Collaborative Localisation of Autonomous Vehicles

Developing a Minimum-Energy Filter and Experimental Testbed

Jack Henderson

2023

Collaborative Localisation of Autonomous Vehicles

Developing a Minimum-Energy Filter and Experimental Testbed

Author:
Jack HENDERSON

Supervisory Panel:
Prof. Jochen TRUMPF
Dr. Mohammad ZAMANI
Prof. Robert MAHONY

A thesis submitted for the degree of Doctor of Philosophy of The Australian National University

03/11/2023

COLLEGE OF ENGINEERING, COMPUTING & CYBERNETICS

AUSTRALIAN NATIONAL UNIVERSITY

Declaration of Authorship

I, Jack HENDERSON, declare that the work presented in this thesis is my own original work. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- No part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution.
- Where I have referenced or quoted the published work of others, this is always clearly attributed.

The following is a list of publications I have produced in collaboration with colleagues whilst undertaking this research. Much of the work from these publications is incorporated into this thesis.

- J. Henderson, J. Trumpf and M. Zamani, “A Minimum Energy Filter for Distributed Multirobot Localisation,” in *IFAC-PapersOnLine*, vol. 53, 2020, pp. 4916–4922. DOI: 10.1016/j.ifacol.2020.12.1068
- J. Henderson, M. Zamani, R. Mahony and J. Trumpf, “A Minimum Energy Filter for Localisation of an Unmanned Aerial Vehicle,” in *2020 59th IEEE Conference on Decision and Control (CDC)*, Jeju Island, Korea (South): IEEE, Dec. 2020, pp. 4188–4193. DOI: 10.1109/CDC42340.2020.9303730
- J. Henderson, M. Zamani, R. Mahony and J. Trumpf, *Inertial Collaborative Localisation for Autonomous Vehicles using a Minimum Energy Filter*, Apr. 2021. arXiv: 2104.05897 (Preprint)
- Z. Wang, Y. Ng, J. Henderson and R. Mahony, “Smart Visual Beacons with Asynchronous Optical Communications using Event Cameras,” in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct. 2022, pp. 3793–3799. DOI: 10.1109/IROS47612.2022.9982016

Signed: 

Date: 03/11/2023

Acknowledgements

FIGURE 1: The team celebrates the successful completion of the experimental trials. From left to right; Alex Miles, Behzad Zamani, Riley Lodge, Phil Smith, Jack Henderson, Andrew Tridgell, Alex Martin, Madeleine Cochrane, Wanqi (Peter) Yao

While a PhD is never easy nor without its unique challenges, the people supporting you can have a big impact in helping you to navigate the trials and tribulations of the PhD journey. I will be forever grateful for the supervision, guidance and mentorship of my primary supervisor Jochen Trumpf. You have provided me with an amazing opportunity and have been unwavering in your support throughout the entire process. You have taught me not only a wealth of technical knowledge but also the tricks of navigating the academic bureaucracy.

I am also extremely grateful for the support of my secondary supervisor Behzad (Mohammad) Zamani. Your guidance and encouragement throughout the entire journey have been invaluable and have given me the motivation to persevere through many tough challenges.

I also wish to thank my secondary supervisor Rob Mahony and the entire Systems Theory and Robotics group. Thank you, Rob, for welcoming me into the group and providing key nuggets of wisdom at precisely the right moment. I have thoroughly enjoyed working with the STR group, and they have been a great source of knowledge, support, and friendship. While the membership of the group is often in flux, I would particularly like to mention Pieter van Goor, Ziwei Wang, Shawn Ge, Yonhon Ng, Zheyu Zhuang, Benjamin Hofmann, and Alexander Ollman.

Without the technical support of Alex Martin, I would not have been able to conduct such a complex set of experiments. Thank you for the many hours of technical assistance and design work you put into this project, and for the many times you repaired the UAV after my many unsuccessful experiments.

Special thanks also go to the team of volunteers, pictured in Figure 1, who helped me collect the experimental data that is used in this thesis. Thank you all for your hard work, patience and time over two long days. I would also like to thank Ron Miller and the Gundaroo Park Land Management Board for allowing us exclusive use of Gundaroo Park for our experiments.

Finally, I would not have made it to this point without the support of my friends, family, and partner, Alex. As the first in my family to go through the PhD process, it has been a learning experience for us all, but I have received nothing but support, love and encouragement from all of you.

This research was supported by the Commonwealth of Australia as represented by the Defence Science and Technology Group of the Department of Defence, and an Australian Government Research Training Program (RTP) Scholarship.

Abstract

Multi-robot autonomous systems are rapidly gaining traction across many industries and are at the cusp of widespread mainstream adoption. Compared with a single-robot, coordinated multi-robot operations unlock new capabilities and enhance system robustness. In particular, the task of robot localisation can be transformed by the use of multi-robot systems. Collaboration among robots facilitates improved localisation accuracy, resilience to sensor degradation, and navigation in previously inaccessible environments. However, this collaboration introduces increased complexity, communication overhead, and computational demands. Efficiently managing data dependencies arising from inter-robot information exchange stands as a fundamental challenge in collaborative localisation algorithm design.

This research contributes to the advancement of collaborative localisation of multi-robot systems through the development of innovative algorithms designed for modern robotics platforms and the construction of a unique experimental dataset to validate such algorithms.

The thesis begins with an in-depth study of existing approaches to collaborative localisation and a scrutiny of the mathematical structure that allows inter-robot dependencies to be accounted for with only minimal network communication. Despite considerable advancements in state estimation algorithms over the past three decades, the collaborative localisation literature primarily leans towards Kalman filter-based methods, which are ill-suited for modern aerial robots. To address this limitation, we introduce a novel algorithm for single-robot velocity-aided inertial localisation which utilises the concept of minimum-energy filtering. Extending this filter to the multi-robot case, we leverage existing techniques that reduce communication overhead while maintaining track of inter-robot dependencies. Simulations of these algorithms highlight the significant improvements they offer over non-collaborative localisation, while also revealing trade-offs between the level of communication between robots and localisation accuracy.

To validate the real-world performance of the proposed algorithms, we recognise the need for real-world data. However, no publicly available dataset currently fulfils the requirements for testing inertial collaborative localisation algorithms. In the latter part of this thesis, we address this gap by designing a heterogeneous robot fleet of both aerial and ground-based vehicles. Using the fleet, we conduct outdoor experiments and produce five collaborative localisation datasets that validate the proposed algorithms and may, in future, serve as a common benchmark dataset within the research community.

Notation Reference

Symbol	Description	Reference
$\dot{x}$	Time derivative of x	
$A \succ 0$	A is positive definite	
$A \succ B$	$(A - B)$ is positive definite	
$A \succeq 0$	A is positive semi-definite	
$(.)_\times$	Cross Product operator	A.3.1
$(.)^\wedge$	Wedge operator	A.3.1
$(.)^\vee$	Vee operator	A.3.1
$(.)^\top$	Matrix transpose	
$(.)^{T_X G}$	Exponential functor	A.2.1
$\|.\|_A$	Weighted 2-norm of a vector	A.1.4
$[\,.\,,\,.\,]$	Lie bracket operator	
$\langle\,.\,,\,.\,\rangle$	Inner product	
ad_Γ	Adjoint action of Γ on $\mathfrak{g}$	
blkdiag	Block-Diagonal Matrix constructor	A.1.2
$\mathbf{d}f(X) \circ X\Gamma$	Directional derivative of f, evaluated at X, in direction $X\Gamma$	
det	Matrix determinant	
exp	Exponential map	A.3.3
Exp	Vector form of exponential map	A.3.3
G	A Lie Group	
$\mathfrak{g}$	The Lie algebra of G	
$\mathfrak{g}^*$	Dual of the Lie algebra $\mathfrak{g}$	
Hess	Hessian operator	
$\boldsymbol{I}_n$	Identity matrix of size $n \times n$	
$\mathbb{P}_s$	Symmetric projection	A.1.1

Symbol	Description	Reference
$\mathbb{R}$	Real numbers	
$\mathrm{SE}(n)$	Special Euclidean Group	A.3
$\mathfrak{se}(n)$	Lie algebra of $\mathrm{SE}(n)$	A.3
$\mathrm{SE}_2(3)$	Extended Special Euclidean Group	A.3
$\mathfrak{se}_2(3)$	Lie algebra of $\mathrm{SE}_2(3)$	A.3
$\mathrm{SO}(n)$	Special Orthogonal Group	A.3
$\mathfrak{so}(n)$	Lie algebra of $SO(n)$	A.3
T	Torsion Tensor on $\mathfrak{g}$	A.2.2
tr	Matrix trace	
$T_e L_X \circ \Gamma$	Tangent map at e of the left translation of Γ, denoted by the shorthand $X\Gamma$	
Λ	Connection function on $\mathfrak{g}$	A.2.2
Γ, Ψ	Arbitrary elements of the Lie group $\mathfrak{g}$	

Contents

Chapter 1

Introduction

The motivations for this thesis arise by examining the intersection of two current trends in autonomous vehicle technology; the emergence of interacting networks of autonomous vehicles, and the ongoing push for better localisation performance in increasingly challenging conditions. To provide some context around these trends, we first give an overview of existing applications for autonomous vehicles and then explore how networked autonomous systems present a range of new opportunities. We then show how the localisation task fits in as a component of an autonomous vehicle and the importance of accurate positioning for mission success. Connecting these two ideas we introduce the main focus of the thesis, collaborative localisation, and postulate on the potential advantages of such schemes. From there, we provide an overview of the remainder of the thesis and the primary contributions.

1.1 Autonomous Vehicles in the Modern Day

Autonomous vehicles have matured from experimental toys to practical, useful tools which have become increasingly invaluable across a range of industries. They amplify productivity, reduce health and safety risks, minimise downtime and increase quality. Autonomous vehicles are already being deployed in a variety of environments with a range of configurations and capabilities. For example, the Australian mining company, BHP, has commenced transitioning its fleet of mining trucks to be fully autonomous [5], increasing safety on the mine site and allowing for longer operating hours. In agriculture, farmers are using autonomous vehicles both at ground level and from the air in order to gain vital information about crop health, control weeds, and even muster livestock [6]. Zipline [7] is using autonomous aircraft to deliver vital medical supplies, such as vaccines and medication, to remote parts of African nations where existing transport infrastructure is limited. Examples from some of these applications can be seen in Figure 1.1.

Recent military conflicts have also shown the utility and prevalence of autonomous vehicles in the battlespace, primarily through the use of autonomous and remotely-piloted aircraft. These range from large-scale fixed-wing aircraft, such as the MQ-9 Reaper and MQ-4C Triton, to mid-size aircraft, including the Bayraktar TB2, to small-scale quad rotors, seen in Figure 1.2. Evidence

FIGURE 1.1: Autonomous vehicles are in use across a range of industries and civilian domains. [8]–[10]

FIGURE 1.2: From the large-scale to the micro-scale, militaries are using autonomous systems in a range of different roles. [12], [13]

from the Nagorno-Karabakh War [11] and more recently, the Russia-Ukrainian conflict has demonstrated how effective these mid and small-size aircraft can be, especially due to their low cost, small size, ease of deployment and simplicity of use. Soldiers have also been seen using commercial off-the-shelf aircraft such as the DJI Mavic for surveillance and reconnaissance as well as modifying them to add small weapons payloads.

1.2 Towards the Future of Autonomous Vehicles

Autonomous vehicles have already demonstrated their utility and value across a wide range of civilian and military applications. As advancements in sensor technology, computing power, and intelligence continue, the capabilities of these platforms will grow, while their size and cost decrease. However, in the vast majority of use cases that we see today, these autonomous systems are composed of a single vehicle, with either one or many human operators supervising the process. While in some cases there may be multiple autonomous vehicles operating in one environment, the interactions between these systems are limited and are mostly focused towards detect-and-avoid operations. The use of a network of many autonomous vehicles, operating in collaboration with each other opens up a vast array of new capabilities while improving performance in many existing single-vehicle applications. For example, in a surveillance application, a networked autonomous system can provide different perspectives from different locations, can cover a given

FIGURE 1.3: A record-breaking Intel Shooting Star demonstration with 1,218 unmanned aerial vehicles. [18]

area faster, provides robustness to failure, and enables a heterogeneous mix of sensor and effector suites to be used.

An example of a new capability enabled by networked autonomous systems is the establishment of a long-range and dynamically reconfigurable communications mesh network over an area with poor local infrastructure. Vehicles can collectively monitor the status of the network and optimise performance by determining where to move each vehicle. Adapting to failures and changing circumstances to increase robustness and resilience is a key element of why networked autonomous systems will become invaluable.

While the benefits of networked autonomous systems are clear, the maturity of the technology is relatively low. Momentum, however, is building — as evident by the rise in such applications as drone light shows. In 2015, Ars Electronica and the Intel Corporation set a world record for 'Most Unmanned Aerial Vehicles Airborne Simultaneously' at 100 [14]. Three years later, at the 2018 Winter Olympics opening ceremony, a new record of 1,218 UAVs was set [15], pictured in Figure 1.3. As of 2023, the current record stands at 5,164 [16], demonstrating the rapid pace of advancement in the sector.

Though drone light shows may seem like just an interesting novelty, the development and operation of such systems serve as a proof of concept for large-scale networked autonomous systems and reveal the unique challenges and pitfalls not previously encountered in single-vehicle or small-scale multi-robot systems. Yan, Jouandeau and Cherif [17] provides a comprehensive overview of the advantages of multi-robot systems, the challenges that arise in multi-robot environments, and how the literature is working to address these challenges.

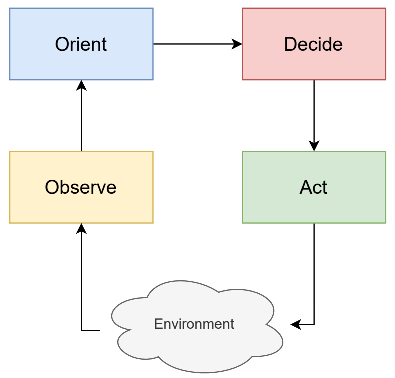


FIGURE 1.4: A simplified model of Boyd's OODA Loop

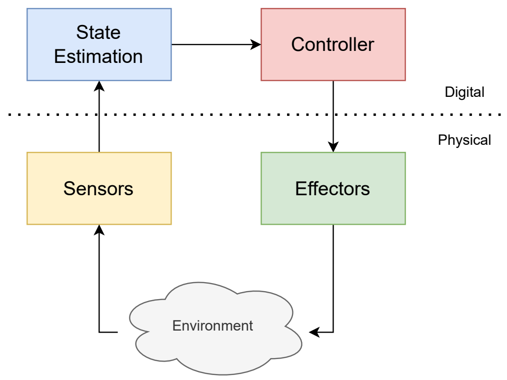


FIGURE 1.5: An autonomous system's feedback control loop

1.3 The Role of Localisation in Autonomous Vehicles

In the 1960s, fighter pilot and renowned military strategist John Boyd created the OODA loop to describe how pilots operate in combat [19]. Shown in Figure 1.4, the OODA loop has four elements; Observe, Orient, Decide, and Act. An effective pilot first *observes* the environment in which they are situated. These observations are used to *orient* the pilot and build a picture of the operational environment. The pilot then uses this information in conjunction with the goals and objectives to *decide* on the best course of action, and then *acts* in accordance with the plan. The pilot who can complete the loop faster and "get inside their opponent's OODA loop" has the upper hand in combat since they can make decisions and correct errors faster than their opponent. However, speed is not the only factor — the accuracy of each step is just as important. A pilot with a keen eye may notice something others miss, which in turn gives them a better understanding of their environment. One might have a better strategic understanding and thus make better-informed decisions, or be a more skilled pilot capable of executing a plan more effectively.

Since its inception, the applicability of the OODA loop has expanded to broader military strategy, business, and politics. The concept is also remarkably similar to the common feedback control system used in robotics, shown in Figure 1.5. In an autonomous system, the sensors provide observations of the environment, which are then fused to form an estimate of the state of the system or a model of the operational environment. This estimate is then fed into a control algorithm, which determines a set of control inputs that are then applied to the robot's effectors in order to interact with the environment.

The lessons from Boyd's model also apply to the feedback control loop. For high-performance applications the speed at which the loop executes is essential, but just as important is the accuracy of each step. High-quality sensors provide better sensor data for the state estimation process. With better data and advanced algorithms, the state estimation process can provide more accurate position information to the controller. In turn, the controller is better informed and can make better decisions, which can be precisely executed by quality effectors. On an autonomous vehicle, there will always be physical limitations, particularly in terms of size, mass and power, as well as external limitations such as cost, procurement and manufacturing. Naturally, trade-offs must be made to satisfy requirements, and this affects all aspects of the control loop. A key difference though is that the sensors and effectors exist in the physical domain, while the control and state estimator exist in the digital domain. Thus, there is a higher degree of freedom in controller and estimator design, and intelligent algorithms can offset deficiencies in hardware.

As alluded to in the previous section, state estimation is the process of converting raw sensor data into meaningful information about the state of the system and its surrounding environment. For autonomous vehicles, the primary system state that is of interest is the position of the vehicle within its environment. This information is essential for the controller in order to navigate to specific locations, avoid collisions and interact with other elements of the environment. Again, the better this information is, the easier it is for the controller to do its job, and the better the performance of the overall system will be. Two key elements that affect this quality are the incoming sensor data and how good the localisation algorithm is at transforming this data into accurate information.

1.4 The future of Localisation Technology

Given the growing interest in networked autonomous systems, a natural question that arises is whether one can take advantage of the network to improve the localisation performance of individuals within it. Depending on the information and sensors available, information collected by one robot could be shared with others to improve localisation performance. This idea, broadly known as collaborative localisation, has the potential to unlock a wide range of new and exciting applications for autonomous vehicles. Consider a scenario where a single autonomous vehicle using GNSS as its primary localisation source travels into an area of dense tree coverage where GNSS performance is degraded. Ordinarily, this could lead to severely reduced localisation performance and could affect the vehicle's ability to perform its mission. However, if the vehicle is part of a network, other vehicles in the area could share their localisation information and provide a reference for the lost vehicle.

GNSS-denied scenarios are just one example of how collaborative localisation can be used to improve existing methods. Another example is in aggregating sensor information from multiple vehicles. Consider another scenario in which there are several robots in a network, each one within the range of a different ground-based beacon. The sensor information each vehicle has alone is not sufficient to determine a location, but when the sensor data is shared between vehicles, each vehicle can accurately determine its position.

A third example of how collaborative localisation could be utilised is in a network of heterogeneous vehicles. Different vehicles in the network may have different capabilities and be equipped with different sensors, perhaps due to cost or weight restrictions. Sharing the information from these sensors with other vehicles in the network could enable better localisation performance on vehicles that are not as well-equipped. In turn, this has the potential to reduce the number and complexity of sensors required on each vehicle, reducing cost and freeing up space for other payloads.

While these examples show how collaborative localisation could open up a wide range of new applications, there are many challenges involved with implementing such a concept. This thesis aims to explore these complexities and works towards addressing a number of the key issues within the area of collaborative localisation.

1.5 Primary Contributions and Thesis Structure

The primary contributions of this thesis are summarised as follows;

- Development of a novel minimum-energy filter for localisation of a mobile robot using inertial and landmark measurements (Algorithm 1).
- Analysis of existing Kalman filter-based collaborative localisation algorithms and unification of concepts around decentralised filtering (Section 4.2).
- Derivation of a minimum energy filter for inertial collaborative localisation (Algorithm 9), and an extension of the filter for decentralised computation (Algorithm 10).
- Demonstration and analysis of the performance of single and multi-vehicle localisation algorithms using a purpose-built simulation test bench (Sections 5.2, 5.3, 5.4).
- Design and construction of a fleet of heterogeneous autonomous vehicles with a novel sensor package (Section 6.3).
- Compilation of an experimental dataset for testing and validation of collaborative localisation algorithms (Section 7.4).

