable) to 5 (most favorable). A detailed explanation of the figure is provided in 1.4.4.

In this dissertation, infrastructure-based localization systems have not been considered as a solution to the problem, as these systems require the deployment of dedicated installations within the environment. In some real-world scenarios, these installations are unfeasible, while in others they are excessively expensive. In this context, UAV–UGV cooperation may be employed to overcome these limitations by attaching anchors to the UGV and utilizing them to localize the drone with respect to the rover. This solution is proposed in [105–109]. The effectiveness of this approach is typically related to the assembly configuration, which is highly constrained on a UGV platform. For proper trilateration and good observability, the anchors require large baselines and non-coplanar configurations. Additionally, multi-path and non-line-of-sight occurrences within the limited operative range of around 50–100 m must be considered in real-world scenarios. For these reasons, this solution cannot operate as a standalone system.

However, the methodologies introduced in this dissertation can be utilized in combination with UWB systems to further improve localization performance. A configuration in which the UAV relies on a monocular VIO algorithm in combination

with UWB, whose anchors are attached to a LiDAR-based navigating UGV, might be extremely convenient. In such a configuration, the VIO pipeline provides real-time navigation with reasonable computational costs and minimal installation constraints. Therefore, the UAV can be considerably reduced in size, improving flight endurance and minimizing power consumption. The UWB system would operate as a secondary localization source with the primary objective of reducing drift in position estimation and mitigating the failures of monocular VIO algorithms.

Although this sensor configuration appears extremely promising from the above discussion, it poses several challenges in its development. One of the most relevant issues is related to the minimal baseline required by UWB systems to provide reliable estimates. Since the UAV is assumed to be small in this configuration, the tag displacement would be limited by its reduced dimensions, compromising localization accuracy. This aspect could be investigated by developing advanced techniques for signal trilateration or by employing sensor fusion approaches such as the one provided in Chapter 4. Moreover, since the assumption of exact position for the UGV may not hold in certain scenarios, collaborative localization algorithms might improve overall system localization. Another relevant aspect is the integration of a proper filter to handle these multiple sources of position information. To provide an accurate and robust solution, a rigorous formulation is required, without ignoring the need to bound computational complexity.

As extensively discussed throughout this dissertation, collaborative autonomous robotic systems operating in GNSS-denied environments represent a highly promising solution for a wide range of applications. It is expected that the considerations and findings presented in this work will inspire further research in this direction.

References

- [1] Han Liang, Seong-Cheol Lee, Woosung Bae, Jeongyun Kim, and Suyoung Seo. Towards UAVs in construction: Advancements, challenges, and future directions for monitoring and inspection. *Drones*, 7(3):202, 2023.
- [2] Ridha Guebsi, Sonia Mami, and Karem Chokmani. Drones in precision agriculture: A comprehensive review of applications, technologies, and challenges. *Drones*, 8(11):686, 2024.
- [3] Adam C. Watts, Vincent G. Ambrosia, and Everett A. Hinkley. Unmanned aircraft systems in remote sensing and scientific research: Classification and considerations of use. *Remote Sensing*, 4(6):1671–1692, 2012.
- [4] Ismael Colomina and Pere Molina. Unmanned aerial systems for photogrammetry and remote sensing: A review. *ISPRS Journal of Photogrammetry and Remote Sensing*, 92:79–97, 2014.
- [5] Farah Syazwani Shahar, Mohamed Tariq Hameed Sultan, Marek Nowakowski, and Andrzej Łukaszewicz. UGV-UAV integration advancements for coordinated missions: A review. *Journal of Intelligent & Robotic Systems*, 111:69, 2025.
- [6] Haojun Luo and Chih-Yung Wen. A low-cost relative positioning method for UAV/UGV coordinated heterogeneous system based on visual-LiDAR fusion. *Aerospace*, 10(11):924, 2023.
- [7] Alberto Pretto, Stéphanie Aravecchia, Wolfram Burgard, Nived Chebrolu, Christian Dornhege, Tillmann Falck, Freya Fleckenstein, Alessandra Fontenla, Marco Imperoli, Raghav Khanna, Frank Liebisch, Philipp Lottes, Andres Milioto, Daniele Nardi, Sandro Nardi, Johannes Pfeifer, Marija Popović, Ciro Potena, Cédric Pradalier, Elisa Rothacker-Feder, Inkyu Sa, Alexander Schaefer, Roland Siegwart, Cyrill Stachniss, Achim Walter, Wera Winterhalter, Xiaolong Wu, and Juan Nieto. Building an aerial-ground robotics system for precision farming: An adaptable solution. *IEEE Robotics & Automation Magazine*, 28(3):29–49, 2021.
- [8] Marius Beul, David Droeschel, Matthias Nieuwenhuisen, Jan Quenzel, Sebastian Houben, and Sven Behnke. Fast autonomous flight in warehouses for inventory applications. *IEEE Robotics and Automation Letters*, 3(4):3121–3128, October 2018.