Expanding on this list, we now provide a detailed overview of the remainder of the thesis.

In Chapter 2, we provide a comprehensive survey of the existing literature surrounding localisation. We explore the tools, techniques and algorithms used for localisation, both in the context of single-vehicle and multi-vehicle scenarios. Additionally, we look at existing datasets and experimental work used to validate localisation algorithm performance and examine the different kinds of sensor modalities used in localisation. Based on this analysis, we highlight two areas of interest that are poorly addressed in the existing literature. One is the area of high-performance collaborative localisation algorithms, and the other is high-quality multi-robot datasets on which to test these algorithms. These two topics motivate the primary contributions of this thesis.

In Chapter 3, we establish the foundations for our subsequent work on collaborative localisation by deriving a minimum energy filter for the localisation of a single vehicle. We aim to create a realistic filter capable of being implemented on a physical vehicle, and so we pose the problem as a 3-dimensional pose and velocity estimation problem using an inertial measurement unit, complete with biases and the effects of gravity. We identify the minimum energy framework as a robust and high-performance design able to deal with the complex geometry of the state space. Building on the abstract formulation of the minimum energy filter, we specialise the result to the given localisation problem and present a novel discrete-time pose estimation algorithm for mobile robots.

In Chapter 4, we bring together several complex ideas around collaborative localisation. We begin by providing a detailed breakdown of the structure of existing distributed localisation algorithms which are all based on the extended Kalman filter (EKF). This provides a unified way to analyse these different algorithms, determines the exact conditions under which they can be constructed, and also describes unambiguously what information is transmitted between agents and when. Independently of this, we build on the single-vehicle filter derived in the previous chapter to construct a centralised multi-vehicle inertial collaborative localisation algorithm using the minimum energy framework. Then, using the knowledge from our analysis, we show how the centralised minimum energy filter equations can be decoupled and distributed across the network in the same way as the existing EKF equations can be decoupled and distributed. We go on to show how other techniques, previously only applied to the EKF, can equivalently be applied to the minimum energy filter which enables a reduction in communication between agents.

In Chapter 5, we then validate the performance of the proposed filters in a simulation. Due to the lack of a suitable dataset, we must construct a synthetic dataset with simulated inter-vehicle and landmark measurements. The baseline simulation results demonstrate that the minimum energy collaborative localisation filter is capable of accurately localising each robot in the network and maintaining stability throughout the duration of the simulation. We go on to compare the different variations of the minimum energy collaborative localisation algorithms that we propose in Chapter 4. The results demonstrate the effectiveness of collaborative methods over non-collaborative ones and, among the collaborative methods, they show a clear trade-off between the level of communication between robots and the accuracy of localisation. Further simulations demonstrate two different scenarios of degraded sensor performance in which the collaborative algorithms provide a clear benefit over the non-collaborative filter.

Motivated by the need for a new collaborative localisation dataset, in Chapter 6, we detail the process of designing and building a fleet of uncrewed ground and aerial vehicles. The vehicles are designed to be capable of operating in a multi-vehicle environment and are equipped with sensors to make relative measurements between vehicles. The sensor package of each vehicle contains a mix of off-the-shelf components, as well as bespoke hardware and software. We focus particularly on the development of the novel camera system which measures relative bearings to other vehicles and landmark beacons, as well as the integration of the ultra-wideband sensor system which measures relative distances between nodes.

In Chapter 7, we then use the new fleet of vehicles developed to conduct experiments and record sensor data. We process the raw data, perform calibration and construct a dataset that contains 5 separate mission sequences. Each mission contains sensor data for 3 aerial vehicles and 3 ground vehicles operating in proximity in an open outdoor environment. Drawing together the different elements of this thesis, we use this new dataset to evaluate the localisation performance of the different collaborative localisation algorithms that were developed in Chapters 3 and 4. The results of this evaluation demonstrate both the quality and utility of the newly created dataset, as well as the ability of the proposed algorithms to work in a real-world environment.

Finally, in Chapter 8, we summarise the findings of the thesis, highlight the key contributions and provide an outlook on future extensions to this research.

Chapter 2

Literature Review

Interest in the field of mobile robotics accelerated rapidly in the 1990s, as the miniaturisation of sensors, improvements in computing power, and cost reductions began to enable an increasing range of potential applications. In those early days, mobile robots were mostly wheeled ground-based platforms constrained to simple environments such as inside buildings or smooth roads. More recently, flying robots and quadcopters in particular have become tremendously popular across a range of commercial, industrial, military and recreational activities. An almost universal problem for all mobile robots, whether they be aerial or ground-based, is localisation. In order for a robot to independently navigate through an environment, interact with objects, and achieve mission objectives, it must have a sense of its place within that environment.

Robot Localisation is the task of using a mobile robot's sensors to determine the position (and in many cases the orientation) of the robot with respect to some specified frame of reference. Within this field, there is a wide array of different strategies, techniques and algorithms dependent on the specific application, sensors available and environment that the robot is operating within. In this literature review, we will explore the benefits and drawbacks of these approaches, with a particular focus on filtering techniques. From there, we will explore the area of multi-robot localisation and how this problem differs from the single-robot case. Finally, we will analyse how localisation algorithms are tested and evaluated using real-world experimental data.

2.1 Approaches to Localisation

How localisation can be performed is highly dependent on the kind of sensors equipped on the robot, and what information the robot has about its environment. Two common and simple methods of localisation are trilateration [20], [21] and triangulation [22], [23]. If a robot is capable of measuring the distance (trilateration) or bearing (triangulation) to three or more known points, then the position of the robot can be geometrically determined. More than three measurements can be used to reduce the error caused by imperfect sensor measurements. Many kinds of localisation systems utilise these concepts, including radio-navigation aids such as Non-Directional Beacons (NDB) and

VHF Omnidirectional Range (VOR) stations commonly used in the aviation industry [24]. Also included in this category are optical motion capture systems, such as VICON [25] and OptiTrack [26], which use bearing measurements from multiple fixed cameras to identify and localise objects. However, the limiting aspect of these approaches is that they only provide location information at a single point in time and at a single location. If a robot is not stationary, it must recalculate its position based on new sensor measurements, as there is no way to relate a previous measurement to a current one. Additionally, if the sensor data does not contain enough information, for example only two range measurements, there may be multiple ambiguous solutions of the position.

2.1.1 Dead Reckoning

A different approach to the localisation problem is the process of dead reckoning. Sensors that measure the robot's movement can be used to calculate the position of the robot relative to its starting point. For example, on wheeled robots, it is common to have a rotary encoder on the wheels. Assuming that the robot does not slip or skid, then measurements of the wheel's rotation can be input into a mathematical model to calculate how far the robot has moved. This process is called odometry, and an early implementation for simple wheeled robots can be seen in Crowley [27]. Similarly, inertial navigation is the process of measuring acceleration and angular velocity and integrating the measurements using a kinematic model to calculate the robots position and orientation [28], [29]. The drawback of dead reckoning methods is that the calculated position drifts away from the true position over time, and the pose can only be calculated relative to an initial starting point. Each sensor measurement contains a small error that slowly accumulates over time and, if left uncorrected, can result in significant errors in the position.

Inertial navigation was commonly used in aircraft in the early jet age to navigate across long distances such as over the ocean where traditional radio-navigation aids were not available. These Inertial Navigation Systems (INS) were bulky, expensive, highly complex, and resulted in a drift in position of approximately 4 km per hour [30]. Even modern-day systems with ring-laser gyroscopes can only achieve drift rates on the order of 1 km per hour [30]. The advent of the MEMS (Micro Electromechanical Systems) IMU has enabled small-scale robotic aircraft to take flight. However, these miniature IMUs have significantly larger measurement errors making drift orders of magnitude higher than high-grade aircraft INS. Note that odometry and inertial techniques only provide the robot with a position relative to its starting point, and it must know where it started in order to localise within the environment.

An example of the dangers of relying solely on inertial navigation is the tragic incident of Korean Airlines flight KAL007 [31]. In 1983, a Boeing 747 departed Anchorage, Alaska en route to Seoul, South Korea. Due to the lack of ground-based radio navigation aids along most of the route, navigation of the aircraft was primarily conducted using the aircraft's triple-redundant inertial navigation system. As the flight progressed, the aircraft slowly drifted

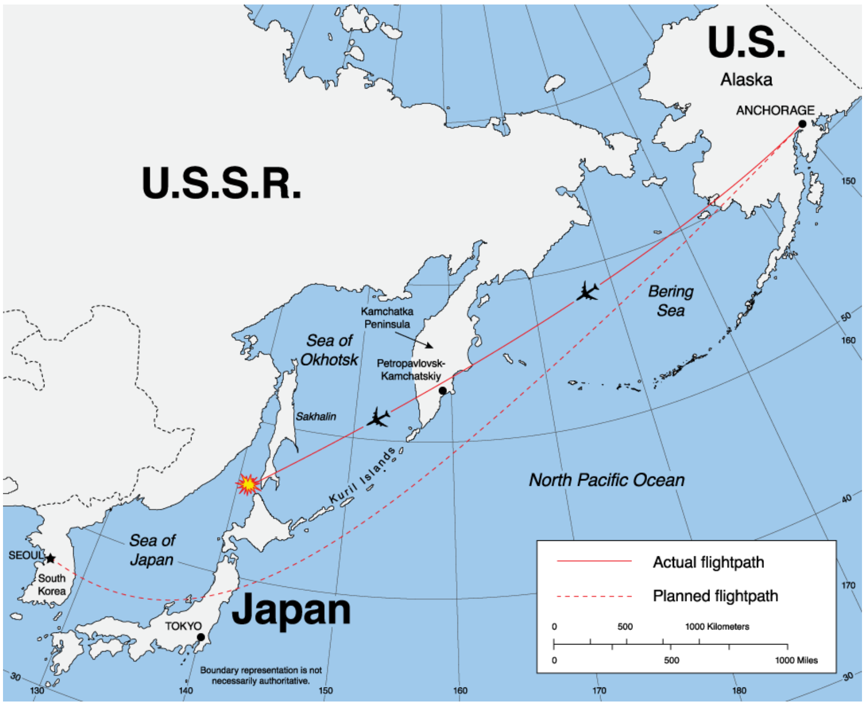


FIGURE 2.1: Map showing the flight path of KAL007 compared to the planned route. [33]

off-course from its assigned route and into the airspace of what was then the Soviet Union, shown in Figure 2.1. The pilots were seemingly unaware of their navigation error, and with no ground-based beacons to validate their position, they did not detect their incursion into Soviet airspace. The aircraft was intercepted by Soviet fighter jets and shot down over the Sea of Japan, killing all 269 people on board. While the exact cause of the navigation error was likely due to pilot error, and an alert crew should have detected their deviation, the incident still serves as a warning of the real dangers that result from localisation and navigation errors. The incident also served as a major contributor to the US Government allowing civilian access to the Global Position System (GPS) network of satellites, which at that time, was only used for military purposes [32].

2.1.2 Satellite Navigation

The advent of satellite navigation technology, beginning with GPS, revolutionised outdoor navigation. At present, several Global Navigation Satellite Systems (GNSS) are in operation, including GPS, GLONASS, BeiDou, and Gallileo, however, they all operate under the same principle, which is based on trilateration. A network of orbiting satellites make continual radio broadcasts of their position and current time which are received by the GNSS receiver. The receiver uses the information from a minimum of four satellites to determine the time of flight (distance) to each satellite and thus can trilaterate the position of the receiver [34]. The use of additional satellites, and even multiple GNSS constellations can improve positioning accuracy down to the order of 1

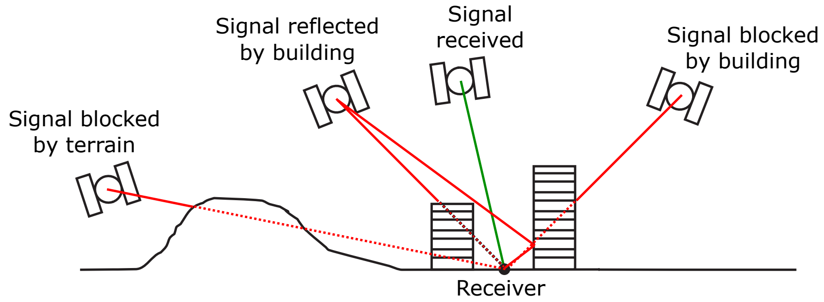


FIGURE 2.2: Illustration of different environmental factors affecting satellite-navigation performance. Adapted from [34].

metre. Further use of ground-based (D-GPS, RTK) or satellite-based (SBAS) correction data can increase accuracy down to the order of 1 to 10 centimetres.

While satellite navigation promises to provide sub-metre accurate positioning information anywhere in the world with relatively cheap hardware, satellite navigation can become unusable in many environmental conditions. Due to their long range and wide coverage, the received signal from GNSS satellites is very weak, on the order of -160 dBW, compared to the environmental noise floor of -140 dBW [34]. This makes the GNSS signal highly sensitive to attenuation and interference from the environment. Buildings, tree cover, terrain obstructions and many other factors can degrade satellite navigation performance, as well as malicious actors intentionally jamming the signal. Some examples of this are illustrated in Figure 2.2. Due to these limitations, one must be careful not to rely solely on satellite navigation for localisation. Further, there are environments where satellite navigation is entirely unusable, including indoors, underground, and undersea.

2.2 Sensor Fusion and Filtering

In addition to the sensor types discussed above, an almost countless number of other sensors have been used in mobile robotics, including LIDAR, vision, SONAR, ultrasound, and barometers. Each sensor provides the robot with different information about the robot and its environment with varying levels of accuracy, quality, frequency and reliability. Rather than just using a single sensor to perform localisation, the fusion of multiple different sources of information can offset the limitations of a single sensor and provide better localisation accuracy [35].

One of the key elements of robot localisation is that it is not a one-off process of determining a robot's location at a single point in time. Rather, it is a process of continually estimating the location of the robot as it moves through the environment. Thus, in order to perform sensor fusion with data from different times and different locations, we must incorporate a motion model of how the robot moves through the environment in order to provide a relationship between disparate sensor measurements. This allows localisation to be performed using the entire history of measurement data, not just the data sensed at the current time. This leads not only to more accurate localisation performance but also can enable localisation in cases where the information at a single point in time is not sufficient [36].

The process of estimating the state of a system over time based on a model of the system and a sequence of measurements is called filtering [37], [38]. Several properties make a filter distinct from other similar types of algorithms such as estimators, smoothers, and optimisers, however, it is rare to find an example in the literature where these properties are well articulated. These properties are;

- System Model. As discussed above, a filter must incorporate a model of how the system evolves over time in order to relate measurements from different times. An implicit stationary model may be present in many algorithms, which assumes the system state is static.
- Non-Anticipating. Data from the 'future' is not required to calculate the estimate at a given point in time. This is in contrast to global optimisation methods or 'smoothers', which use 'future' data to increase the estimate accuracy.
- Recursive. The computation and memory requirements do not grow unbounded over time. This is essential for long-running missions on limited hardware.
- Online. An estimate of the current position of the robot is provided in 'real time'. This is an essential property if the estimate is to be used as part of the robot's control system.
- Measure of estimate quality. In contrast to observers, which provide only an estimate, a filter provides a quantification of the quality of the estimate, for example as error bounds or statistics on the error. This is a valuable tool when interpreting the estimate for use in other parts of the system.

It is also a common misconception that filters only apply to stochastic processes and thus must be treated in a probabilistic context. This is not the case, as the introduction of stochastic processes is a modelling choice and there are many examples of deterministic filtering algorithms which we will explore in subsequent sections, including least-squares filters [39] and complementary filters [40]. More precise definitions of a filter and related terminology are developed in Appendix B.

In the literature on filters for robot localisation, three common robot localisation problems drive filter design, namely simultaneous localisation and mapping, global localisation, and tracking. Each of these problems assumes different information is available to the robot and has different output requirements. In turn, this informs and constrains the design of filter algorithms to suit the specific set of requirements for each problem. We will explore these problems and filtering approaches in the subsequent sections.

2.2.1 Simultaneous Localisation and Mapping

If a robot is given no information about its environment, then it must build an understanding of the environment as well as localise itself within — a task known as Simultaneous Localisation and Mapping (SLAM) [41], [42]. SLAM is commonly performed where the goal of the task is to build a map of an unfamiliar environment. Sensors such as cameras and laser scanners are commonly used as they provide rich, dense information to be able to build detailed maps. However, without a means of global localisation, SLAM can only provide pose or position information relative to the starting point or local features. SLAM also suffers from the same drift problem as inertial navigation and must utilise techniques such as loop closure and bundle adjustment to correct for these effects and produce an internally consistent map. While there are online SLAM methods [43], many SLAM algorithms involve the use of global optimisation methods, often run offline, which optimise the whole trajectory of the robot taking into consideration all the measurement data collected.

2.2.2 Global Localisation

In cases where the environment is known, but the initial position of the robot is unknown, the task of localising the robot is called global localisation. Probabilistic methods are a common approach to the global localisation problem. The position of the robot is represented as a probability distribution over the environment, and sensor information is used to update this distribution and reduce the uncertainty in the location of the robot. One of the major challenges of probabilistic methods for global localisation is in how to represent the probability distribution, as symmetries in the environment and non-linear sensor measurements can often lead to multi-modal distributions that can not be easily parameterised.

Markov localisation [44] is a technique in which the environment is divided into a set of discrete set of locations, such as a grid of cells. The robot's location can then be represented as a discrete distribution over this set and Bayes' rule can be used to incorporate sensor data. This technique works well for environments that are naturally discrete, such as identifying which room a robot is in, or when the size of the environment is small, such as localising within a room. However, it becomes computationally intractable for larger environments. In addition, the precision of localisation is limited by the resolution of the discretization.

Monte-Carlo Localisation (MCL), an application of the method of Particle filtering (PF), is another class of probabilistic localisation methods [45]. Particle filters approximate probability distributions with a finite set of samples, known as particles. Sensor measurements are incorporated by weighting each particle according to Bayes' rule. The weighted density of particles then provides an approximation of the probability distribution of the robot's position. Theoretically, with enough particles, the particle filter is capable of accurately representing any arbitrary probability distribution. However, as the number of particles increases, so do the memory and computational requirements. Thus, particle filters must make a trade-off between accuracy and computational performance.

2.2.3 Tracking

Tracking is the problem of localisation within a known environment and a known (or estimated) starting position. While most global localisation methods can be used for tracking, many additional algorithms have been developed to take advantage of the initial information. The Kalman filter [46], [47] is perhaps the most widely known example of a tracking filter, which takes a stochastic approach to find the minimum variance estimate of the system state, conditioned on the measurement history. For a linear system where the sensor noise is white (uncorrelated over time and zero-mean) and has a known covariance, the Kalman filter is provably optimal in the minimum-variance sense. How the Kalman filter is formulated also means it is recursive — the state estimate depends only on the previous estimate and the new measurement data, not the entire history of measurements.

The tracking problem can also be considered in a deterministic framework. For example, Mortensen [39] and subsequently Willems [48], describe the tracking problem as a deterministic least-squares minimisation problem. Hijab [49] uses the term 'minimum energy' estimation to describe this approach to filtering, which we will adopt in this thesis. In Appendix C, we provide a comprehensive introduction to minimum energy filtering, including the reasoning behind the choice of terminology and a worked example for a linear system. As shown in the appendix, for linear systems, one observes that the stochastic Kalman filter provides exactly the same estimate as the minimum energy filter.

The Kalman filter was a significant breakthrough in the field of control, however, it only provides a solution for linear systems, limiting its use and applicability. Since its development in the early 1960s, a significant research effort has been devoted to developing filters for non-linear systems. We will explore these developments in the subsequent section.

2.3 Filtering for Non-Linear Systems

The Kalman filter was developed at the same time as the USA was entering the space race. NASA saw an opportunity to use a Kalman filter to estimate the orbital trajectory of the space capsule of the Apollo missions [50], [51]. However, the orbital and attitude dynamics of the space capsule's trajectory are non-linear and the Kalman filter is only suitable for linear systems. Schmidt, along with other researchers at NASA, proposed to linearise the system model around the current state estimate [52], [53], developing what is now known as the extended Kalman filter (EKF). The EKF provides a simple way to perform state estimation for arbitrary non-linear systems, but in doing so, it loses many of the properties of the original Kalman Filter, namely the optimality of the state estimate, and the consistency of the covariance matrix.

The problem of filtering on non-linear systems presents a much more complex and difficult challenge than the linear case. At around the same time, Mortensen [39] was also exploring approaches to non-linear state estimation. He highlights three key problems in the literature of the time.

> "First, it is difficult to justify the approximations made in deriving these equations. Second, there is no way of making a theoretical comparison of the various possible approximations; hence it is impossible to say, if one of the approximate filters were to be implemented, how much the resulting estimate deviates from the true conditional mean. Third, even the approximate nonlinear filtering equations are extremely cumbersome". [39]

Mortensen proposed a filtering approach, based not on statistics and minimum variance estimators, but on a deterministic least-squares minimisation of the error. As discussed above, this least-squares approach provides the same result as the Kalman filter on linear systems. However, for arbitrary non-linear systems, the exact least-squares solution "suffers from the same kind of moment problem or closure problem as does minimum-variance nonlinear filtering" [39]. Thus, a choice is made to approximate the solution to one that can be solved, however, this results in a different solution to that obtained by the EKF. Mortensen's work gained little momentum in the community, perhaps due to the perceived issues and similarities with the EKF and other statistical filters.

Other deterministic filtering approaches were also being explored, such as complementary filtering [54] which is based on the more historic frequency domain and transfer function approach to the problem. The complementary filter is attractive as it is a simple design and is efficient to compute. This was particularly important in the days when computers were orders of magnitude less powerful than they are today.

Another approach to extending the Kalman filter to non-linear systems was sigma-point filters. Rather than trying to directly model how the covariance transforms through the non-linear system model, sigma-point filters such as the Unscented Kalman Filter (UKF) [55], [56] compute the transformation by selecting a small representative set of samples from the distribution, applying the non-linear transform to each one individually, and then computing the covariance of the transformed samples. This method provides a computationally cheap alternative to linearising the measurement and system models, and in many cases results in a better approximation of the non-linear transform than the EKF does.

Despite its limitations, the extended Kalman filter emerged as the filter of choice for many applications, striking the balance between performance and simplicity, both mathematically and computationally. By simply linearising the system around the current state estimate, and then optimally filtering the linear system, the EKF is a filter that is simple to understand and implement. The trade-off is that the EKF suffers from robustness and stability issues and requires extensive parameter tuning in order to work effectively. Extensive analysis has been performed on the properties of the EKF and a wide array of parameter tweaks, filter resetting strategies, and other heuristics have been developed to skirt around some of these issues [57]. Despite these issues, the EKF is perhaps the most prevalent filter in the world and is very widely implemented to control an array of dynamic systems. In many use cases, the performance of the EKF is simply 'good enough' for the task at hand, likely in part due to non-complex dynamics and accurate sensor measurements.

2.3.1 Non-Linear Filters for Attitude and Pose

One application that is particularly challenging for non-linear filters is attitude estimation and, by extension, pose estimation. When used on the Apollo spacecraft, NASA found the performance of the EKF to be less than ideal, observing that "success or failure with this approach may be dependent on many factors, such as computer round-off errors, inadequate statistical models, and nonlinearities in the problem, any or all of which may trigger the filter's potential instability or inaccuracy" [50]. A comprehensive but slightly outdated survey of attitude estimation methods is presented by Crassidis, Markley and Cheng [58].

A significant factor discovered by Lefferts, Markley and Shuster [59] was the parameterisation of the attitude state in the filter. Geometrically, there are three degrees of freedom for an attitude state, and so the natural choice is to use a 3-parameter representation for attitude, such as Euler angles, or an angle-axis representation. However, these representations suffer from singularities (also known as gimbal lock) and complex non-linear behaviour. Higher-dimensional representations, such as quaternions (unit-length 4-vectors) or rotation matrices (3×3 direction cosine matrix (DCM)), do not have singularities, but instead are over-parameterised. An extended Kalman filter operating directly on quaternions or rotation matrices must re-project the state estimate back to the set of valid representations after each measurement update. Instead, Lefferts, Markley and Shuster [59] proposed a variation on the EKF, a 'reference state' is stored as a quaternion, and the filter operates on a 3-dimensional 'error state' representing the difference from the reference state. After each measurement update, the error state is reset to zero, and the reference state is updated by incorporating the error. This scheme, known as the Multiplicative Extended Kalman filter (MEKF) [60], combines the advantages of the quaternion representation for the state while using a 3-dimensional representation for the update. As the error state is reset to zero at every update, the state avoids the singularities of the 3-parameter representations.

The MEKF resulted in significantly improved performance over the standard EKF and has become the de facto standard choice for attitude filtering [61]. More recently, several extensions have been proposed to the MEKF, for example, Filipe, Kontitsis and Tsiotras [62] present a MEKF-based filter for pose estimation, while Martin and Salaün [63] derive a generalised version of the MEKF for estimating pose and velocity of an aircraft. The interested reader is referred to Solà [64], which gives a comprehensive description and derivation of the Error-State Extended Kalman Filter (ES-EKF), a MEKF-based filter for pose and velocity estimation with IMU measurements, including estimation of IMU biases and the gravity vector.

The improvement in the performance of the MEKF compared with the EKF makes it clear that the representation of attitude plays a key role. Underlying this is the concept that the rotations have a particular group structure, namely the Special Orthogonal Group $\mathrm{SO}(3)$. By understanding this group, filters can be designed to take advantage of the structural properties and geometry to provide improved performance over general non-linear filters that are not designed with these properties in mind.

The Invariant Extended Kalman Filter (IEKF) presented by Bonnabel, Martin and Salaün [65]–[67], provides a way to transform the error so that it is invariant to the estimated state of the filter, similarly to how the MEKF represents the state as the combination of a reference state and an error state. The IEKF is derived for a more general Lie Group structure, of which $\mathrm{SO}(3)$ is a member, and when applied to attitude estimation, it can be seen to be an extension of the MEKF with a slightly modified correction term. Further developments of the IEKF include the incorporation of sigma-point filtering, applications to SLAM, and analysis of stability and consistency [68]–[70]

On the deterministic front, several filter designs have been proposed both for attitude and pose estimation to similarly take advantage of the underlying group structure of $\mathrm{SO}(3)$ for attitude estimation, $\mathrm{SE}(3)$ for pose estimation, and Lie groups in general. A Lie group complementary filter for attitude estimation was developed by Mahony, Hamel and Pflimlin [40], [71] and also extended to pose estimation [72]. Due to their robust nature and simplicity of computation, these complementary filters were instrumental in enabling a new era of low-cost small unmanned aerial vehicles such as quadcopters which were able to utilise cheap MEMS IMUs and low-powered onboard compute modules to perform attitude estimation and stabilisation.

Additionally, the deterministic least-squares filtering concept, originally developed by Mortensen [39] has been extended to attitude and pose estimation [73]–[76]. These filters are also referred to as 'minimum energy' filters as the cost functional that is being minimised can be thought of as a measure of the energy of the unknown error signals. By minimising the total energy of the error signals, the filter produces an estimate of the state most compatible with the system model and the measurement model. Note that, as discussed by Mortensen, the exact minimum energy solution for arbitrary systems is an infinite dimensional optimisation problem, and thus the solution is approximated to second order. Saccon, Trumpf *et al.* [77] provide a general framework for deriving second-order optimal minimum energy filters on arbitrary Lie groups. However, the filter is presented in an abstract form and requires the derivation of a matrix form in order to be implementable on a computer. The structure of these minimum energy filters is similar to that of the IEKF, however, the minimum energy filter also corrects for the curvature of the Lie group.

2.4 Collaborative Localisation

As mobile robots became cheaper and improvements were made to wireless communication technology, a concept emerged of using multiple mobile robots, networked together, to perform a task. The use of a network of robots opens up a number of new opportunities but also presents many new challenges. Multiple robots provide a wider array of sensing capabilities due to the increased number of sensors, but also due to the spatial distribution of robots, which provides information from different perspectives. By sharing this information among the robots in the network, each robot gains a better picture of the environment than it could have created alone. However, the additional constraints imposed by network connectivity, increased number of failure points, and distributed computation all have the potential to create more problems than they solve. With careful system design and the right choice of algorithms, many of these issues can be mitigated, and networked robotics can become more robust and resilient to failure than single-robot systems.

Many similar terms have been used to describe the concept of multiple robots sharing information for localisation; collaborative localisation [78], collective localisation [79], cooperative localisation [80], cooperative navigation [81], and network localisation [82], just to name a few. In essence, these terms all relate to the same concept, and we will use the term collaborative localisation throughout this thesis when in the context of robotics applications, and the term networked filtering when speaking more generally.

Several distinguishing features separate collaborative localisation from other localisation and estimation problems:

- There is a group of robots that are operating in proximity to one another within a common environment,
- There is a means to share information between robots in the group *i.e.* a communications network,
- Robots are equipped with a variety of sensors that provide information about the robot and its environment,
- Some or all of the robots are equipped with sensors that can measure relative information about another robot, and
- Each robot, at a minimum, aims to estimate its own position.

There is a subtle distinction between the collaborative localisation problem and distributed or decentralised data fusion problem. In collaborative localisation, each robot is only tasked with estimating its own position, whereas in distributed data fusion, all robots are estimating a common, shared state.

Variations of the collaborative localisation problem may involve additional constraints to the core features described above such as:

- The number of robots in the group may not be known to all robots, or may change over time.
- The topology of the communication network may be unknown and may change over time.
- The bandwidth, quality, or latency of the network may be constrained and variable.
- Robots may be homogeneous, meaning that they all share the same attributes, or heterogeneous, meaning that they are different. This could include different movement capabilities (aerial and ground vehicles), different sensor payloads, or different computation resources.
- The sensors may be restricted to certain types, for example, bearing-only or range-only.
- Sensors may provide information about the robot's position with respect to the environment (absolute position) or only with respect to other robots (relative position).

The key common element that distinguishes collaborative localisation from other state estimation problems is that the sensors, computation, and storage are physically separated and are connected by a communication network. It is understanding and modelling this communication network that is key to understanding the collaborative localisation problem — how information flows between where it is created, where it is needed, and where it is stored. Part of this challenge is the task of accurately tracking the provenance of data within the network. If not tracked correctly, this can lead to a situation where information is mistakenly re-used and can reinforce erroneous beliefs and result in over-confidence in a robot's state estimate. This phenomenon is known as the double-counting problem, data incest, the 'whispering in the hall' problem, or the over-confidence problem and Julier and Uhlmann [83] provide several examples of how this can occur in networked estimation problems such as collaborative localisation.

Within the literature on collaborative localisation, there are two broad approaches, what we term the 'top-down' approach and the 'bottom-up' approach. Conveniently, this mirrors the terminology used by Crespi, Galstyan and Lerman [84] to describe different philosophies of multi-agent system design. Both of these methods attempt to deal with the double-counting problem in different ways, while still considering the constraints imposed by the communication network. Top-down methods are generally characterised by high levels of communication overhead but better performance. Approximations can be made to these methods to reduce the communication requirements at the expense of filter performance. On the other hand, bottom-up methods start with a foundation of independent robots with no communication. Information is shared between robots in order to improve estimates but suffers from diminishing returns as the number of messages increases. We will see that, while these methods approach the problem in different ways, they both inevitably converge to a middle ground of leveraging communication to share information when it significantly improves estimate quality and discarding information that has a negligible impact in order to save bandwidth.

2.4.1 The Bottom-Up Approach

In the bottom-up approach, a set of interacting filters are designed — one for each robot which tracks the state of that robot. When relative measurements are made between robots, the state estimates are communicated over the network and the information is incorporated carefully to ensure that double counting is avoided. This can be done in a number of ways with varying degrees of complexity. For example, Fox, Burgard *et al.* [78] present a particle filter for collaborative localisation in which each robot maintains an estimate of its own position. Upon detecting another robot, the estimated position of the other robot is shared, and this is used in conjunction with the relative position measurement to update the robot's own estimate. In order to avoid the double-counting problem, they propose that "once a robot has been sighted, [the filter] blocks the ability to see the same robot again until the detecting robot has travelled a pre-specified distance". [78] This essentially skirts around the problem by restricting the amount of measurement data that can be used and relying on the fact that other errors introduced into the system gradually reduce the correlation between state estimates of different robots.

A less restrictive approach is to explicitly track the provenance of each piece of data and record which robots have and have not incorporated the data into their estimates, a process often described as 'bookkeeping'. Bahr, Walter and Leonard [81] propose an EKF-based filter that stores additional information about each measurement and which measurements have been received by other robots. This approach demonstrates that it is possible to track this information, but the complexity and storage requirements become impractical for large networks of robots or prolonged operations. Similarly, Howard, Mataric and Sukhatme [85] proposes a particle-filter approach for relative localisation in which each robot maintains a full estimate of every other robot in the network. They address the double counting problem by creating a 'dependency tree', which tracks dependencies between estimates. Estimates are only updated using information from ancestors in the tree, ensuring that information does not propagate in loops, but with a penalty of reduced performance.

Rather than trying to exactly track the complex interdependencies between robots, an alternative is to use a method called covariance intersection (CI), proposed by Julier and Uhlmann [86]. Covariance intersection provides a way to safely fuse information regardless of how correlated or dependent the information is. When fusing two Gaussian-distributed random variables with unknown correlation, the CI method determines an upper bound on the covariance of the fusion result over all possible correlations between the two variables. This ensures that the estimated covariance of the fused variable will always be greater (in the positive definite sense) than the true covariance regardless of the actual correlation between the variables. In the literature, this is often referred to as a 'consistent' estimate, while others use the term 'conservative' estimate. In Appendix B we explore this property in detail and clarify these definitions.

The CI method avoids the double counting problem at the expense of a reduction in performance, as the method takes a conservative approach to using the information. In the context of collaborative localisation, Carrillo-Arce, Nerurkar *et al.* [87] demonstrated an EKF-based collaborative localisation filter using CI, while Zamani and Hunjet [88] allude to how the principles of covariance intersection can be adapted for use in a minimum-energy filter. Extensions to CI, such as split covariance intersection [89], [90], and other covariance upper-bounding techniques [80] have also been developed for collaborative localisation.

The common element that makes CI and other bottom-up methods attractive is that they only require local communication between robots when a relative measurement is made from one robot to another. The downside is that the measurement information is not used to its maximum potential to improve estimates of other robots in the network.

2.4.2 The Top-Down Approach

In the top-down approach, a centralised filter is designed which estimates the joint state of the entire network with no consideration of communication constraints. Because the centralised filter has access to all the information in the network, it can accurately track the data provenance and avoid the double counting problem. The filter equations are then decomposed, and often further approximated, so that they can be computed in a decentralised way and distributed among the robots in the network.

Roumeliotis and Bekey provide an insightful analogy to motivate the shift in thinking from individual robots each performing their own localisation task, to thinking of the system as a whole;

> "The key idea for performing collective localisation is that the group of robots must be viewed as one entity — the 'group organism' — with multiple 'limbs' (the individual robots in the group)".
>
> "The group organism has access to a large number of sensors [and] it spreads itself across a large area, and thus it can collect far more rich and diverse exteroceptive information".
>
> "When two robots communicate for information exchange, this can be seen as the 'group organism' allowing information to travel back and forth from its limbs". [91]

Roumeliotis and Bekey [79] were the pioneers of this approach, demonstrating how the equations of a centralised EKF could be distributed among the robots in the network to create a set of distributed filters that are equivalent to the original centralised filter. The key to this approach is in a decomposition of the covariance matrix which allows each robot to perform the prediction step of the filter independently with no communication. The limitation is that in order to process a measurement made by any robot, full all-to-all communication is required to re-synchronise the covariance matrix and update each robot's state. Despite this restriction, Roumeliotis and Bekey provided the foundation for the top-down approach to localisation which, in theory, provides superior localisation performance as it maximally utilises all measurement data while correctly accounting for interdependencies through the covariance matrix.

In the 20 years since its creation, there has been a large body of research that builds on this concept of top-down collaborative localisation. For example, Basu, Gao *et al.* [92], Dieudonne, Labbani-Igbida and Petit [93], and Martinelli and Siegwart [94] explore the geometric structure of relative sensor measurements and how they influence the observability of the localisation problem. Mourikis and Roumeliotis [95] investigates how the topology of the relative position measurement graph (RPMG) affects filter convergence and uncertainty bounds.

Regardless of the sensor measurement types or topology, a major limitation of Roumeliotis and Bekey's filter design is that it still requires full connectivity of the communication network at each time a measurement is made. Kia, Rounds and Martinez [96] provides a reformulation of the EKF update step so that, instead of all-to-all communication required, information is compiled at an 'intermediate master' and then propagated to all other robots. This reduces the number of messages that need to be sent through the network but does not relax the constraint on the network that all robots must be reachable every time an update is performed.

Leung, Barfoot and Liu [97] proposes an alternative approach, similar in nature to the bookkeeping approach of Bahr, Walter and Leonard. In their formulation, each robot estimates the state of the entire network. Measurements received from other robots are stored until reaching a 'checkpoint', at which time the robot can be certain that it has received all measurements up to a given time step. At this point, the robot can update its estimate using the measurements and then apply the Markov property to discard the measurement history. They show that this approach will eventually reach the centralised-equivalent estimate, but this is dependent on the connectivity and topology of the communication network.

Rather than trying to exactly reconstruct the centralised-equivalent estimate, Luft, Schubert *et al.* [98] demonstrate how the communication requirements can be reduced by only performing a measurement update on a subset of the robots in the network. The filter is based on the same distributed EKF from [79], however, the update step is modified. Firstly, sensor measurements of landmarks made by each robot are used to update the state estimate of only that robot. This is equivalent to the partial update EKF, otherwise known as the Schmidt Kalman filter (SKF) or the 'consider' Kalman filter [51], [99]. This modification means that no communication is required between robots when any robot makes a measurement of a landmark. Even under this partial update framework, for relative measurements between robots, communication between all robots is still required to reconstruct some required terms in the covariance matrix. Thus, Luft, Schubert *et al.* describes a method to approximate the required terms while only requiring communication between the two robots involved in the relative measurement. This approximation sacrifices filter consistency, meaning that the covariance estimate can no longer be guaranteed to be accurate (i.e. weakly consistent as per Appendix B). Despite the partial update and loss of consistency, simulation results show that the filter performs quite well compared to the centralised EKF.