- [9] T. M. Fern'andez-Caram'es, O. Blanco-Novoa, I. Froiz-M'iguez, and P. Fraga-Lamas. Towards an autonomous Industry 4.0 warehouse: A UAV and blockchain-based system for inventory and traceability applications in big data-driven supply chain management. *Sensors*, 19(10):2394, May 2019.
- [10] Mauro S'ergio Mafra Moreira, Daniel Khede Dourado Villa, and M'ario Sarcinelli-Filho. Controlling a virtual structure involving a uav and a ugv for warehouse inventory. *Journal of Intelligent & Robotic Systems*, 110(3):121, 2024.
- [11] Caio CG Ribeiro, Leonardo HMC dos Santos, and Douglas G Macharet. Collaborative ugv/uav path planning for inventory management in warehouses. In *2022 Latin American Robotics Symposium (LARS), 2022 Brazilian Symposium on Robotics (SBR), and 2022 Workshop on Robotics in Education (WRE)*, pages 121–126. IEEE, 2022.
- [12] Bernardo Martinez Rocamora, Rog'erio R. Lima, Kieren Samarakoon, Jeremy Rathjen, Jason N. Gross, and Guilherme A. S. Pereira. Oxpecker: A tethered uav for inspection of stone-mine pillars. *Drones*, 7(2), 2023.
- [13] David Akhiehiero, Uthman Olawoye, Shounak Das, and Jason Gross. Cooperative localization for gnss-denied subterranean navigation: A uav–ugv team approach. *NAVIGATION: Journal of the Institute of Navigation*, 71(4), 2024.
- [14] Jason Gross, Matteo De Petrillo, Jared Beard, Hayden Nichols, Tommy Swiger, Ryan Watson, Connor Kirk, Cagri Kilic, Jacob Hikes, Emily Upton, et al. Field-testing of a uav-ugv team for gnss-denied navigation in subterranean environments. In *Proceedings of the 32nd International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2019)*, pages 2112–2124, 2019.
- [15] Viktor L. Orekhov and Timothy H. Chung. The darpa subterranean challenge: A synopsis of the circuits stage. *Field Robotics*, 2:735–747, 2022.
- [16] Marco Tranzatto, Frank Mascarich, Lukas Bernreiter, Carolina Godinho, Marco Camurri, Shehryar Khattak, Tung Dang, Victor Reijgwart, Johannes L'öje, David Wisth, Samuel Zimmermann, Huan Nguyen, Marius Fehr, Lukas Solanka, Russell Buchanan, Marko Bjelonic, Nikhil Khedekar, Mathieu Valsecchini, Fabian Jenelten, Mihir Dharmadhikari, Timon Homberger, Paolo De Petris, Lorenz Wellhausen, Mihir Kulkarni, Takahiro Miki, Satchel Hirsch, Markus Montenegro, Christos Papachristos, Fabian Tresoldi, Jan Carius, Giorgio Valsecchi, Joonho Lee, Konrad Meyer, Xiangyu Wu, Juan Nieto, Andy Smith, Marco Hutter, Roland Siegwart, Mark Mueller, Maurice Fallon, and Kostas Alexis. CERBERUS: Autonomous legged and aerial robotic exploration in the tunnel and urban circuits of the DARPA subterranean challenge. *Field Robotics*, 2(1):274–324, 2022.
- [17] Nicolas Hudson, Fletcher Talbot, Mark Cox, Jason Williams, Thomas Hines, Alex Pitt, Brett Wood, Dennis Frousheger, Katrina Lo Surdo, Thomas Molnar,