In Chapter 4, we analyse in more detail the structure of the centralised EKF, the Schmidt-Kalman filter, and Luft, Schubert *et al.*'s approximated version.

2.4.3 Collaborative Pose Estimation

Despite the numerous advances in filter design for pose estimation which we discussed in Section 2.3.1, the vast majority of collaborative localisation algorithms utilise only basic filter designs such as the EKF. This is perhaps understandable, as the focus of collaborative localisation is not necessarily on the best filter performance, but on analysing the communication topology, information dependencies and data flows through the network. However, if one seeks to implement such a collaborative filter in an environment where the robot dynamics are more complex than a simple two-wheeled drive, then the EKF will soon become the limiting factor, regardless of what communication topology or information structure is used.

Some progress in this area has been made, for example by Zamani and Hunjet [88], who demonstrate a 'bottom-up' collaborative localisation algorithm for heterogeneous robots in $SE(3)$, which is based on a minimum energy filter design and uses an adapted form of covariance intersection to avoid the double counting problem. A different algorithm, designed by Jung, Brommer and Weiss [100], uses the approach of Luft, Schubert *et al.* [98] to create a collaborative localisation algorithm for robots in $SE(3)$ with inertial measurements and a vision-based system for relative measurements. Rather than using a standard EKF, they use a quaternion error-state extended Kalman filter (ES-EKF), which is better designed for estimation on $SE(3)$ but retains the same structure required to apply the decentralisation techniques from Luft, Schubert *et al.*

2.5 The Value of Simulation and Experimentation

Having explored the wide variety of approaches and algorithms for collaborative localisation, one is faced with the challenge of testing and evaluating their performance in order to make meaningful comparisons. Given the complexity of the environment, the sensor systems, and the localisation algorithms themselves, it is difficult to construct mathematical guarantees on performance without making broad assumptions or approximations. Conversely, just because such proof cannot be found does not mean that the algorithm will perform poorly.

Simulations and real-world experiments are two tools that can be used to evaluate algorithm performance using data that closely resembles the intended use case. While it is not possible to test every single scenario, properly constructed simulations and experiments can help to build confidence in the expected level of performance, as well as validate any assumptions that are made in the design process. They can also be used to compare the performance of different algorithms by providing a standardised, repeatable set of tests on which to measure.

It is important to note that 'performance' can not be measured by a single metric alone. In the context of collaborative localisation, some measures of performance include, but are not limited to;

- Localisation accuracy: How close the estimated pose is to the true pose. Many metrics exist to measure accuracy, for example, position accuracy may be measured as the Euclidean distance between the true and estimated positions, and for attitude one may use the angle between the two orientations. However, there is no universal way to measure the error between two poses as a scalar metric.
- Consistency: In cases where the algorithm provides an estimate of the error in the estimate, this can be compared with the measured error to determine if the estimated error is accurate.
- Bandwidth: How much data was sent between nodes, both in the number of messages, and message size.
- Network topology: Whether the filter requires specific network topologies in order to function, or if the performance of the filter is affected by the network topology.
- Memory and computation: How the memory usage and computation requirements of the filter change over time, and how they scale as the number of robots increases or the topology of the network changes.

Furthermore, for each of these measures, one may be interested in the best-case, typical, worst-case, or some other secondary metric.

2.5.1 Simulation

Simulations can be constructed at varying levels of fidelity, depending on the requirements. Numerical simulations with basic sensor models and simple vehicle trajectories can be easily constructed in any programming language. They are useful at the initial stages of algorithm development to test basic functionality and implementation correctness. More detailed simulation environments such as Gazebo[1], provide a richer set of tools and models which allow for integration testing with other system components, and basic physics and sensor modelling. Flight simulators, such as X-Plane[2] and JSBSim[3], provide detailed modelling of aerodynamics and flight dynamics. A recent project by Microsoft, AirSim[4], uses Unreal Engine to create realistic renderings of the environment, which is especially useful for modelling vision-based sensors.

With the relatively low cost of simulation, one can run large numbers of simulations to understand how the algorithms perform in different conditions, testing over a range of trajectories, sensor properties, noise parameters, and other properties. Ground truth information is readily accessible, as it is often constructed as part of the simulation process, which allows for easy comparisons between estimate and true system states. However, there are limits to how closely simulation data can match real-world data, often called the sim-to-real gap, which restricts how generalisable simulation performance is to the real world. It does provide a good first step, for if a filter does not work in simulation then it is almost certain it will fail in the real world.

While simulations are a valuable tool to compare performance between algorithms, in the collaborative localisation literature there is no agreed-upon standard or benchmark set of simulation tools or profiles. Without a benchmark, authors are free to pick and choose simulation results that support their argument and disregard those that don't. Most papers also provide little or no details on simulation parameters and do not make the simulation code or raw simulation results publicly available. This makes replicating most simulation results near-impossible, and one must trust the authors that the filter implementation is correct and matches what is written in the paper.

2.5.2 Datasets from Real-World Experiments

Simulations do have their place in early prototyping and validation, however, one cannot rely solely on simulation data as a means to validate filter performance. Real-world experiments, while more costly and time-consuming to perform, provide data that cannot be matched by simulations. Due to the complexity and cost, it is common for research groups to perform experiments and make the data available for others to use. Compared with other robotics experiments, the challenges of performing experiments for collaborative localisation are further complicated by the need for multiple vehicles, and the management of interactions between the vehicles, which we discuss further in Section 2.5.4. This has resulted in a noticeable lack of high-quality datasets available in the literature for collaborative localisation.

Perhaps the most prominent and widely used dataset for collaborative localisation is the UTIAS Multi-Robot Collaborative Localisation and Mapping

[1] `https://gazebosim.org/home`
[2] `https://www.x-plane.com/`
[3] `https://jsbsim.sourceforge.net/`
[4] `https://github.com/microsoft/AirSim`

FIGURE 2.3: MRCLAM experimental setup showing mobile robot configurations, landmarks and roof-mounted VICON system. [101]

(MRCLAM) dataset [101]. Published in 2011, its popularity is due to its good quality data, accessible format, and uniqueness in that it is one of the few publicly available datasets of this kind. The dataset consists of a fleet of five wheeled robots navigating through a 15×8 m open space with 15 fixed landmarks located within the space. Each robot is equipped with wheel odometry sensors, a camera, and a visual marker, which can be seen in Figure 2.3. The camera on each robot detects the visual markers on other robots, as well as the fixed landmarks, and produces a range and bearing measurement of each detected marker. The ground truth pose of each robot is recorded through a VICON motion capture system to a reported accuracy on the order of 10^{-3} m. Sullivan, Grainger and Cazzolato [102] provides some analysis of the sensor errors in the MRCLAM dataset, however, the data also contains several mistakes that do not appear to have been identified in the literature, which we discuss in Appendix D.

In the last decade, a number of attempts at producing collaborative localisation datasets have been made, but with limited success and uptake within the community. Hartzer and Saripalli [103] collected experimental data from a large-scale wheeled vehicle to demonstrate their collaborative localisation algorithm. The data collected included wheel odometry, steering angle, Real-Time Kinematic (RTK) GNSS position, IMU measurements, and range measurements from a UWB sensor. The dataset was intended to represent a range-based collaborative localisation system, with each vehicle equipped with a UWB sensor, capable of measuring relative distances between vehicles. However, due to personnel limitations, only one vehicle was moving, while the other UWB sensors were stationary. The experiment was performed with a simple straight-line trajectory of the vehicle, and with only two other UWB sensors in stationary positions, which is not sufficiently complex to properly evaluate localisation algorithms. And, with only one moving vehicle, its utility in evaluating collaborative localisation algorithms is limited.

Güler, Abdelkader and Shamma [104] perform different range-based collaborative localisation experiments on two UAVs. One of the UAVs is equipped with three UWB sensors, and the second UAV is equipped with one. Experiments are performed both indoors and outdoors, with several trajectories collected for each. While the experimental data was not published, the experiment as described in the paper gets close to what is desired in a good collaborative localisation dataset. The primary limitation of the experiment is that it only uses two UAVs, which is not sufficient to evaluate many arbitrary-scale multi-robot collaborative localisation algorithms.

2.5.3 Hybrid Datasets from Real and Synthetic Data

Given the lack of any other high-quality public experimental datasets for collaborative localisation, especially for 3-dimensional trajectories, many researchers must rely on simulation. However, rather than using purely synthetic data, a hybrid of real-world and synthetically generated data can be used to increase realism and reduce the sim-to-real gap.

A common example of this is the European Robotics Challenge Micro Aerial Vehicle (EuRoC MAV) dataset [105], which is a sequence of 11 datasets each with a single MAV navigating through a complex environment. The dataset provides 200 Hz IMU data from the MAV, as well as an onboard stereo camera. For 6 of the sequences, ground truth pose is provided by a VICON motion tracking system, but for the remaining 5 sequences, only ground truth position is provided by a laser tracking station. Although the EuRoC MAV dataset was originally created to evaluate SLAM and Visual Inertial Odometry algorithms, Jung, Brommer and Weiss [100] demonstrate how a collaborative localisation dataset can be generated from the original data. To do this, several sequences from the EuRoC MAV dataset are combined to simulate multiple aircraft flying simultaneously. Each sequence is offset from its original location to increase the distance between the trajectories of each aircraft. Using the ground truth data of the combined sequence, synthetic relative measurements are generated to simulate a sensor onboard the aircraft detecting another aircraft. Jung, Brommer and Weiss also perform a similar process using data from the Technical University of Munich Visual-Inertial (TUM VI) benchmark dataset [106].

Chakraborty, Taylor *et al.* [107] also perform a similar procedure, based on a dataset of 50 fixed-wing UAVs flying simultaneously. However, because these aircraft have no sensors to measure the relative positions of other aircraft, these measurements are again generated synthetically. The motivation behind these hybrid-data approaches comes from a lack of available high-quality 3-dimensional collaborative localisation datasets. The hybrid-data approach at least allows the algorithms to be tested using real-world IMU data and real-world aircraft trajectories, but the fact that the relative position measurements between aircraft need to be synthetically generated is a major limitation.

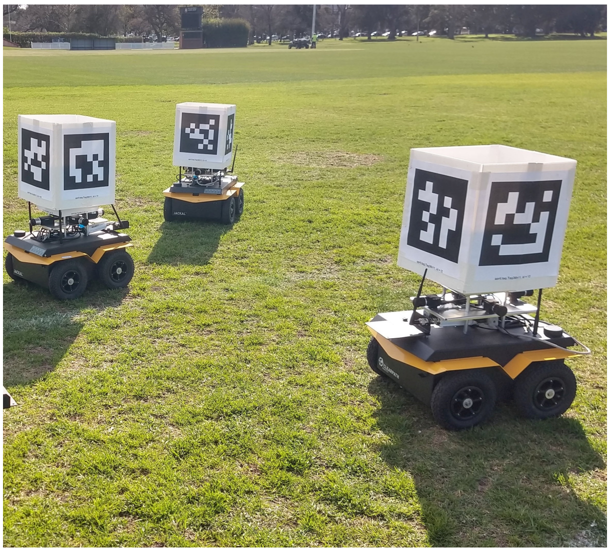
2.5.4 The Challenges of Creating a High-Quality Dataset

There are several reasons why such a dataset does not currently exist. Any kind of real-world robotics experimentation is expensive, time-consuming and technically challenging work. Aerial robotics adds another layer of complexity due to weight constraints, runtime limitations, fragility of components, and regulatory burdens. Performing experiments with multiple aerial robots adds even more complexity due to the increase in points of failure, risk of collision, network constraints, and number of people required to assist. The recent work by Baca, Petrlik *et al.* [108] describes the countless challenges of constructing and operating a fleet of autonomous vehicles. These reasons are also why datasets such as MRCLAM are so valuable to the research community. They provide a common tool to evaluate and compare different collaborative localisation algorithms without every single researcher having to dedicate time and money to create their own complex experiment.

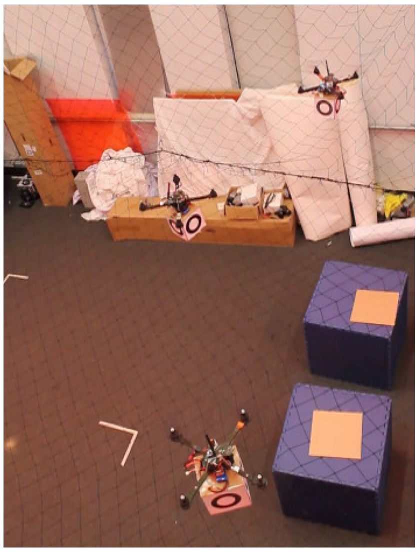
Another factor that makes collaborative localisation experiments challenging is how to generate relative measurements between robots. Relative measurements may take different forms, including relative bearing, distance, position, velocity, or pose. There must also be a method to identify which robot the measurement corresponds to. Vision-based systems are a common way to generate relative measurements. With a calibrated camera fixed to one robot, other robots can be identified in the image and the image location can be mapped to a bearing measurement. If the camera is capable of measuring depth, such as a stereo camera, or by structured light, then a distance measurement can also be generated. The challenge is then in the identification of the robot in the image, for which there are many approaches.

Direct detection of UAVs using depth cameras has been studied by Vrba, Heřt and Saska [109], however, these methods require high-quality cameras and are only useful at short ranges. They also do not provide a method of uniquely identifying a detected robot. Given the challenges of direct detection, a common method is to place an easily identifiable marker on each robot to aid in detection and identification.

The use of fiducial markers, such as barcodes or QR codes, is a common method as it serves as both the method of locating the robot and identification. The approach taken in the MRCLAM dataset is to use large barcode markers on each robot which encode a unique identifier for each robot. We see a similar approach in the system described by Krajník, Nitsche *et al.* [110] and in the experimental setup of Zhu and Kia [80], who use ARTags [111] which are a 2D square fiducial marker. Sullivan [112] also makes use of AprilTag fiducial markers [113] to generate relative measurements between robots, shown in Figure 2.4. The 2D fiducial markers such as ARTags, AprilTags and QR codes have the additional advantage that they provide a way to measure the relative pose of the robot based on a homography estimation. The downside to these methods is that the maximum range is limited by the size of the markers and the resolution of the camera. Additionally, 2D fiducial markers are only visible from some angles, and thus a set of markers must be arranged in a



(A) Sullivan [112]



(B) Krajník, Nitsche *et al.* [110]

FIGURE 2.4: Examples of fiducial markers used for relative measurements between robots.

cube to guarantee identification from all angles. This can often result in bulky, obtrusive assemblies attached to vehicles, like those shown in Figure 2.4, which can degrade operational performance and impact co-located sensors, especially on aerial vehicles.

Rather than using passive fiducial markers, the use of active markers such as flashing Light Emitting Diodes (LED) appears to be a promising idea. Walter, Staub *et al.* [114], [115] show how LEDs flashing at different frequencies can be used to track, identify and provide bearing information measurements to multiple robots. They also show how the perceived brightness of the LED can be used to estimate the distance to a varying degree of accuracy. While LED-camera communication systems are not novel [116], the addition of an Ultraviolet (UV) band-pass filter on the camera lens, and the selection of UV emitting LEDs for the markers allows for a significant reduction in background noise in the image and makes the tracking and identification problem significantly easier.

Beyond vision sensors, there are many other kinds of sensor types. Ultra-Wideband (UWB) sensors are increasing in popularity due to their high accuracy, low cost and small form factor. UWB sensors are radio beacons that can measure time-of-flight to other UWB sensors, and thus measure distance. Despite being a relatively new technology, interest and experimentation with these sensors for collaborative localisation is already underway [103], [104], [117]. UWB is not a perfect solution though, as its range is limited to around 100 m, it degrades quickly when obstacles interrupt line of sight (LOS) and measurements contain biases that need to be corrected. Also, all UWB transmitters share the same frequency spectrum and must implement a time division multiple access (TDMA) scheme in order to avoid collisions. As the number of UWB transmitters increases, the number of measurements each one can make in a given amount of time will decrease.

2.6 Summary

Based on the detailed survey and analysis of the existing literature we have presented, several areas of interest and research avenues become apparent. While there has been significant effort in the development of high-performance filtering algorithms for autonomous vehicle localisation, little of this research has extended to multi-vehicle collaborative localisation. Filtering approaches for collaborative localisation have predominantly focused on simpler algorithms, such as the extended Kalman filter, perhaps to separate the complexities of the filtering algorithms from the additional challenges of multi-vehicle interactions. However, in order to obtain good performance on real-world systems with nonlinear dynamics such as UAVs, we need an integrated solution that addresses the complexities of both of these aspects together. This leads to a gap in the existing literature at the intersection of state-of-the-art filtering and collaborative localisation.

Secondly, the lack of a high-quality multi-vehicle 3-dimensional experimental dataset makes testing and evaluating different collaborative localisation approaches more difficult. Without a benchmark dataset with realistic trajectories, physical sensor measurements and true inter-vehicle relative measurements, the community is unable to compare the performance of different algorithms or validate whether their theoretical and simulation-based results match with real-world performance. While some datasets have been published in this area, all of them have significant limitations or drawbacks, whether they be 2-dimensional only, do not produce inter-vehicle measurements, have poor ground-truth data, or do not make public all required information. This highlights the significant potential for the impact of a new collaborative localisation dataset that addresses these issues and provides the community with a new benchmark to test, analyse and compare different collaborative localisation approaches.

As highlighted previously in the Introduction, it is these two areas of research that motivate the work in this thesis and form the basis of the thesis contributions.

Chapter 3

A Minimum Energy Filter for Robot Localisation

Through the survey of the literature in the preceding chapter, it is clear that high-performance localisation algorithms remain an area of intense research focus. There is a large body of work encompassing the Kalman filter and its derivatives, including the EKF, MEKF, UKF and IEKF to name a few. These stochastic filters have become the localisation algorithm of choice in many industries whereas, comparatively speaking, deterministic filters such as the minimum energy filter remain on the fringes. The literature is sparse on deterministic filters for localisation, especially in the area of collaborative localisation, however, there is significant potential in these algorithms to address some drawbacks of traditional stochastic algorithms.

In this chapter, we present the derivation of a new filter for robot localisation based on the principle of minimum energy filtering. We build on existing work by Saccon, Trumpf *et al.* [77], which provides a general framework in which to construct second-order minimum energy filters on Lie groups, by specialising the results to the particular case of interest. As we will see, for the minimum-energy filter, this is neither a straightforward nor simple task.

We begin in Section 3.1 by defining the problem and describing the models for the movement of the robot and the sensor measurements. In Section 3.2 we introduce the general minimum energy filtering framework and then derive the explicit representation of the filter in Section 3.3. In Section 3.4 the filter is transformed from continuous time to discrete time to facilitate implementation on a computer and we discuss potential tuning strategies. Much of the work in this chapter is based on the author's previously published work [2] and preprints [3].

While the filter derived in this chapter may find use as a high-performance single-vehicle localisation algorithm, the goal of this chapter is in laying the groundwork for eventual extension into a multi-vehicle collaborative localisation algorithm which we present in the latter half of Chapter 4.