- Ryan Steindl, Matt Wildie, Inkyu Sa, Navinda Kottege, Kazys Stepanas, Emili Hernandez, Gavin Catt, William Docherty, Brendan Tidd, Benjamin Tam, Simon Murrell, Mitchell Bessell, Lauren Hanson, Lachlan Tychsen-Smith, Hajime Suzuki, Leslie Overs, Farid Kendoul, Glenn Wagner, Duncan Palmer, Peter Milani, Matthew O'Brien, Shu Jiang, Shengkang Chen, and Ronald C. Arkin. Heterogeneous ground and air platforms, homogeneous sensing: Team CSIRO Data61's approach to the DARPA subterranean challenge. *Field Robotics*, 2(1):595–636, 2022.
- [18] Mihir Kulkarni, Mihir Dharmadhikari, Marco Tranzatto, Samuel Zimmermann, Victor Reijgwart, Paolo De Petris, Huan Nguyen, Nikhil Khedekar, Christos Papachristos, Lionel Ott, Roland Siegwart, Marco Hutter, and Kostas Alexis. Autonomous teamed exploration of subterranean environments using legged and aerial robots. In *2022 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3306–3313. IEEE, 2022.
- [19] Aiden Morrison, Nadezda Sokolova, and Erik Holthe Eriksen. Collaborative navigation for defence and emergency services: Augmenting team navigation and situational awareness. *European Journal of Navigation*, 13(3):17–24, 2015.
- [20] Victor O. Sivaneri and Jason N. Gross. UGV-to-UAV cooperative ranging for robust navigation in GNSS-challenged environments. *Aerospace Science and Technology*, 71:245–255, 2017.
- [21] Chenfa Shi, Zhi Xiong, Mingxing Chen, Rong Wang, and Jun Xiong. Cooperative navigation for heterogeneous air-ground vehicles based on interoperation strategy. *Remote Sensing*, 15(8):2006, 2023.
- [22] David Akhihero, Uthman Olawoye, Shounak Das, and Jason N. Gross. Cooperative localization for GNSS-denied subterranean navigation: A UAV–UGV team approach. *NAVIGATION: Journal of the Institute of Navigation*, 71(4):navi.677, 2024.
- [23] Gregg Rabideau, Joseph Russino, Andrew Branch, Nihal Dhamani, Tiago Stegun Vaquero, Steve Chien, Jean-Pierre de la Croix, and Federico Rossi. Planning, scheduling, and execution on the moon: The CADRE technology demonstration mission. In *Proceedings of the 24th International Conference on Autonomous Agents and Multiagent Systems*, pages 1727–1735, 2025.
- [24] NASA Jet Propulsion Laboratory. Nasa performs first aircraft accident investigation on another world, 2024. Accessed: 2025-04-07.
- [25] Imen Jarraya, Abdulrahman Al-Batati, Muhammad Bilal Kadri, Mohamed Abdelkader, Adel Ammar, Wadii Boulila, and Anis Koubaa. Gns-denied unmanned aerial vehicle navigation: Analyzing computational complexity, sensor fusion, and localization methodologies. *Satellite Navigation*, 6(1):9, 2025.