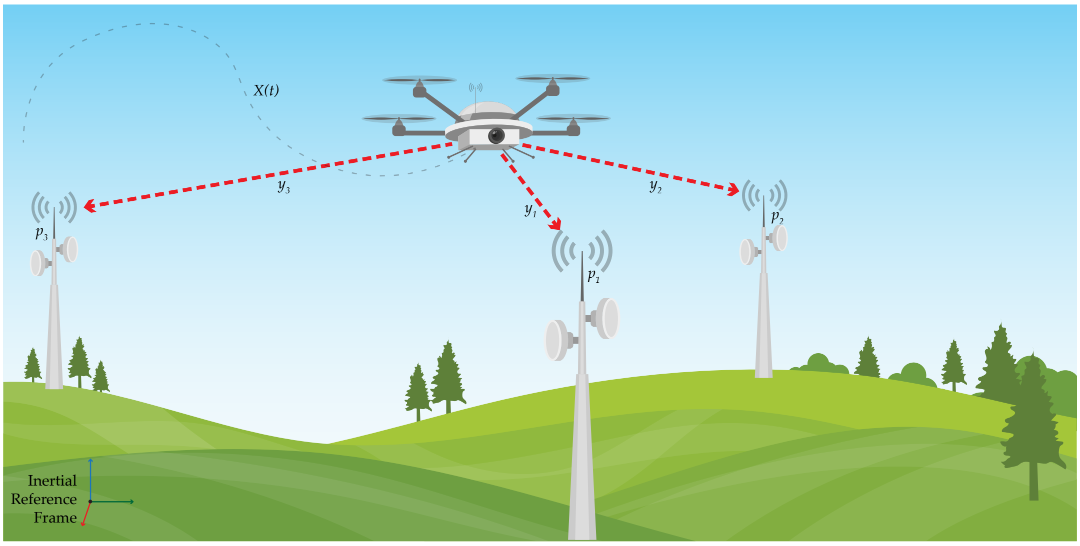


FIGURE 3.1: Illustration of the proposed single-vehicle localisation problem.

3.1 Problem Formulation

As discussed in Chapter 2, there are many variations on the collaborative localisation problem, especially when considering the different sensors that may be available to the robot. We wish to examine a single instance of the robot localisation problem, chosen in such a way that it captures the main structure of a real-world localisation problem, without introducing unnecessary complexity.

A key element of this is the use of an Inertial Measurement Unit (IMU) for measuring linear acceleration and angular velocity, as they are a ubiquitous component on almost any mobile robot, especially aerial robots. Incorporating IMU sensor data into a filter can be quite challenging, due to the double-integrator relationship between position and acceleration, the coupling of rotational and linear motion due to Coriolis and centripetal forces, the additional effects of gravity, and the possibility of time-varying sensor biases. Thus, if we wish to design a realistic filter for mobile robots, modelling an IMU sensor is essential.

As for other sensors, there is an almost limitless choice, including GNSS, LIDAR, vision and many others. To keep things relatively simple, we will consider a sensor that measures the relative positions of a set of known landmarks. This is akin to a radar sensor, where the range and bearing to an identified point are measured, but without the cross-correlation between components. An illustration of the problem formulation is shown in Figure 3.1.

3.1.1 State Representation and Kinematics

Consider a rigid-body vehicle in 3-D space in a gravitational field. The orientation, R, position, x, and linear velocity, v, of the body-fixed frame with respect to the earth-fixed frame are all expressed in the coordinates of the inertial frame. Following similar formulations in the literature [118, Sec. 4.3.5], [68], [63], the kinematics of the vehicle can be represented by

$$R \in \mathrm{SO}(3) \qquad x \in \mathbb{R}^3 \qquad v \in \mathbb{R}^3 \tag{3.1}$$
$$\dot{R} = R\omega_\times \qquad \dot{x} = v \qquad \dot{v} = Ra + g \tag{3.2}$$

where $\omega \in \mathbb{R}^3$ is the angular velocity and $a \in \mathbb{R}^3$ is the linear acceleration of the body-fixed frame with respect to the inertial frame, expressed in the body-fixed frame. The acceleration of the earth-fixed frame due to the gravitational field is represented by $g \in \mathbb{R}^3$ and is approximated by $g \approx \begin{bmatrix} 0 & 0 & -9.81 \end{bmatrix}^\top$ in the East-North-Up coordinate frame. Additionally, the rotation of the earth is assumed to be negligible over the time scales and distances of the scenario.

Measurements from MEMS IMU sensors are commonly prone to time-varying biases [28], which can introduce significant errors if unaccounted for. To account for this, we model an offset of θ to the angular velocity measurements and an offset of ϕ to the linear acceleration measurements. These biases vary slowly over time according to some unknown processes, δ_θ and δ_ϕ, weighted by $B_\theta, B_\phi \in \mathbb{R}^{3\times 3}$ respectively. This gives

$$\theta \in \mathbb{R}^3 \qquad \phi \in \mathbb{R}^3 \tag{3.3}$$
$$\dot{\theta} = B_\theta \delta_\theta \qquad \dot{\phi} = B_\phi \delta_\phi. \tag{3.4}$$

Using the product group structure presented in Appendix A.3.2, the state of the vehicle is represented as

$$X := \left(\begin{bmatrix} R & x & v \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \theta, \phi \right) \in \mathrm{G}.$$

Under this Lie group representation of the state, the vehicle kinematics, (3.2) and (3.4), can then be represented by the left translation operation by

$$\dot{X} = X\Omega$$
$$\Omega = \left(\begin{bmatrix} \omega_\times & R^\top v & a - R^\top g \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, B_\theta\delta_\theta, B_\phi\delta_\phi \right) \in \mathfrak{g}.$$

We assume that there is some initial start-time of the system, t_0, and the initial state at that time is represented by $X(t_0) = X_0$.

3.1.2 IMU Sensor Model

The vehicle is equipped with a strap-down IMU, which measures linear acceleration u_a and angular velocity, u_ω, in the body-fixed frame. The measurement vector u is then

$$u := \begin{bmatrix} u_\omega \\ u_a \end{bmatrix} = \begin{bmatrix} \omega + \theta + B_\omega \delta_\omega \\ a + \phi + B_a \delta_a \end{bmatrix} \in \mathbb{R}^6,$$

which is composed of the true angular velocity, ω, true acceleration, a, the time-varying biases, θ and ϕ, and $\delta_\omega, \delta_a \in \mathbb{R}^3$ which are unknown error signals weighted by $B_\omega, B_a \in \mathbb{R}^{3\times 3}$.

Substituting the measurement model into the vehicle kinematics gives

$$\dot{X} = X\left(\lambda(X,u) + B(\delta)\right) \tag{3.5}$$

where $\lambda : \mathrm{G} \times \mathbb{R}^6 \to \mathfrak{g}$ is given by

$$\lambda(X,u) := \begin{bmatrix} u_\omega - \theta \\ R^\top v \\ u_a - \phi - R^\top g \\ 0 \\ 0 \end{bmatrix}^\wedge, \tag{3.6}$$

and the linear map $B : \mathbb{R}^{12} \to \mathfrak{g}$ is given by

$$B(\delta) := (\check{B}\delta)^\wedge, \qquad \delta := \begin{bmatrix} \delta_\omega \\ \delta_a \\ \delta_\theta \\ \delta_\phi \end{bmatrix}, \qquad \check{B} := \begin{bmatrix} -B_\omega & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & -B_a & 0 & 0 \\ 0 & 0 & B_\theta & 0 \\ 0 & 0 & 0 & B_\phi \end{bmatrix}. \tag{3.7}$$

3.1.3 Landmark Measurement Model

In addition to the IMU, each vehicle is equipped with a sensor that measures relative translations to multiple fixed landmarks, $l \in \boldsymbol{L}$, in the environment. The sensor measurement made by the vehicle of a single landmark l is denoted as $y_l \in \mathbb{R}^3$. It is modelled as the true relative translation, $h_l : \mathrm{G} \to \mathbb{R}^3$, corrupted by some unknown measurement error, $\epsilon_l \in \mathbb{R}^3$;

$$h_l(X) = R^\top (p_l - x), \tag{3.8}$$
$$y_l(t) = h_l(X(t)) + D\epsilon_l(t), \tag{3.9}$$

where $p_l \in \mathbb{R}^3$ is the known position of the landmark in the earth-fixed coordinate frame and $D \in \mathbb{R}^{3\times 3}$ is invertible.

3.2 Abstract Minimum Energy Filter

In this section, we define the minimum energy filtering problem on the system and present the resulting filter equations for the second-order optimal minimum energy filter.

Initially, we pose the filtering problem in continuous time and suppose that both the IMU measurements, u, and the sensors measurements, y, are continuous-time signals that are always present. By using this formulation, we can take advantage of the existing body of work in the literature on continuous-time minimum-energy filtering on Lie groups. This yields a state estimate in the form of a continuous-time differential equation, but only in an abstract form which cannot be implemented on a computer. In the proceeding sections, we will transform these abstract equations into an explicit set of continuous-time matrix equations and then present one approach to discretising the continuous-time equations. This will result in a set of explicit discrete-time filter equations capable of being implemented on a digital computer.

3.2.1 Minimum Energy Filtering Problem Definition

Following the behavioural approach of Polderman and Willems [119], which is discussed in more detail in Appendix B, the signals u and y form the manifest signal, as they are the only signals that are directly observable. The remaining signals, X, δ, and ϵ form the latent variable signal. In the context of filtering, The time domain, $\mathbb{T} = [t_0, t]$, represents the interval between the initial time, t_0, and the current time t. Thus, the manifest and latent variables signals are, respectively,

$$\left(u_{[t_0,t]}, y_{[t_0,t]}\right) \in \mathbb{M}^{\mathbb{T}} := (\mathbb{R}^6 \times \mathbb{R}^{3\times|L|})^{\mathbb{T}}$$
$$\left(X_{[t_0,t]}, \delta_{[t_0,t]}, \epsilon_{[t_0,t]}\right) \in \mathbb{L}^{\mathbb{T}} := (\mathrm{G} \times \mathbb{R}^6 \times \mathbb{R}^{3\times|L|})^{\mathbb{T}}.$$

From this, the full behaviour of the system can be defined as

$$\mathfrak{B}_f := \left\{ (u_{[t_0,t]}, y_{[t_0,t]}, X_{[t_0,t]}, \delta_{[t_0,t]}, \epsilon_{[t_0,t]}) \mid (3.5) \text{ and } (3.9) \text{ are satisfied} \right\}.$$

Naturally, this results in an infinite number of trajectories in the full behaviour, as there are an infinite number of combinations of latent variable signals that can satisfy the system model (3.5) and the measurement model (3.9). Out of all the trajectories in the full behaviour, we wish to determine the system state corresponding to the 'best' trajectory. To do this, we must impose a measure or cost on the trajectories in the full behaviour to define what we mean by 'best'.

Inspired by related works, [39], [48], [77], we will consider a cost functional on the system which measures the energy in the two error signals, δ and ϵ, which we assume to be square integrable. In addition to the signal energy, we will also impose a cost on the initial state of the system, to represent the prior information known about the initial state. Together, the cost functional is defined as

$$J_t(\delta_{[t_0,t]}, \epsilon_{[t_0,t]}, X_0) := \frac{1}{2} J_0(X_0) + \frac{1}{2}\int_{t_0}^{t} \|\delta(\tau)\|_W^2 + \sum_{l \in \mathbf{L}} \|\epsilon_l(\tau)\|_{Q_l}^2 \mathrm{d}\tau \tag{3.10}$$

where $J_0 : \mathrm{G} \to \mathbb{R}$ is the cost on the initial state, X_0, with a unique global minimum. The terms $Q_l \succ 0 \in \mathbb{R}^{3\times 3}$ and $W \succ 0 \in \mathbb{R}^{12\times 12}$ allow us to add relative weights to each of the signal energies and can be modified to tune the cost as desired (see A.1.4 for an explanation of the weighted norm).

The optimal trajectory of the system is defined as the trajectory in the full behaviour that minimises the cost functional. The optimal state trajectory, denoted by $X^*_{[t_0,t]} \in \mathrm{G}^{\mathbb{T}}$, is then the state component of the optimal trajectory. We then define the terminal point of X^* to be the estimate of the current state of the system at the current time, t, denoted by $\hat{X}(t) \in \mathrm{G}$. More formally,

$$\hat{X}(t) := X^*_{[t_0,t]}(t).$$

Note that the full behaviour, the cost functional, and the state estimate are all defined in terms of the current time, t. As time progresses, say from t to t', the cost functional, $J_{t'}$, is defined over an extended time domain $\mathbb{T} = [t_0, t']$, resulting in a different optimal trajectory, $X^*_{[t_0,t']}$, and thus a different terminal point $\hat{X}(t')$. In the derivation of the filter, we will see that we can define the state estimate, $\hat{X}$, recursively as an ODE. This means we do not have to compute the entire optimal trajectory, X^*, for every time instance t, nor must we store the entire measurement trajectories, u and y.

3.2.2 The Second-Order Minimum Energy Filter

The filtering problem described in Section 3.1 and the accompanying cost functional in (3.10) is a particular case of the filtering problem described by Saccon, Trumpf *et al.* [77]. Given this, we adapt the solution from [77] to create a second-order minimum energy filter for the 15-DOF localisation problem.

The second-order minimum energy estimate for the state of the system described above is defined by a pair of ordinary differential equations (ODE) which describe the time evolution of the state estimate, $\hat{X}(t) \in \mathrm{G}$, and the associated gain operator, $K(t) : \mathfrak{g}^* \to \mathfrak{g}$. They are given by

$$\dot{\hat{X}} = \hat{X}\left(\lambda(\hat{X}, u) + K \circ r\right) \tag{3.11}$$

$$\dot{K} = A \circ K + K \circ A^* - K \circ E \circ K + B \circ W^{-1} \circ B^* - \Lambda_{K \circ r} \circ K - K \circ \Lambda^*_{K \circ r} \tag{3.12}$$

with initial conditions

$$\hat{X}(t_0) = \arg\min_X J_0(X)$$

$$K(t_0) = \left(T_e L^*_{\hat{X}_0} \circ \mathrm{Hess}\, J_0(\hat{X}_0) \circ T_e L_{\hat{X}_0}\right)^{-1}.$$

The operator Λ is the connection function (see A.2.2), and the residual, $r = r(\hat{X}, y) \in \mathfrak{g}^*$, is given by

$$r := \sum_{l \in \boldsymbol{L}} T_e L^*_{\hat{X}} \circ \left(\left(M_l \circ (y_l - \hat{y}_l) \right) \circ \mathbf{d}\hat{y}_l \right) \tag{3.13}$$

where

$$\begin{aligned} \hat{y}_l &:= h_l(\hat{X}) \\ M_l &:= (D^{-1})^\top Q_l D^{-1}. \end{aligned}$$

The operators $A = A(\hat{X}, u)$ and $E = E(\hat{X}, y)$ are defined by

$$A := \mathbf{d}_1 \lambda(\hat{X}, u) \circ T_e L_{\hat{X}} - \mathrm{ad}_{\lambda(\hat{X},u)} - T_{\lambda(\hat{X},u)}, \tag{3.14}$$

$$E := -T_e L^*_{\hat{X}} \circ \left(\sum_{l \in \boldsymbol{L}} E_l \right) \circ T_e L_{\hat{X}}, \tag{3.15}$$

where

$$E_l := \left(M_l \circ (y_l - \hat{y}_l) \right)^{T_{\hat{X}}\mathrm{G}} \circ \mathrm{Hess}\, \hat{y}_l - (\mathbf{d}\hat{y}_l)^* \circ M_l \circ \mathbf{d}\hat{y}_l$$

and $(.)^{T_{\hat{X}}\mathrm{G}}$ is the exponential functor (see A.2.1).

3.3 Explicit Matrix Representation

The filter equations presented in Section 3.2.2, describe a set of abstract operators on the Lie group G and the corresponding Lie algebra. In order to be able to implement this filter on a computer, we need a basis in which to operate and a matrix representation of all the abstract operators used in the filter. In this section, we present a series of definitions for the matrix representations of the operators, r, A, E, and K in order to be able to derive an explicit implementation for the filter estimate $\hat{X}$. For each of these definitions, we also present a corresponding lemma that shows how the term can be numerically calculated, enabling implementation on a computer.

Definition 3.1. *Consider the adjoint action of the Lie algebra,* $\mathrm{ad}_\Gamma : \mathfrak{g} \to \mathfrak{g}$, *where* $\Gamma \in \mathfrak{g}$. *The matrix representation,* $\check{\mathrm{ad}}_\Gamma \in \mathbb{R}^{15\times 15}$, *of the adjoint action is implicitly defined by the equation*

$$(\mathrm{ad}_\Gamma \Psi)^\vee = \check{\mathrm{ad}}_\Gamma \Psi^\vee$$

for all $\Psi \in \mathfrak{g}$.

Lemma 3.2. *The matrix representation* $\check{\mathrm{ad}}_\Gamma$ *from Definition 3.1 is given by*

$$\check{\mathrm{ad}}_\Gamma = \begin{bmatrix} \Gamma_{R\times} & 0 & 0 & 0 & 0 \\ \Gamma_{x\times} & \Gamma_{R\times} & 0 & 0 & 0 \\ \Gamma_{v\times} & 0 & \Gamma_{R\times} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Proof. For a Lie algebra, the adjoint action is given by the Lie bracket

$$\mathrm{ad}_\Gamma(\Psi) = [\Gamma, \Psi].$$

Applied to the product group's Lie algebra, $\mathfrak{g}$, this becomes

$$[\Gamma, \Psi] = ([\Gamma_P, \Psi_P], [\Gamma_\theta, \Psi_\theta], [\Gamma_\phi, \Psi_\phi])$$

For the matrix group, the Lie bracket is equivalent to the matrix commutator, while for the Abelian groups, the Lie bracket is zero. Thus,

$$\begin{aligned}[\Gamma, \Psi] &= (\Gamma_P\Psi_P - \Psi_P\Gamma_P, 0, 0) \\ &= \left(\begin{bmatrix} \Gamma_{R\times}\Psi_{R\times} - \Psi_{R\times}\Gamma_{R\times} & \Gamma_{R\times}\Psi_x - \Psi_{R\times}\Gamma_x & \Gamma_{R\times}\Psi_v - \Psi_{R\times}\Gamma_v \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, 0, 0 \right)\end{aligned}$$

Using the identities from Appendix A.4, we have

$$\begin{aligned}[\Gamma, \Psi] &= \left(\begin{bmatrix} (\Gamma_{R\times}\Psi_R)_\times & \Gamma_{R\times}\Psi_x + \Gamma_{x\times}\Psi_R & \Gamma_{R\times}\Psi_v + \Gamma_{v\times}\Psi_R \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, 0, 0 \right) \\ &= \begin{bmatrix} \Gamma_{R\times}\Psi_R \\ \Gamma_{R\times}\Psi_x + \Gamma_{x\times}\Psi_R \\ \Gamma_{R\times}\Psi_v + \Gamma_{v\times}\Psi_R \\ 0 \\ 0 \end{bmatrix}^\wedge \\ &= \left(\begin{bmatrix} \Gamma_{R\times} & 0 & 0 & 0 & 0 \\ \Gamma_{x\times} & \Gamma_{R\times} & 0 & 0 & 0 \\ \Gamma_{v\times} & 0 & \Gamma_{R\times} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \Psi^\vee \right)^\wedge\end{aligned}$$

and the result follows. $\square$

Definition 3.3. *Recall from (3.8) the landmark measurement model,* $h_l : \mathrm{G} \to \mathbb{R}^3$*, for a given landmark,* $l \in \mathbf{L}$*. The matrix representation,* $H_l(X) \in \mathbb{R}^{3\times 15}$*, of the derivative of the measurement model at a point* $X \in \mathrm{G}$*, is implicitly defined by*

$$\mathbf{d}h_l(X) \circ X\Gamma = H_l(X)\Gamma^\vee$$

for all $\Gamma \in \mathfrak{g}$.