- [26] Saleh Alghamdi, Sultan Alahmari, Saud Yonbawi, Khalid Alsaleem, Fahad Ateeq, and Faris Almushir. Autonomous navigation systems in gps-denied environments: a review of techniques and applications. In *2025 11th International Conference on Automation, Robotics, and Applications (ICARA)*, pages 290–299. IEEE, 2025.
- [27] Mahmoud Elsanhoury, Janne Koljonen, Petri Välisuo, Mohammed Elmusrati, and Heidi Kuusniemi. Survey on recent advances in integrated gnss towards seamless navigation using multi-sensor fusion technology. In *Proceedings of the 34th international technical meeting of the satellite division of the institute of navigation (ION GNSS+ 2021)*, pages 2754–2765, 2021.
- [28] D. Strelow and S. Singh. Optimal motion estimation from visual and inertial measurements. In *Sixth IEEE Workshop on Applications of Computer Vision, 2002. (WACV 2002). Proceedings.*, pages 314–319, 2002.
- [29] J. Lobo and J. Dias. Vision and inertial sensor cooperation using gravity as a vertical reference. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(12):1597–1608, 2003.
- [30] Anastasios I. Mourikis and Stergios I. Roumeliotis. A multi-state constraint kalman filter for vision-aided inertial navigation. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pages 3565–3572, 2007.
- [31] Stefan Leutenegger, Simon Lynen, Michael Bosse, Roland Siegwart, and Paul Furgale. Keyframe-based visual-inertial odometry using nonlinear optimization. *The International Journal of Robotics Research*, 34, 02 2014.
- [32] Tong Qin, Peiliang Li, and Shaojie Shen. Vins-mono: A robust and versatile monocular visual-inertial state estimator. *IEEE Transactions on Robotics*, 34(4):1004–1020, 2018.
- [33] Patrick Geneva, Kevin Eickenhoff, Woosik Lee, Yulin Yang, and Guoquan Huang. Opencvins: A research platform for visual-inertial estimation. *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4666–4672, 2020.
- [34] Michael Bloesch, Sammy Omari, Marco Hutter, and Roland Siegwart. Robust visual inertial odometry using a direct ekf-based approach. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 298–304, 2015.
- [35] V. Usenko, J. Engel, J. Stückler, and D. Cremers. Visual-inertial mapping with non-linear factor recovery. *IEEE Robotics and Automation Letters*, 5(2):422–429, 2020.
- [36] Antoni Rosinol, Marcus Abate, Yun Chang, and Luca Carlone. Kimera: an open-source library for real-time metric-semantic localization and mapping. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2020.

- [37] Mingyang Li and Anastasios I. Mourikis. High-precision, consistent ekf-based visual-inertial odometry. *The International Journal of Robotics Research*, 32:690 – 711, 2013.
- [38] J. Delmerico and D. Scaramuzza. A benchmark comparison of monocular visual-inertial odometry algorithms for flying robots. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2502–2509, Brisbane, Australia, 2018.
- [39] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. M. Montiel, and J. D. Tardós. Orb-slam3: An accurate open-source library for visual, visual-inertial, and multi-map slam. *IEEE Transactions on Robotics*, 37(6):1874–1890, 2021.
- [40] Stefan Leutenegger. Okvis2: Realtime scalable visual-inertial slam with loop closure. *arXiv preprint arXiv:2202.09199*, 2022.
- [41] Lukas von Stumberg and Daniel Cremers. Dm-vio: Delayed marginalization visual-inertial odometry. *IEEE Robotics and Automation Letters*, 7(2):1408–1415, April 2022.
- [42] R. Mur-Artal and J. D. Tardós. Visual-inertial monocular slam with map reuse. *IEEE Robotics and Automation Letters*, 2(2):796–803, 2017.
- [43] Paul J. Besl and Neil D. McKay. A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, 1992.
- [44] Yang Chen and Gérard Medioni. Object modeling by registration of multiple range images. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 2724–2729. IEEE, 1992.
- [45] Peter Biber and Wolfgang Straßer. The normal distributions transform: A new approach to laser scan matching. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2743–2748. IEEE, 2003.
- [46] Ji Zhang and Sanjiv Singh. Loam: Lidar odometry and mapping in real-time. In *Robotics: Science and Systems (RSS)*, 2014.
- [47] Wei Xu, Yixi Cai, Dongjiao He, Jiarong Lin, and Fu Zhang. Fast-lío: A fast, robust lidar-inertial odometry package. *IEEE Robotics and Automation Letters*, 6(2):3317–3324, 2021.
- [48] Patrick Geneva, Kevin Eickenhoff, and Guoquan Huang. Lins: A lidar-inertial state estimator for robust and efficient navigation. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8899–8906. IEEE, 2019.

- [49] Tixiao Shan and Brendan Englot. Lego-loam: Lightweight and ground-optimized lidar odometry and mapping on variable terrain. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4758–4765. IEEE, 2018.
- [50] Ji Zhang and Sanjiv Singh. Visual-lidar odometry and mapping: Low-drift, robust, and fast. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 2174–2181. IEEE, 2015.
- [51] Chao Qin, Haoyang Ye, Christian E. Pranata, Jun Han, and Ming Liu. LINS: A lidar-inertial state estimator for robust and fast navigation. *CoRR*, abs/1907.02233, 2019.
- [52] Tixiao Shan and Brendan Englot. Lio-sam: Tightly-coupled lidar inertial odometry via smoothing and mapping. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5135–5142. IEEE, 2020.
- [53] Wei Xu, Yixi Cai, Dongjiao He, Jiarong Lin, and Fu Zhang. Fast-lio2: Fast direct lidar-inertial odometry. *IEEE Transactions on Robotics*, 38(4):2053–2073, 2022.
- [54] Jiarong Lin, Wei Xu, and Fu Zhang. R2live: A robust, real-time, lidar-inertial-visual tightly-coupled state estimator and mapping. *IEEE Robotics and Automation Letters*, 6(4):7469–7476, 2021.
- [55] Sinan Gezici. A survey on wireless position estimation. *Wireless Personal Communications*, 44(3):263–282, 2008.
- [56] Cung Lian Sang, Michael Adams, Marc Hesse, and Ulrich Rückert. Bidirectional uwb localization: A review on an elastic positioning scheme for gnss-deprived zones. *IEEE Journal of Indoor and Seamless Positioning and Navigation*, 1:161–179, 2023.
- [57] Anton Ledergerber, Michael Hamer, and Raffaello D’Andrea. A robot self-localization system using one-way ultra-wideband communication. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3131–3137. IEEE, 2015.
- [58] Jing Yang, BaiShun Dong, and Jiliang Wang. Vuloc: Accurate uwb localization for countless targets without synchronization. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 6(3):1–25, 2022.
- [59] Sungtae Moon and Wonkeun Youn. A novel movable uwb localization system using uavs. *IEEE Access*, 10:41303–41312, 2022.
- [60] Ziang Niu, Huizhen Yang, Lei Zhou, Mohamed Farag Taha, Yong He, and Zhengjun Qiu. Deep learning-based ranging error mitigation method for uwb localization system in greenhouse. *Computers and electronics in agriculture*, 205:107573, 2023.

- [61] Siyu He, Shin-Heng Cheng, Jiming Chen, and Youxian Sun. Wifi fingerprint-based indoor positioning: Recent advances and comparisons. *IEEE Communications Surveys & Tutorials*, 18(1):466–490, 2016.
- [62] Ghazaleh Panahandeh, Chao X. Guo, Magnus Jansson, and Stergios I. Roumeliotis. Observability analysis of a vision-aided inertial navigation system using planar features on the ground. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4187–4194, 2013.
- [63] Dimitrios Kottas, Joel Hesck, Sean Bowman, and Stergios Roumeliotis. On the consistency of vision-aided inertial navigation. *Springer Tracts in Advanced Robotics*, 88, 01 2013.
- [64] John W. Langelaan. *State Estimation for Autonomous Flight in Cluttered Environments*. PhD thesis, Stanford University, Stanford, CA, USA, 2006.
- [65] Dennis Strelow. *Motion Estimation from Image and Inertial Measurements*. PhD thesis, Carnegie Mellon University, Pittsburgh, PA, USA, 2004.
- [66] L. L. Ong, M. Ridley, J. H. Kim, E. Nettleton, and S. Sukkarieh. Six dof decentralised slam. In *Proceedings of the Australasian Conference on Robotics and Automation (ACRA)*, pages 10–16, Brisbane, Australia, December 2003.
- [67] Ethan Eade and Tom Drummond. Scalable monocular slam. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 469–476, June 2006.
- [68] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza. On-manifold preintegration for real-time visual-inertial odometry. *IEEE Transactions on Robotics*, 32(6):1309–1322, 2016.
- [69] Frank Dellaert. Factor graphs and gtsam: A hands-on introduction. 2012.
- [70] Anders Heyden and Marc Pollefeys. Multiple view geometry. In Nikos Paragios, Yunmei Chen, and Olivier Faugeras, editors, *Emerging Topics in Computer Vision*, pages 1–26. Springer, 2005.
- [71] Laurent Kneip, Stephan Weiss, and Roland Siegwart. Deterministic initialization of metric state estimation filters for loosely-coupled monocular vision-inertial systems. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2235–2241, September 2011.
- [72] Tong Qin and Shaojie Shen. Robust initialization of monocular visual-inertial estimation on aerial robots. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4225–4232, 2017.
- [73] J. Wu, J. Xiong, and H. Guo. Improving robustness of line features for vio in dynamic scenes. *Measurement Science and Technology*, 33(6):065204, 2022.