Lemma 3.4. *The matrix representation* $H_l(X)$ *from Definition 3.3 is given by*

$$H_l(X) = \begin{bmatrix} h_l(X)_\times & -\boldsymbol{I}_3 & \mathbf{0}_{3\times 9} \end{bmatrix}$$

Proof. The derivative of the measurement model in an arbitrary direction $X\Gamma \in T_X\mathrm{G}$ is given by

$$\begin{aligned} \mathbf{d}R^\top(p_l - x) \circ X\Gamma &= (R\Gamma_{R\times})^\top(p_l - x) + R^\top(-R\Gamma_x) \\ &= -\Gamma_{R\times}R^\top(p_l - x) - \Gamma_x \\ &= h_l(X)_\times\Gamma_R - \Gamma_x \\ &= \begin{bmatrix} h_l(X)_\times & -\boldsymbol{I}_3 & \mathbf{0}_{3\times 9} \end{bmatrix} \Gamma^\vee. \end{aligned}$$

□

Lemma 3.5. *Consider, again, the landmark measurement model,* $h_l : \mathrm{G} \to \mathbb{R}^3$ *of a known landmark,* $l \in \mathbf{L}$, *from (3.8). The Hessian of the measurement model at a point* $X \in \mathrm{G}$ *in two arbitrary directions* $X\Gamma, X\Psi \in T_X\mathrm{G}$ *is given by*

$$\operatorname{Hess} h_l(X)(X\Gamma)(X\Psi) = -\frac{1}{2}\left(\Psi_{R\times}H_l(X)\Gamma^\vee + \Gamma_{R\times}H_l(X)\Psi^\vee\right)$$

Proof. As described in [120], the Hessian on a manifold is given by

$$\operatorname{Hess} h_l(X)(X\Gamma)(X\Psi) = \mathbf{d}\big(\mathbf{d}h_l(X) \circ X\Psi\big) \circ X\Gamma - \mathbf{d}h_l(X) \circ X\Lambda_\Gamma(\Psi) \quad (3.16)$$

where Λ is the connection function. Evaluating the first term, we have

$$\begin{aligned} \mathbf{d}\big(\mathbf{d}h_l(X) \circ X\Psi\big) \circ X\Gamma &= \mathbf{d}\big(-\Psi_{R\times}h_l(X) - \Psi_x\big) \circ X\Gamma \\ &= -\Psi_{R\times}H_l(X)\Gamma^\vee. \end{aligned}$$

Using the (0)-connection function, Λ_0, the second term evaluates to

$$\begin{aligned} \mathbf{d}h_l(X) \circ X\Lambda_\Gamma(\Psi) &= \frac{1}{2}\mathbf{d}h_l(X) \circ X[\Gamma, \Psi] \\ &= \frac{1}{2}\Big(-(\Gamma_{R\times}\Psi_{R\times} - \Psi_{R\times}\Gamma_{R\times})h_l(X) - (\Gamma_{R\times}\Psi_x - \Psi_{R\times}\Gamma_x)\Big) \\ &= \frac{1}{2}\Big(\Gamma_{R\times}H_l(X)\Psi^\vee - \Psi_{R\times}H_l(X)\Gamma^\vee\Big) \end{aligned}$$

Combining these two terms together gives the result

$$\begin{aligned} \operatorname{Hess} h_l(X)(X\Gamma)(X\Psi) &= -\Psi_{R\times}H_l(X)\Gamma^\vee - \frac{1}{2}\Big(\Gamma_{R\times}H_l(X)\Psi^\vee - \Psi_{R\times}H_l(X)\Gamma^\vee\Big) \\ &= -\frac{1}{2}\Big(\Psi_{R\times}H_l(X)\Gamma^\vee + \Gamma_{R\times}H_l(X)\Psi^\vee\Big). \end{aligned}$$

□

Definition 3.6. *Consider the operator $A : \mathfrak{g} \to \mathfrak{g}$ defined in (3.14). The operator $\check{A} \in \mathbb{R}^{15\times 15}$ is the matrix representation of A, implicitly defined by the equation*

$$(A \circ \Gamma)^\vee = \check{A}\Gamma^\vee$$

for all $\Gamma \in \mathfrak{g}$.

Lemma 3.7. *The matrix representation $\check{A}$ from Definition 3.6 is given by*

$$\check{A} = -\begin{bmatrix} (u_\omega - \hat{\theta})_\times & \mathbf{0} & \mathbf{0} & \boldsymbol{I} & \mathbf{0} \\ \mathbf{0} & (u_\omega - \hat{\theta})_\times & -\boldsymbol{I} & \mathbf{0} & \mathbf{0} \\ (u_a - \hat{\phi})_\times & \mathbf{0} & (u_\omega - \hat{\theta})_\times & \mathbf{0} & \boldsymbol{I} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix}$$

Proof. Firstly, we consider the first term in A, applied to an arbitrary $\Gamma \in \mathfrak{g}$,

$$\begin{aligned} \mathbf{d}_1\lambda(\hat{X}, u) \circ T_e L_{\hat{X}} \circ \Gamma &= \mathbf{d}\begin{bmatrix} u_\omega - \hat{\theta} \\ \hat{R}^\top \hat{v} \\ u_a - \hat{\phi} - \hat{R}^\top g \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}^\wedge \circ \hat{X}\Gamma \\ &= \begin{bmatrix} -\Gamma_\theta \\ (\hat{R}\Gamma_{R\times})^\top \hat{v} + \hat{R}^\top(\hat{R}\Gamma_v) \\ -\Gamma_\phi - (\hat{R}\Gamma_{R\times})^\top g \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}^\wedge \\ &= \begin{bmatrix} -\Gamma_\theta \\ (\hat{R}^\top \hat{v})_\times \Gamma_R + \Gamma_v \\ -\Gamma_\phi - (\hat{R}^\top g)_\times \Gamma_R \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}^\wedge \\ &= \left(\begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & -\boldsymbol{I} & \mathbf{0} \\ (\hat{R}^\top \hat{v})_\times & \mathbf{0} & \boldsymbol{I} & \mathbf{0} & \mathbf{0} \\ -(\hat{R}^\top g)_\times & \mathbf{0} & \mathbf{0} & \mathbf{0} & -\boldsymbol{I} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \Gamma^\vee \right)^\wedge \end{aligned}$$

From Lemma 3.2, the matrix representation of the second term in A is given by

$$-\check{\mathrm{ad}}_{\lambda(\hat{X},u)} = -\begin{bmatrix} (u_\omega - \hat{\theta})_\times & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ (\hat{R}^\top \hat{v})_\times & (u_\omega - \hat{\theta})_\times & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ (u_a - \hat{\phi} - \hat{R}^\top g)_\times & \mathbf{0} & (u_\omega - \hat{\theta})_\times & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix}.$$

With the choice of the (0)-connection, the torsion tensor is zero, thus

$$T_{\lambda(\hat{X},u)} = 0.$$

Summing these three terms gives the result of the lemma. □

Definition 3.8. *Consider the operator* $E : \mathfrak{g} \to \mathfrak{g}^*$ *as defined in* (3.15). *The operator* $\check{E} \in \mathbb{R}^{15\times 15}$ *is the matrix representation of* E*, is implicitly defined by the equation*

$$(E \circ \Gamma) \circ \Psi = (\Psi^\vee)^\top \check{E} \Gamma^\vee$$

for all $\Gamma, \Psi \in \mathfrak{g}$.

Lemma 3.9. *The matrix representation* $\check{E}$ *from Definition* 3.8 *is given by*

$$\check{E} = \sum_{l \in \boldsymbol{L}} \mathbb{P}_s \Big(F\big(M_l(y_l - \hat{y}_l)\big)^\top H_l(\hat{X}) \Big) + H_l(\hat{X})^\top M_l H_l(\hat{X})$$

where $F : \mathbb{R}^3 \to \mathbb{R}^{3\times 15}$ *is defined by*

$$F(s) := \begin{bmatrix} s_\times & \mathbf{0}_{3\times 12} \end{bmatrix},$$

Proof. From (3.15), we have

$$E := -T_e L_{\hat{X}}^* \circ \sum_{l \in \boldsymbol{L}} \Big(\big(M_l \circ (y_l - \hat{y}_l)\big)^{T_{\hat{X}} G} \circ \operatorname{Hess} \hat{y}_l - (\mathbf{d}\hat{y}_l)^* \circ M_l \circ \mathbf{d}\hat{y}_l \Big) \circ T_e L_{\hat{X}},$$

which we split into two components

$$\begin{aligned}
E_1(l) &:= -T_e L_{\hat{X}}^* \circ \Big(\big(M_l \circ (y_l - \hat{y}_l)\big)^{T_{\hat{X}} G} \circ \operatorname{Hess} \hat{y}_l \Big) \circ T_e L_{\hat{X}} \\
E_2(l) &:= T_e L_{\hat{X}}^* \circ \Big((\mathbf{d}\hat{y}_l)^* \circ M_l \circ \mathbf{d}\hat{y}_l \Big) \circ T_e L_{\hat{X}} \\
E &= \sum_{l \in \boldsymbol{L}} (E_1(l) + E_2(l))
\end{aligned}$$

Applying the E_1 operator to two arbitrary elements $\Gamma, \Psi \in \mathfrak{g}$, we have

$$\begin{aligned}
(E_1 \circ \Gamma) \circ \Psi &= -\Big\langle T_e L_{\hat{X}}^* \circ \Big(\big(M_l \circ (y_l - \hat{y}_l)\big)^{T_{\hat{X}} G} \circ \operatorname{Hess} \hat{y}_l \Big) \circ T_e L_{\hat{X}} \circ \Gamma, \Psi \Big\rangle \\
&= -\Big\langle \Big(\big(M_l \circ (y_l - \hat{y}_l)\big)^{T_{\hat{X}} G} \circ \operatorname{Hess} \hat{y}_l \Big) \circ \hat{X}\Gamma, \hat{X}\Psi \Big\rangle
\end{aligned}$$

Using the identity from (A.5),

$$(E_1 \circ \Gamma) \circ \Psi = -(M_l \circ (y_l - \hat{y}_l)) \circ (\operatorname{Hess} \hat{y}_l \circ \hat{X}\Gamma) \circ \hat{X}\Psi$$

Substituting in the result from Lemma 3.5 gives

$$(E_1 \circ \Gamma) \circ \Psi = \frac{1}{2} (M_l(y_l - \hat{y}_l))^\top \left(\Psi_{R\times} H_l(\hat{X}) \Gamma^\vee + \Gamma_{R\times} H_l(\hat{X}) \Psi^\vee \right)$$

$$(E_1 \circ \Gamma) \circ \Psi = -\frac{1}{2}\Big(\Psi_R^\top \big(M_l(y_l - \hat{y}_l)\big)_\times H_l(\hat{X})\Gamma^\vee \\ + \Gamma_R^\top \big(M_l(y_l - \hat{y}_l)\big)_\times H_l(\hat{X})\Psi^\vee\Big)$$

$$(E_1 \circ \Gamma) \circ \Psi = \frac{1}{2}\Big((\Psi^\vee)^\top F\big(M_l(y_l - \hat{y}_l)\big)^\top H_l(\hat{X})\Gamma^\vee \\ + (\Gamma^\vee)^\top F\big(M_l(y_l - \hat{y}_l)\big)^\top H_l(\hat{X})\Psi^\vee\Big)$$

$$(E_1 \circ \Gamma) \circ \Psi = (\Psi^\vee)^\top \mathbb{P}_s \left(F\big(M_l(y_l - \hat{y}_l)\big)^\top H_l(\hat{X})\right) \Gamma^\vee.$$

Applying the E_2 operator to two arbitrary elements $\Gamma, \Psi \in \mathfrak{g}$, we have

$$\begin{aligned}(E_2 \circ \Gamma) \circ \Psi &= \left\langle T_e L_{\hat{X}}^* \circ \left((\mathbf{d}\hat{y}_l)^* \circ M_l \circ \mathbf{d}\hat{y}_l\right) \circ T_e L_{\hat{X}} \circ \Psi, \Gamma \right\rangle \\ &= \left\langle M_l \circ \mathbf{d}\hat{y}_l \circ \hat{X}\Psi, \mathbf{d}\hat{y}_l \circ \hat{X}\Gamma \right\rangle\end{aligned}$$

and then using the result from Lemma 3.4,

$$\begin{aligned}(E_2 \circ \Gamma) \circ \Psi &= \left\langle M_l H_l(\hat{X})\Psi^\vee, H_l(\hat{X})\Gamma^\vee \right\rangle \\ &= (\Psi^\vee)^\top H_l(\hat{X})^\top M_l H_l(\hat{X})\Gamma^\vee.\end{aligned}$$

The result then follows from combining E_1 and E_2. □

Definition 3.10. *Consider the operator $r \in \mathfrak{g}^*$, defined in (3.13). The operator $\check{r} \in \mathbb{R}^{15}$ is the vector (Reisz) representation of r, defined by the relationship*

$$r \circ \Gamma = \langle \check{r}, \Gamma^\vee \rangle = \check{r}^\top \Gamma^\vee$$

for all $\Gamma \in \mathfrak{g}$.

Lemma 3.11. *The operator $\check{r}$ from Definition 3.10 is given by*

$$\check{r} = \sum_{l \in \boldsymbol{L}} H_l(\hat{X})^\top M_l(y_l - \hat{y}_l)$$

Proof. Recall the definition of r from (3.13), and consider it as applied to an arbitrary element $\Gamma \in \mathfrak{g}$,

$$\begin{aligned}r \circ \Gamma &= \sum_{l \in \boldsymbol{L}} T_e L_{\hat{X}}^* \circ \left(M_l \circ (y_l - \hat{y}_l) \circ \mathbf{d}\hat{y}_l\right) \circ \Gamma \\ &= \sum_{l \in \boldsymbol{L}} M_l \circ (y_l - \hat{y}_l) \circ \mathbf{d}\hat{y}_l \circ \hat{X}\Gamma\end{aligned}$$

Recalling the derivative of the measurement function from Lemma 3.4, we have

$$r \circ \Gamma = \sum_{l \in \boldsymbol{L}} (y_l - \hat{y}_l)^\top M_l^\top H_l(\hat{X}) \Gamma^\vee$$

and the result follows. □

Definition 3.12. *Consider the gain operator, $K(t) : \mathfrak{g}^* \to \mathfrak{g}$, defined in (3.12). The matrix operator, $\check{K} \in \mathbb{R}^{15\times 15}$, is defined by*

$$K \circ \mu = (\check{K}\mu^\vee)^\wedge$$

for all $\mu \in \mathfrak{g}^*$

With this collection of definitions and lemmas, we now have the required components to describe the filter using an explicit matrix representation.

Theorem 3.13. *The filter equations (3.11) and (3.12) can be equivalently represented by the pair of matrix ordinary differential equations*

$$\dot{\hat{X}} = \hat{X}\left(\lambda^\vee(\hat{X}, u) + \check{K}\check{r}\right)^\wedge \tag{3.17}$$

$$\dot{\check{K}} = \check{A}\check{K} + \check{K}\check{A}^\top - \check{K}\check{E}\check{K} + \check{B}\check{W}^{-1}\check{B}^\top - \frac{1}{2}\check{\mathrm{ad}}_{Kr}\check{K} - \frac{1}{2}\check{K}(\check{\mathrm{ad}}_{Kr})^\top. \tag{3.18}$$

Proof. This follows from Definitions 3.1, 3.6, 3.8, 3.10, and 3.12. □

3.4 A Discrete-Time Filter Implementation

The filter proposed in Section 3.3 considers the system in continuous time, with all measurements available at all times. However, this is rarely the case in a real-world implementation. Digital sensors report sensor data at discrete points in time, rather than as a continuous-time signal. For example, modern IMU sensors are capable of sampling rates in the hundreds of Hz, and some into the kHz range. Other sensors, such as a camera-based system of detecting landmarks and other vehicles may only be capable of sampling rates in the order of 10 to 100 Hz. Additionally, while it is reasonable to assume that IMU measurements are uninterrupted, it is unlikely that all landmark measurements will be available at all sampling times. Several factors, including the range and field-of-view of the sensor, occlusion, and interference, mean that landmark measurements may only be available sporadically and with variable intervals between measurements.

A second issue with a continuous-time filter is that digital computers are not capable of representing or operating on arbitrary continuous-time signals. Thus, if we wish to implement the filter on a digital computer, and accept that sensors do not provide continuous-time signals, we need to derive a discrete-time implementation of the filter.

There is no universally 'correct' way to perform this discretisation, and there are different approaches, each with advantages and trade-offs. Our proposed method of discretisation relies on numerically integrating the terms in (3.17) and (3.18) which correspond to the external measurements separately to the IMU measurements and at different time intervals. If we firstly just consider the terms in (3.17) and (3.18) related to the IMU measurement, we have

$$\dot{\hat{X}} = \hat{X}\lambda(\hat{X}, u), \tag{3.19}$$
$$\dot{\check{K}} = \check{A}\check{K} + \check{K}\check{A}^{\top} + \check{B}\check{W}^{-1}\check{B}^{\top}. \tag{3.20}$$

We assume that the IMU has a fixed sampling rate of f_u Hz, with a corresponding sampling period of Δt_u seconds, and will use a sample-and-hold strategy. It is then straightforward to numerically integrate (3.19) and (3.20) forward in time from a time t to time $t + \Delta t_u$,

$$\begin{aligned} \hat{X}(t + \Delta t_u) &= \hat{X}(t) \cdot \exp\big(\Delta t_u \cdot \lambda(\hat{X}, u)\big), \\ \check{K}(t + \Delta t_u) &= \check{K}(t) + \Delta t_u \Big(\check{A}\check{K}(t) + \check{K}(t)\check{A}^{\top} + \check{B}\check{W}^{-1}\check{B}^{\top}\Big) \end{aligned} \tag{3.21}$$

where exp is the exponential map (refer to Appendix A.3.3).

We apply the same strategy to the external measurements but must make some additional considerations. Considering a single landmark measurement, y_l, the relevant terms in the state estimate ODEs, (3.17) and (3.18), are

$$\begin{aligned} \dot{\hat{X}} &= \hat{X}\left(\check{K}(t)\check{r}_l(\hat{X})\right)^{\wedge}, \\ \dot{\check{K}} &= -\check{K}\check{E}_l\check{K} - \mathbb{P}_s\left(\check{\mathrm{ad}}_{Kr_l}\check{K}\right). \end{aligned}$$

While the frequency of measurements may not be constant, we can still measure the period between subsequent landmark measurements, which we denote as Δt_l. A landmark measurement received at time t can be numerically integrated in a similar way as the IMU measurements;

$$\hat{X}(t^+) = \hat{X}(t) \cdot \mathrm{Exp}\left(\Delta t_l \cdot \check{K}(t^+)\check{r}_l(\hat{X})\right) \tag{3.22}$$
$$\check{K}(t^+) = \left(I + \Delta t_l\check{K}(t)\left(\check{E}_l + \mathbb{P}_s\left(\check{K}^{-1}(t)\check{\mathrm{ad}}_{Kr_l}\right)\right)\right)^{-1}\check{K}(t) \tag{3.23}$$

In practice, t must be a multiple of Δt_u, and so the precise time the landmark measurement is received is rounded up to the next multiple of Δt_u. Given that Δt_u is small and considering the velocities of vehicles within this time frame, this approximation introduces negligible error. Measurements of different landmarks which are received within the same time interval are processed sequentially.