- [74] D. Yang, H. Liu, X. Jin, J. Chen, C. Wang, X. Ding, and K. Xu. Enhancing vio robustness under sudden lighting variation: A learning-based imu dead-reckoning for uav localization. *IEEE Robotics and Automation Letters*, 9(5):4535–4542, 2024.
- [75] T. Oskiper, Z. Zhu, S. Samarasekera, and R. Kumar. Visual odometry system using multiple stereo cameras and inertial measurement unit. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, Minneapolis, MN, USA, 2007.
- [76] S. Urban and S. Hinz. Multicol-slam – a modular real-time multi-camera slam system, 2016. arXiv:1610.07336.
- [77] P. Kaveti, S. N. Vaidyanathan, A. T. Chelvan, and H. Singh. Design and evaluation of a generic visual slam framework for multi camera systems. *IEEE Robotics and Automation Letters*, 8:7368–7375, 2023.
- [78] L. Zhang, D. Wisth, M. Camurri, and M. Fallon. Balancing the budget: Feature selection and tracking for multi-camera visual-inertial odometry. *IEEE Robotics and Automation Letters*, 7(2):1182–1189, 2022.
- [79] S. Li, L. Pang, and X. Hu. Multicam-slam: Non-overlapping multi-camera slam for indirect visual localization and navigation, 2024. arXiv:2406.06374.
- [80] S. Yang, S. A. Scherer, X. Yi, and A. Zell. Multi-camera visual slam for autonomous navigation of micro aerial vehicles. *Robotics and Autonomous Systems*, 93:116–134, 2017.
- [81] H. Yu, J. Wang, Y. He, W. Yang, and G.-S. Xia. Mcvo: A generic visual odometry for arbitrarily arranged multi-cameras, 2025. arXiv:2412.03146.
- [82] Y. He, H. Yu, W. Yang, and S. Scherer. Toward efficient and robust multiple camera visual-inertial odometry, 2021. arXiv:2109.12030.
- [83] Danpeng Chen, Nan Wang, Runsen Xu, Weijian Xie, Hujun Bao, and Guofeng Zhang. Rnin-vio: Robust neural inertial navigation aided visual-inertial odometry in challenging scenes. In *ISMAR*, pages 275–283, 2021.
- [84] Tali Orlev Shapira and Itzik Klein. Icd-net: Inertial covariance displacement network for drone visual-inertial slam, 2025.
- [85] Monica Ballesta, Arturo Gil, Óscar Reinoso, Miguel Juliá, and Luis Jiménez. Multi-robot map alignment in visual slam. *WSEAS TRANSACTIONS on SYSTEMS*, 9:213–222, 02 2010.
- [86] Ergin Ozkucur and H. Levent Akin. Cooperative multi-robot map merging using fast-slam. In *RoboCup 2009: Robot Soccer World Cup XIII*, volume 5949 of *Lecture Notes in Computer Science*, pages 449–460. Springer, 2010.

- [87] Carlos Alberto Velásquez Hernández and Flavio Augusto Prieto Ortiz. A real-time map merging strategy for robust collaborative reconstruction of unknown environments. *Expert Systems with Applications*, 145:113109, 2020.
- [88] Raul Acuna, Zaijuan Li, and Volker Willert. MOMA: visual mobile marker odometry. *CoRR*, abs/1704.02222, 2017.
- [89] Nikolaos Evangeliou and Anthony Tzes. Visual collaboration for leader-follower uav formation. *Frontiers in Robotics and AI*, 8, 2022.
- [90] Thien Hoang Nguyen, Thien-Minh Nguyen, and Lihua Xie. Flexible and resource-efficient multi-robot collaborative visual-inertial-range localization. *IEEE Robotics and Automation Letters*, 7(2):928–935, 2022.
- [91] Thomas Ziegler, Marco Karrer, Patrik Schmuck, and Margarita Chli. Distributed formation estimation via pairwise distance measurements. *IEEE Robotics and Automation Letters*, 6(2):3017–3024, 2021.
- [92] Anders Grunnet-Jepsen, Mark Harville, Brian Fulkerson, David Piro, Steven Brook, and James Radford. Introduction to intel realsense visual slam and the t265 tracking camera. <https://dev.intelrealsense.com/docs/intel-realsensetm-visual-slam-and-the-t265-tracking-camera>, 2019. Intel RealSense Product Documentation. Accessed: 2026-02-26.
- [93] Michael Burri, Janosch Nikolic, Pascal Gohl, Thomas Schneider, Joern Rehder, Sammy Omari, Markus Achtelik, and Roland Siegwart. The euroc micro aerial vehicle datasets. *The International Journal of Robotics Research*, 35, 01 2016.
- [94] Bruce D. Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 674–679, Vancouver, Canada, 1981.
- [95] Jieru Zhao, Yang Li, Dada Hu, and Zhongcai Pei. Design on altitude control system of quad rotor based on laser radar. In *2016 IEEE international conference on aircraft utility systems (AUS)*, pages 105–109. IEEE, 2016.
- [96] Liu Yang, Hechuan Wang, Yousef El-Laham, José Ignacio Lamas Fonte, David Trillo Pérez, and Mónica F Bugallo. Indoor altitude estimation of unmanned aerial vehicles using a bank of kalman filters. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5455–5459. IEEE, 2020.
- [97] Hriday Bavle, Jose Luis Sanchez-Lopez, Alejandro Rodriguez-Ramos, Carlos Sampedro, and Pascual Campoy. A flight altitude estimator for multirotor uavs in dynamic and unstructured indoor environments. In *2017 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 1044–1051, 2017.

- [98] P. C. Mahalanobis. *Mahalanobis Distance*, pages 325–326. Springer, New York, NY, 2008.
- [99] Shinya Kawabata, Jae Hoon Lee, and Shingo Okamoto. Obstacle avoidance navigation using horizontal movement for a drone flying in indoor environment. In *2019 International Conference on Control, Artificial Intelligence, Robotics & Optimization (ICCAIRO)*, pages 1–6, 2019.
- [100] Stanislav Alexovič, Milan Lacko, and Ján Bačík. 3d mapping with a drone equipped with a depth camera in indoor environment. *Acta Electrotechnica et Informatica*, 23(1):18–24, 2023.
- [101] Stephen M. Nogar. Autonomous landing of a uav on a moving ground vehicle in a gps denied environment. In *2020 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, pages 77–83, 2020.
- [102] Václav Pritzl, Matouš Vrba, Petr Štěpán, and Martin Saska. Fusion of visual-inertial odometry with lidar relative localization for cooperative guidance of a micro-scale aerial vehicle, 2026.
- [103] Jorge Peña Queralta, Li Qingqing, Fabrizio Schiano, and Tomi Westerlund. Vio-uwbbased collaborative localization and dense scene reconstruction within heterogeneous multi-robot systems. *CoRR*, abs/2011.00830, 2020.
- [104] Igor Spasojevic, Xu Liu, Alejandro Ribeiro, George J. Pappas, and Vijay Kumar. Active collaborative localization in heterogeneous robot teams, 2023.
- [105] Yu Xianjia, Yulin Zhang, Wei Li, and Jian Wang. Cooperative uwb-based relative localization for uav and ugv systems in gnss-denied environments. *Sensors*, 21(7):2415, 2021.
- [106] Shule Wang, Jinsong Wang, and Jiajun Chen. Ultra-wideband-based localization for multi-uav systems and collaborative heterogeneous multi-robot systems. *Procedia Computer Science*, 175:298–305, 2020.
- [107] Paola Torrico Morón, Francesco Schiano, and Antonio Franchi. Towards large-scale relative localization in multi-robot systems with dynamic uwb role allocation. *IEEE Robotics and Automation Letters*, 7(2):3118–3125, 2022.
- [108] Jorge Peña Queralta, Tuan Nguyen Gia, Hannu Tenhunen, and Tomi Westerlund. Uwb-based system for uav localization in gnss-denied environments: Characterization and dataset. *IEEE Access*, 8:179931–179947, 2020.
- [109] Thien-Minh Nguyen, Minh Cao, and Lihua Xie. Robust target-relative localization with ultra-wideband ranging and communication. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 2312–2319. IEEE, 2018.