To summarise using the standard *prediction* and *update* terminology common in the literature, the discrete-time minimum-energy filter is given by

Algorithm 1: Discrete Time Minimum Energy Filter for Inertial Localisation

Predict:

$$\hat{X}(t+\Delta t_u) = \hat{X}(t)\cdot \exp\left(\Delta t_u \cdot \lambda(\hat{X},u)\right) \tag{3.24a}$$
$$\check{K}(t+\Delta t_u) = \check{K}(t) + \Delta t_u\left(\check{A}\check{K}(t) + \check{K}(t)\check{A}^\top + \check{B}\check{W}^{-1}\check{B}^\top\right) \tag{3.24b}$$

Update: For a measurement y_l

$$\hat{X}(t^+) = \hat{X}(t)\cdot \mathrm{Exp}\left(\Delta t_l \cdot \check{K}(t^+)\check{r}_l(\hat{X})\right) \tag{3.25a}$$
$$\check{K}(t^+) = \left(I + \Delta t_l \check{K}(t)\left(\check{E}_l + \mathbb{P}_s\left(\check{K}^{-1}(t)\check{\mathrm{ad}}_{Kr_l}\right)\right)\right)^{-1}\check{K}(t) \tag{3.25b}$$

where, using Lemmas 3.4, 3.7, 3.9, and 3.11, we have

$$\lambda(X,u) = \begin{bmatrix} u_\omega - \theta \\ R^\top v \\ u_a - \phi - R^\top g \\ 0 \\ 0 \end{bmatrix}^\wedge \tag{3.26a}$$

$$\check{A} = -\begin{bmatrix} (u_\omega - \hat{\theta})_\times & \mathbf{0} & \mathbf{0} & \boldsymbol{I} & \mathbf{0} \\ \mathbf{0} & (u_\omega - \hat{\theta})_\times & -\boldsymbol{I} & \mathbf{0} & \mathbf{0} \\ (u_a - \hat{\phi})_\times & \mathbf{0} & (u_\omega - \hat{\theta})_\times & \mathbf{0} & \boldsymbol{I} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \tag{3.26b}$$

$$\check{r}_l = H_l(\hat{X})^\top M_l(y_l - \hat{y}_l) \tag{3.26c}$$
$$\check{E}_l = \mathbb{P}_s\left(F(M_l(y_l - \hat{y}_l))^\top H_l(\hat{X})\right) + H_l(\hat{X})^\top M_l H_l(\hat{X}) \tag{3.26d}$$
$$H_l(X) = \begin{bmatrix} h_l(X)_\times & -\boldsymbol{I}_3 & \mathbf{0}_{3\times 9}\end{bmatrix} \tag{3.26e}$$
$$F(s) = \begin{bmatrix} s_\times & \mathbf{0}_{3\times 12}\end{bmatrix} \tag{3.26f}$$
$$M_l = (D^{-1})^\top Q_l D^{-1}. \tag{3.26g}$$

3.4.1 Tuning

A practical consideration to make when implementing any filter is the choice of different tuning parameters. For the minimum-energy filter presented in this paper, there is a free choice in B, D, W, Q, and J_0. The cost in the initial estimate, J_0, is typically implicitly defined by initialising the filter with a priori information about the initial state of the system. When choosing the remaining parameters, it is necessary to have an understanding of the behaviour of the physical sensors in the system. A convenient choice for B and D is the square root of the covariance of the sensor measurement error, which can be measured and calibrated a priori.

In choosing the sensor covariance, we must also account for the difference between the continuous time model used in the cost functional, and the discrete-time implementation of the filter equations. The sample-and-hold strategy means that the measurement signals are correlated almost perfectly for the length of the hold time. This poses an issue when the sample rate of the IMU sensor is different to that of the external measurements. Consequently, we set $W = \frac{1}{\Delta t_u}\boldsymbol{I}$ and $Q_l = \frac{1}{\Delta t_l}\boldsymbol{I}$ to appropriately weight the terms in the cost functional and correct for the differences in sample rates.

As with any filter, the suggested values above provide an initial starting point for tuning. It may be necessary to adjust these values, particularly B, to improve filter stability and robustness. In Chapter 5, we discuss the specific values of these parameters used in the simulations.

3.4.2 Summary

In this chapter, we have shown how a minimum energy filter for single-vehicle inertial localisation can be derived from the abstract generalised formulation of Saccon, Trumpf *et al.* [77]. The kinematic and measurement models provide a realistic problem formulation, which mirrors that of an aerial vehicle equipped with an inertial measurement unit. This is key for being able to demonstrate the effectiveness of the algorithm on physical platforms or using data captured from real-world experiments. We have also described the challenges of transforming the continuous-time filter equations into discrete time and presented one possible version of the discrete-time minimum energy filter. We will perform some evaluations of the proposed filter using both simulation data in Chapter 5 and real-world data in Chapter 7. However, we note that the derivation of this filter is not the end goal of this thesis. In the next chapter, we will use the foundations of the single-vehicle filter to extend into a multi-vehicle scenario and develop a minimum energy filter for collaborative localisation.

Chapter 4

Collaborative Localisation using a Minimum Energy Filter

In this chapter, we shift our thinking from the localisation of a single vehicle to the collaborative localisation of a network of vehicles. The main challenge that this presents is in accurately modelling the interactions between different vehicles within the network, specifically computation, storage and communication. As discussed in the literature review, the double-counting problem is a key element of this problem, and any collaborative localisation algorithm must address this point in one way or another. We will focus on top-down collaborative localisation approaches (see 2.4.2), which aim to replicate or approximate a centralised filter, which effectively solves the double counting problem by tracking (or approximating) the entire covariance matrix.

The contributions of this chapter can be divided into two major parts. In Sections 4.1 and 4.2 we present a methodical step-by-step derivation of the state-of-the-art top-down collaborative filters to a level of detail not previously elucidated in the literature. Then, in Section 4.3, we apply the methodology and processes developed in the previous sections to develop a distributed minimum energy filter for the collaborative localisation of airborne vehicles.

To begin, in Section 4.1, we introduce the concept of a partitioned filter, where the state vector is divided into multiple components. We then incorporate new definitions of state- and measurement-independence which impose constraints on the structure of the filter. Using the standard extended Kalman filter (EKF) formulation as a case study, we show how these properties allow for the simplification of the EKF filter equations. Following this, we introduce the Schmidt-Kalman filter as a means to perform a partial update of the state vector.

The analysis of partitioned filters leads naturally to distributed filtering in Section 4.2, where partitions represent the individual nodes in the network. Building on the foundations from Section 4.1, we describe the distributed EKF of Roumeliotis and Bekey [79]. To make the distribution process more clear, we graphically illustrate where information is stored, where calculations are performed, and what information is transmitted between nodes.

By combining the distributed EKF from Roumeliotis and Bekey [79], and the concept of the partial-update Schmidt-Kalman filter, we introduce the distributed Schmidt-Kalman filter. This provides the often-overlooked intermediate step in the derivation of the work of Luft, Schubert *et al.* [98], but is worth examining in its own right. We describe and graphically illustrate the changes in information storage, communication and computation that are performed in the distributed SKF compared to the distributed EKF. From this, we then introduce the distributed approximate Schmidt Kalman filter proposed by Luft, Schubert *et al.* [98], which forms the final link in the long chain of filter derivations.

In the quest for improved collaborative localisation performance for challenging non-linear systems such as airborne vehicles, we revisit the minimum-energy single-vehicle localisation algorithm from Chapter 3. Given the literature in this area and the fact that the preceding analysis in this chapter is built around the extended Kalman filter, in Section 4.3, we question whether it is possible to construct a similar top-down collaborative localisation algorithm using a minimum-energy filter. This involves deriving a new multi-vehicle centralised minimum energy filter, incorporating both landmark measurement and relative inter-vehicle measurements, and demonstrating how the decomposition can be performed to create a distributed implementation. Unifying this new distributed filter with the analysis from Section 4.2, we also derive minimum-energy equivalents of the distributed Schmidt Kalman filter as well as the distributed approximate Schmidt Kalman filter. In total, we present a set of four novel minimum energy filters for collaborative localisation, each with different trade-offs between performance and network communication requirements. In subsequent chapters, we evaluate the performance of each of these filters on simulated data, and later on real-world experimental data.

4.1 Centralised State Estimation with Partitioned Filters

In order to develop a complete understanding of top-down collaborative localisation algorithms such as those proposed by Roumeliotis and Bekey [79] and Luft, Schubert *et al.* [98], we must first analyse the structure of centralised filters and the concept of a partitioned filter.

We start by considering the classical discrete-time extended Kalman filter (EKF), such as that described in Anderson and Moore [37]. The system state $x \in \mathbb{R}^n$ is modelled by the difference equation

$$x_{k+1} = f(x_k, u_k) + w_k \tag{4.1}$$

where $u \in \mathbb{R}^m$ is the control input, $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ is the state transition model, $w \in \mathbb{R}^n$ is the model noise, and $k \in \mathbb{N}$ is the time-step. The measurement $z \in \mathbb{R}^p$ is modelled by

$$z_k = h(x_k) + v_k \tag{4.2}$$

where $h : \mathbb{R}^n \to \mathbb{R}^p$ is the measurement model and $v \in \mathbb{R}^p$ is the measurement noise. The noise parameters w and v are assumed to be independent white, Gaussian, zero-mean stochastic processes with covariance Q and R respectively.

The discrete-time extended Kalman filter is given by

Algorithm 2: EKF: Discrete Time Extended Kalman Filter

Predict:

$$\hat{x}_{k|k-1} = f(\hat{x}_{k-1|k-1}, u_k) \tag{4.3a}$$
$$P_{k|k-1} = F_k P_{k-1|k-1} F_k^\top + Q_k \tag{4.3b}$$

Update:

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k(z_k - h(\hat{x}_{k|k-1})) \tag{4.4a}$$
$$P_{k|k} = (I - K_k H_k) P_{k|k-1} \tag{4.4b}$$

where

$$\begin{aligned} K_k &= P_{k|k-1} H_k^\top S_k^{-1} \\ S_k &= H_k P_{k|k-1} H_k^\top + R_k \\ F_k &= \frac{\partial}{\partial x} f(\hat{x}_{k-1|k-1}, u_k) \\ H_k &= \frac{\partial}{\partial x} h(\hat{x}_{k|k-1}) \end{aligned}$$

In subsequent equations, we make use of the shorthand notation $P_- = P_{k|k-1}$, $\hat{x}_- = \hat{x}_{k|k-1}$, $P_+ = P_{k|k}$, $\hat{x}_+ = \hat{x}_{k|k}$ when the time-step k is clear from context.[1]

4.1.1 Partitioned EKF

Consider an arbitrary partitioning of the elements of the state vector x, into two components, x^α and x^β, such that[2]

$$x = \begin{bmatrix} x^\alpha \\ x^\beta \end{bmatrix}.$$

[1] This shorthand may be convenient, but it lacks expressiveness and may lead to ambiguity. For example, in the state prediction step (4.3a), the value of $\hat{x}_+$ from the previous time-step is used to calculate $\hat{x}_-$ of the current time-step, but the time indices are not indicated. This can often lead to subtle errors when performing consecutive prediction and update steps. Thus, we make use of the shorthand notation only when there is no ambiguity and the value of k is clear from the context.

[2] Note that the ordering of elements within the state vector is arbitrary, and thus elements within α and β need not be consecutive.

The corresponding partitions of the covariance matrix are then

$$P = \begin{bmatrix} P^\alpha & P^{\alpha\beta} \\ P^{\beta\alpha} & P^\beta \end{bmatrix}.$$

The state transition model, f, is consequently decomposed into two separate functions, f^α and f^β, such that

$$f(\hat{x}, u) = \begin{bmatrix} f^\alpha(\hat{x}^\alpha, \hat{x}^\beta, u) \\ f^\beta(\hat{x}^\alpha, \hat{x}^\beta, u) \end{bmatrix}$$

while the linearisation, F, becomes

$$F = \begin{bmatrix} F^\alpha & F^{\alpha\beta} \\ F^{\beta\alpha} & F^\beta \end{bmatrix} = \begin{bmatrix} \frac{\partial}{\partial x^\alpha} f^\alpha(\hat{x}^\alpha, \hat{x}^\beta, u) & \frac{\partial}{\partial x^\beta} f^\alpha(\hat{x}^\alpha, \hat{x}^\beta, u) \\ \frac{\partial}{\partial x^\alpha} f^\beta(\hat{x}^\alpha, \hat{x}^\beta, u) & \frac{\partial}{\partial x^\beta} f^\beta(\hat{x}^\alpha, \hat{x}^\beta, u) \end{bmatrix} \tag{4.5}$$

For the update step, the measurement model, h, is overloaded such that

$$h(\hat{x}) = h(\hat{x}^\alpha, \hat{x}^\beta)$$

and the partitioned linearisation of the measurement model becomes

$$H = \begin{bmatrix} H^\alpha & H^\beta \end{bmatrix}.$$

Substituting these new definitions into the EKF filter equations (4.3) and (4.4), gives

Algorithm 3: Partitioned Extended Kalman Filter

Predict:

$$\hat{x}^\alpha_- = f^\alpha(\hat{x}^\alpha_{k-1|k-1}, \hat{x}^\beta_{k-1|k-1}, u_k) \tag{4.6a}$$

$$\hat{x}^\beta_- = f^\beta(\hat{x}^\alpha_{k-1|k-1}, \hat{x}^\beta_{k-1|k-1}, u_k) \tag{4.6b}$$

$$P^\alpha_- = (F^\alpha P^\alpha + F^{\alpha\beta} P^{\beta\alpha})(F^\alpha)^\top + (F^\alpha P^{\alpha\beta} + F^{\alpha\beta} P^\beta)(F^{\alpha\beta})^\top + Q^\alpha \tag{4.6c}$$

$$P^{\alpha\beta}_- = (F^\alpha P^\alpha + F^{\alpha\beta} P^{\beta\alpha})(F^{\beta\alpha})^\top + (F^\alpha P^{\alpha\beta} + F^{\alpha\beta} P^\beta)(F^\beta)^\top + Q^{\alpha\beta} \tag{4.6d}$$

$$P^{\beta\alpha}_- = (P^{\alpha\beta}_-)^\top \tag{4.6e}$$

$$P^\beta_- = (F^{\beta\alpha} P^\alpha + F^\beta P^{\beta\alpha})(F^{\beta\alpha})^\top + (F^{\beta\alpha} P^{\alpha\beta} + F^\beta P^\beta)(F^\beta)^\top + Q^\beta \tag{4.6f}$$

Update:

$$\hat{x}^{\alpha}_{+} = \hat{x}^{\alpha}_{-} + K^{\alpha}(z_k - h(\hat{x}^{\alpha}_{-}, \hat{x}^{\beta}_{-})) \tag{4.7a}$$
$$\hat{x}^{\beta}_{+} = \hat{x}^{\beta}_{-} + K^{\beta}(z_k - h(\hat{x}^{\alpha}_{-}, \hat{x}^{\beta}_{-})) \tag{4.7b}$$
$$P^{\alpha}_{+} = (I - K^{\alpha}H^{\alpha})P^{\alpha}_{-} - K^{\alpha}H^{\beta}P^{\beta\alpha}_{-} \tag{4.7c}$$
$$P^{\alpha\beta}_{+} = (I - K^{\alpha}H^{\alpha})P^{\alpha\beta}_{-} - K^{\alpha}H^{\beta}P^{\beta}_{-} \tag{4.7d}$$
$$P^{\beta\alpha}_{+} = (P^{\alpha\beta}_{+})^{\top} \tag{4.7e}$$
$$P^{\beta}_{+} = (I - K^{\beta}H^{\beta})P^{\beta}_{-} - K^{\beta}H^{\alpha}P^{\alpha\beta}_{-} \tag{4.7f}$$

where

$$K^{\alpha} = (P^{\alpha}_{-}H^{\alpha\top} + P^{\alpha\beta}H^{\beta\top})S^{-1}$$
$$K^{\beta} = (P^{\beta}_{-}H^{\beta\top} + P^{\beta\alpha}H^{\alpha\top})S^{-1}$$
$$S = H^{\alpha}P^{\alpha}_{-}H^{\alpha\top} + H^{\alpha}P^{\alpha\beta}_{-}H^{\beta\top} + H^{\beta}P^{\beta\alpha}_{-}H^{\alpha\top} + H^{\beta}P^{\beta}_{-}H^{\beta\top} + R.$$

Note that these update equations provide exactly the same result as (4.3) and (4.4), but are just expressed in terms of the individual partitions.

Independent State Transition Model

In some cases, the partitioning of the state into x^{α} and x^{β} allows the state transition models, f^{α} and f^{β}, to be mutually independent of each other. By this, we mean that we can partition the control input

$$u = \begin{bmatrix} u^{\alpha} \\ u^{\beta} \end{bmatrix}$$

such that f^{α} is a function of neither x^{β} nor u^{β} and f^{β} is a function of neither x^{α} nor u^{α}. In this case, we have

$$f(x,u) = \begin{bmatrix} f^{\alpha}(x^{\alpha}, u^{\alpha}) \\ f^{\beta}(x^{\beta}, u^{\beta}) \end{bmatrix}$$

and the linearisation of the state transition model simplifies to

$$F = \begin{bmatrix} F^{\alpha} & 0 \\ 0 & F^{\beta} \end{bmatrix}.$$

If we include the additional assumption that $Q = \begin{bmatrix} Q^{\alpha} & 0 \\ 0 & Q^{\beta} \end{bmatrix}$ then we can simplify the partitioned EKF prediction step (4.6) to

$$\hat{x}^{\alpha}_{-} = f^{\alpha}(\hat{x}^{\alpha}_{k-1|k-1}, u^{\alpha}_{k})$$

$$
\begin{aligned}
\hat{x}^{\beta}_{-} &= f^{\beta}(\hat{x}^{\beta}_{k-1|k-1}, u^{\beta}_{k}) \\
P^{\alpha}_{-} &= F^{\alpha} P^{\alpha}_{k-1|k-1} {F^{\alpha}}^{\top} + Q^{\alpha} \\
P^{\alpha\beta}_{-} &= F^{\alpha} P^{\alpha\beta}_{k-1|k-1} {F^{\beta}}^{\top} \\
P^{\beta\alpha}_{-} &= (P^{\alpha\beta}_{-})^{\top} \\
P^{\beta}_{-} &= F^{\beta} P^{\beta}_{k-1|k-1} {F^{\beta}}^{\top} + Q^{\beta}
\end{aligned}
$$

In practice, such a partition might exist when a filter is estimating the state of two separate sub-systems. If the sub-systems do not interact in the state-transition phase and do not share a common control input, then the state-transition models will be mutually independent.

Independent Measurement Models

If a measurement model, h, is not dependent on all elements of the state, it is possible to choose a (not necessarily unique) partition such that $H^{\beta} = 0$. We will call a measurement model with this property "β-independent".

As an example, a filter estimating a robot's pose (position and orientation) may have a GNSS measurement which just measures the robot's position. Thus, the state could be partitioned where α represents the position state variables and β represents the orientation state variables. The measurement model would then be orientation-independent, or β-independent (for this particular choice of β).

This property helps to simplify the partitioned EKF update equations by eliminating all terms containing H^{β}. If a measurement model is β-independent, The update step (4.7) then simplifies to

$$
\begin{aligned}
\hat{x}^{\alpha}_{+} &= \hat{x}^{\alpha}_{-} + K^{\alpha}(z_k - h(\hat{x}^{\alpha}_{-})) \\
\hat{x}^{\beta}_{+} &= \hat{x}^{\beta}_{-} + K^{\beta}(z_k - h(\hat{x}^{\alpha}_{-})) \\
P^{\alpha}_{+} &= (I - K^{\alpha} H^{\alpha}) P^{\alpha}_{-} \\
P^{\alpha\beta}_{+} &= (I - K^{\alpha} H^{\alpha}) P^{\alpha\beta}_{-} \\
P^{\beta\alpha}_{+} &= (P^{\alpha\beta}_{+})^{\top} \\
P^{\beta}_{+} &= P^{\beta}_{-} - K^{\beta} H^{\alpha} P^{\alpha\beta}_{-}
\end{aligned}
$$

where

$$
\begin{aligned}
K^{\alpha} &= P^{\alpha}_{-} {H^{\alpha}}^{\top} S^{-1} \\
K^{\beta} &= P^{\beta\alpha}_{-} {H^{\alpha}}^{\top} S^{-1} \\
S &= H^{\alpha} P^{\alpha}_{-} {H^{\alpha}}^{\top} + R
\end{aligned}
$$

If the partitioned system has both an independent state transition model and a β-independent measurement model, then the partitioned EKF can be expressed as

Algorithm 4: Partitioned EKF with independent state transition and β-independent measurement model

Predict:

$$\hat{x}^{\alpha}_{-} = f^{\alpha}(\hat{x}^{\alpha}_{k-1|k-1}, u^{\alpha}_{k}) \tag{4.8a}$$
$$\hat{x}^{\beta}_{-} = f^{\beta}(\hat{x}^{\beta}_{k-1|k-1}, u^{\beta}_{k}) \tag{4.8b}$$
$$P^{\alpha}_{-} = F^{\alpha} P^{\alpha}_{k-1|k-1} F^{\alpha\top} + Q^{\alpha} \tag{4.8c}$$
$$P^{\alpha\beta}_{-} = F^{\alpha} P^{\alpha\beta}_{k-1|k-1} F^{\beta\top} \tag{4.8d}$$
$$P^{\beta\alpha}_{-} = (P^{\alpha\beta}_{-})^{\top} \tag{4.8e}$$
$$P^{\beta}_{-} = F^{\beta} P^{\beta}_{k-1|k-1} F^{\beta\top} + Q^{\beta} \tag{4.8f}$$

Update:

$$\hat{x}^{\alpha}_{+} = \hat{x}^{\alpha}_{-} + K^{\alpha}(z_k - h(\hat{x}^{\alpha}_{-})) \tag{4.9a}$$
$$\hat{x}^{\beta}_{+} = \hat{x}^{\beta}_{-} + K^{\beta}(z_k - h(\hat{x}^{\alpha}_{-})) \tag{4.9b}$$
$$P^{\alpha}_{+} = (I - K^{\alpha} H^{\alpha}) P^{\alpha}_{-} \tag{4.9c}$$
$$P^{\alpha\beta}_{+} = (I - K^{\alpha} H^{\alpha}) P^{\alpha\beta}_{-} \tag{4.9d}$$
$$P^{\beta}_{+} = P^{\beta}_{-} - K^{\beta} H^{\alpha} P^{\alpha\beta}_{-} \tag{4.9e}$$

An important consideration that has not been made yet is that the partition of the state vector may change arbitrarily over time. To illustrate this point, consider the prediction and update step for the term $P^{\alpha\beta}$. If the initial value of $P^{\alpha\beta}$ is zero, then it will remain zero regardless of any measurements or control inputs made. This would also mean that any measurements made would not alter the value of $\hat{x}^{\beta}$, as the two partitions are essentially acting as completely independent systems. However, this is only when considering a fixed partition and fixed measurement model. If the measurement model, h, changes so that it is no longer β-independent for the chosen partition, then a new partition can be made, for example,

$$x = \begin{bmatrix} x^{\alpha'} \\ x^{\beta'} \end{bmatrix}$$

such that h is β'-independent. This changes and reorders the components of the covariance matrices, and thus can introduce cross covariance in $P^{\alpha'\beta'}$.

Again, the emphasis is on the fact that this filter is still equivalent to the standard EKF, just with a different representation.

4.1.2 Schmidt-Kalman Filter

The Schmidt-Kalman filter (SKF), a variation on the Kalman filter, is often known in the literature as the 'consider Kalman filter'. This name comes from the way the SKF is commonly used to account for (or *consider*) uncertainties in fixed parameters in the system model that are not part of the state vector. Rather than *consider* external parameters, an alternative interpretation of the SKF is as a *partial update* filter, in which some elements of the state vector are not modified during the update step of the filter. We will show how this works with the help of the partitioned filter concept introduced earlier.

Brink [99] provides a comprehensive analysis of the partial update SKF and also shows that one can alternate between using the EKF and SKF update equations and maintain a conservative estimate of the covariance.[3]

The formulation of the SKF is similar to that of the partitioned EKF — the prediction step remains the same while the update step is different. In the SKF update step, a subset of the state variables is selected to not be updated with the measurement information. The selection of which state variables are not updated is application-specific. We can illustrate this by partitioning the state, x, into two subsets, x^α and x^β, where we will choose not to update the state variables in x^β during the update step.

We modify the partitioned EKF update equations (4.7) to give the SKF update

$$\hat{x}^\alpha_+ = \hat{x}^\alpha_- + K^\alpha(z_k - h(\hat{x}^\alpha_-, \hat{x}^\beta_-)) \tag{4.10a}$$

$$\hat{x}^\beta_+ = \hat{x}^\beta_- \tag{4.10b}$$

$$P^\alpha_+ = (I - K^\alpha H^\alpha)P^\alpha_- - K^\alpha H^\beta P^{\beta\alpha}_- \tag{4.10c}$$

$$P^{\alpha\beta}_+ = (I - K^\alpha H^\alpha)P^{\alpha\beta}_- - K^\alpha H^\beta P^\beta_- \tag{4.10d}$$

$$P^{\beta\alpha}_+ = (P^{\alpha\beta}_+)^\top \tag{4.10e}$$

$$P^\beta_+ = P^\beta_- \tag{4.10f}$$

where

$$K^\alpha = (P^\alpha_- H^{\alpha^\top} + P^{\alpha\beta} H^{\beta^\top})S^{-1}$$
$$S = H^\alpha P^\alpha_- H^{\alpha^\top} + H^\alpha P^{\alpha\beta}_- H^{\beta^\top} + H^\beta P^{\beta\alpha}_- H^{\alpha^\top} + H^\beta P^\beta_- H^{\beta^\top} + R$$

Note that the only difference to the partitioned EKF is in (4.10b) and (4.10f). If the partition of the state into x^α and x^β also results in a β-independent measurement model, then the update step of the SKF simplifies similarly to (4.9) which gives

[3] Refer to Appendix B for definitions of consistency and conservative estimates.

Algorithm 5: Schmidt-Kalman filter with β-independent measurement model

Predict: (Identical to partitioned EKF (4.6))

$$\hat{x}^{\alpha}_{-} = f^{\alpha}(\hat{x}^{\alpha}_{k-1|k-1}, \hat{x}^{\beta}_{k-1|k-1}, u_k) \tag{4.11a}$$
$$\hat{x}^{\beta}_{-} = f^{\beta}(\hat{x}^{\alpha}_{k-1|k-1}, \hat{x}^{\beta}_{k-1|k-1}, u_k) \tag{4.11b}$$
$$P^{\alpha}_{-} = (F^{\alpha}P^{\alpha} + F^{\alpha\beta}P^{\beta\alpha})(F^{\alpha})^{\top} + (F^{\alpha}P^{\alpha\beta} + F^{\alpha\beta}P^{\beta})(F^{\alpha\beta})^{\top} + Q^{\alpha} \tag{4.11c}$$
$$P^{\alpha\beta}_{-} = (F^{\alpha}P^{\alpha} + F^{\alpha\beta}P^{\beta\alpha})(F^{\beta\alpha})^{\top} + (F^{\alpha}P^{\alpha\beta} + F^{\alpha\beta}P^{\beta})(F^{\beta})^{\top} + Q^{\alpha\beta} \tag{4.11d}$$
$$P^{\beta\alpha}_{-} = (P^{\alpha\beta}_{-})^{\top} \tag{4.11e}$$
$$P^{\beta}_{-} = (F^{\beta\alpha}P^{\alpha} + F^{\beta}P^{\beta\alpha})(F^{\beta\alpha})^{\top} + (F^{\beta\alpha}P^{\alpha\beta} + F^{\beta}P^{\beta})(F^{\beta})^{\top} + Q^{\beta} \tag{4.11f}$$

Update:

$$\hat{x}^{\alpha}_{+} = \hat{x}^{\alpha}_{-} + K^{\alpha}(z_k - h(\hat{x}^{\alpha}_{-})) \tag{4.12a}$$
$$\hat{x}^{\beta}_{+} = \hat{x}^{\beta}_{-} \tag{4.12b}$$
$$P^{\alpha}_{+} = (I - K^{\alpha}H^{\alpha})P^{\alpha}_{-} \tag{4.12c}$$
$$P^{\alpha\beta}_{+} = (I - K^{\alpha}H^{\alpha})P^{\alpha\beta}_{-} \tag{4.12d}$$
$$P^{\beta\alpha}_{+} = (P^{\alpha\beta}_{+})^{\top} \tag{4.12e}$$
$$P^{\beta}_{+} = P^{\beta}_{-} \tag{4.12f}$$

where

$$K^{\alpha} = P^{\alpha}_{-}{H^{\alpha}}^{\top}S^{-1}$$
$$S = H^{\alpha}P^{\alpha}_{-}{H^{\alpha}}^{\top} + R$$

With the partitioned EKF and the SKF described in the context of a centralised system, we can now explore how these same algorithms are applied in the context of networked systems.

4.2 Networked Filters

A networked system is one where components are modelled as separate entities and information is exchanged between these entities through a communication network. These components can include sensors, actuators, and computers, while the communication links may be physical wires, RF transceivers, optical, or any other means of communication. While almost all physical systems can be modelled as networked systems, the network model approach is typically only used when the communication links between components are required to be explicitly modelled. Networked systems are commonly represented as graphs, where nodes represent the components of the system and edges represent communication channels between nodes.

When applying a filter to a networked system, it is important to consider where state information is stored, where computations are performed and where sensor information is received. For example, if sensor information is received at a different node from where computation occurs, then information must be communicated between nodes to enable the filter equations to be computed.

Consider a network of nodes, $N = \{1, \ldots, n\}$, where each individual node, i, has system state, x^i, control input u^i, and makes some measurement z^i of the system state. In the context of collaborative localisation, we may think of each vehicle as being a node, with its own control inputs and onboard sensors. If we concatenate all the individual state vectors and control inputs, we can see that the overall networked system is an example of the same system described in (4.1) and (4.2) from Section 4.1. One way to perform the filtering task is to collect all the different measurement parts at a central processing node and perform the EKF prediction and update steps on this central node for the state of the entire system.

A graphical representation of the information flows between nodes is shown in Figures 4.1 and 4.2 on Page 64. We can observe that the central-processing system has several drawbacks. Firstly, it requires all nodes to maintain a connection to the central processor at all times. As all the state information is stored at the central node, any disruption to communication will cause a node to cease operating. In the case of collaborative localisation, an aerial vehicle would quickly crash without an up-to-date state estimate.

The central processing node also introduces a single point of failure. Any disruption to the central node will cause the entire network to fail, which negates many of the benefits achieved by adopting a multi-vehicle system approach.

Finally, this also places a large burden on the communications network. Every single control input and measurement must be transmitted over the network to the central processor. In some instances, this is simply too much data to be practical or reliable, especially as the number of nodes in the network increases to large numbers.

Thankfully, there has been a large body of research on distributed state estimation, including distributed collaborative localisation. We will first explore the algorithm proposed by Roumeliotis and Bekey [79], which provides a way of mitigating many of these issues.

4.2.1 Distributed EKF

In the case of the collaborative localisation problem, and many other cases of network state estimation, the state evolution models for each node will be independent (as in the definition in Section 4.1.1). For collaborative localisation, it is relatively easy to demonstrate this fact as, for example, the kinematic model for each vehicle only depends on its own state and control input.

In this case, Roumeliotis and Bekey [79] showed that the calculations for the prediction step of the EKF can be distributed among the nodes of the network. To enable this, the partitions of the state estimate, $\hat{x}$, and covariance matrix, P, are each stored on the corresponding nodes. Considering an arbitrary node $i \in N$, the information stored on node i is

$$\hat{x}^i, \qquad P^i, \qquad p^{ij} \quad \forall j \in 1 \dots n \setminus i \tag{4.13}$$

where p^{ij} is chosen such that $P^{ij} = p^{ij} {p^{ji}}^\top$.

The filter can then be described by

Algorithm 6: Distributed EKF

Predict:
On each node $i \in N$:

$$\hat{x}^i_- = f^i(\hat{x}^i_{k-1|k-1}, u^i) \tag{4.14a}$$

$$P^i_- = F^i P^i_{k-1|k-1} {F^i}^\top + Q^i \tag{4.14b}$$

$$p^{ij}_{k|k-1} = F^i p^{ij}_{k-1|k-1} \quad \forall j \in 1 \dots n \setminus i \tag{4.14c}$$

Update: (Identical to EKF (4.4))

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k(z_k - h(\hat{x}_{k|k-1})) \tag{4.15a}$$

$$P_{k|k} = (I - K_k H_k) P_{k|k-1} \tag{4.15b}$$

A graphical representation of the information flow for the prediction step of the distributed EKF is shown in Figure 4.3 on Page 65. We can observe that no communication between the nodes is required to perform the prediction step. The key development that enables this is the observation that p^{ij} is updated by pre-multiplication only. The actual value of P^{ij} is not required to be known during the prediction step, and it can be reconstructed during the update step by sharing information between the nodes i and j and then computing $P^{ij}_- = p^{ij}_{k|k-1} {p^{ji}_{k|k-1}}^\top$.

If no measurements, z, are made at time k, the estimates are simply propagated forward in time by

$$\hat{x}^i_+ = \hat{x}^i_- \tag{4.16a}$$

$$P^i_+ = P^i_- \tag{4.16b}$$

$$p^{ij}_{k|k} = p^{ij}_{k|k-1} \tag{4.16c}$$

This enables each node to predict forward in time without having to communicate with other nodes until the point that the next measurement, z, is made.

The update step is identical to the standard EKF (4.4), however, the nodes must first communicate to recover the cross-covariance terms, P_-^{ij}. In Figure 4.4 on Page 65, we show an example of a 4-node network where a measurement z_k has been received by Node 1. In Phase 1, in response to a request from Node 1, all other nodes in the network transmit the required terms to Node 1. Node 1 then incorporates its own information to reconstruct all the cross-covariance terms. Then, in Phase 2, Node 1 performs the update step as described in (4.4) and then transmits the relevant terms to the remaining nodes. Upon receiving the new information each node updates its state estimate and covariance.

The example described above, and shown in Figure 4.4, is only one example of how the update step could be performed, where the majority of the computation is performed on Node 1. The computation could be distributed differently across the nodes, which would result in a different communication strategy. However, in all cases, the communication graph is connected, i.e. information is shared between all nodes in the network.

This method of filter distribution solves a number of the previously mentioned issues with the central EKF. There is no longer a single point of failure, and each node stores a local copy of its own state. A complete disruption to the network means that each node will locally propagate its estimate without any measurement updates, resulting in degraded performance, but not complete failure. As discussed above, this technique also reduces the communication burden on the network, only requiring the transmission of information when measurements are made. If measurements are low in frequency compared to control inputs, for example, an 8 kHz IMU with a 25 Hz vision system, this can have a significant impact on network bandwidth.

However, this method does not solve every problem. One issue is that, during the update step, all nodes need to be fully connected and a large amount of information is transmitted across the network. This remains impractical for many situations, especially where network performance can not be guaranteed. Luft, Schubert *et al.* [98] proposes one way in which these problems can be addressed, but in order to analyse this method, we must first introduce the distributed Schmidt-Kalman filter.

4.2.2 Distributed Schmidt-Kalman Filter

As we have seen in Section 4.1.2, the SKF has a very similar structure to the EKF. Thus, it is no surprise that the SKF can also be distributed using a similar technique as shown for the EKF in Section 4.2.1. To construct the distributed SKF, we use the same decoupling strategy for the state estimate and covariance matrix that is used for the distributed EKF, meaning that the information stored on each node is the same as in (4.13). The prediction step is performed in the same way as the distributed EKF (4.14) and the update step is performed using the SKF update equations (4.12), assuming that the measurement model is β-independent.

Algorithm 7: Distributed SKF

Predict: (Identical to Distributed EKF (4.14))

$$\hat{x}^i_- = f^i(\hat{x}^i_{k-1|k-1}, u^i) \tag{4.17a}$$

$$P^i_- = F^i P^i_{k-1|k-1} {F^i}^\top + Q^i \tag{4.17b}$$

$$p^{ij}_{k|k-1} = F^i p^{ij}_{k-1|k-1} \quad \forall j \in 1 \ldots n \setminus i \tag{4.17c}$$

Update: (Identical to SKF (4.12))

$$\hat{x}^\alpha_+ = \hat{x}^\alpha_- + K^\alpha(z_k - h(\hat{x}^\alpha_-)) \tag{4.18a}$$

$$\hat{x}^\beta_+ = \hat{x}^\beta_- \tag{4.18b}$$

$$P^\alpha_+ = (I - K^\alpha H^\alpha) P^\alpha_- \tag{4.18c}$$

$$P^{\alpha\beta}_+ = (I - K^\alpha H^\alpha) P^{\alpha\beta}_- \tag{4.18d}$$

$$P^{\beta\alpha}_+ = (P^{\alpha\beta}_+)^\top \tag{4.18e}$$

$$P^\beta_+ = P^\beta_- \tag{4.18f}$$

A graphical representation of the distributed SKF update is shown in Figure 4.5 on Page 66, where $\alpha = \{1,2\}$ and $\beta = \{3,4\}$. Compared with the distributed EKF, the amount of information transmitted is slightly smaller, as the state estimates of β are not updated, and thus some information is not required to be sent to these nodes.

However, in this configuration, information is still transmitted and received from every node in the network, as information is required from every node to calculate the term $P^{\alpha\beta}_+$. If we expand the right-hand side of (4.18d), we have

$$P^{\alpha\beta}_+ = \begin{bmatrix} I - K^{\alpha_1} H^{\alpha 1} & -K^{\alpha_1} H^{\alpha_2} & \cdots \\ -K^{\alpha_2} H^{\alpha 1} & I - K^{\alpha_2} H^{\alpha_2} & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix} \begin{bmatrix} P^{\alpha_1\beta_1}_- & P^{\alpha_1\beta_2}_- & \cdots \\ P^{\alpha_2\beta_1}_- & P^{\alpha_2\beta_2}_- & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix}. \tag{4.19}$$

Note that in order to calculate any element of the matrix $P^{\alpha\beta}_+$, the respective column of $P^{\alpha\beta}_-$ must be known. As these terms are distributed through the network, nodes must communicate in order to obtain this information.

However, there is one special case where the update step can be computed without any communication between nodes. This occurs if the measurement function is β-independent and α only contains a single node ($|\alpha| = 1$). If we expand the right-hand side of (4.18d) for the specific case of $\alpha = \{i\}$, we have

$$P_+^{\alpha\beta} = \left[I - K^i H^i\right] \begin{bmatrix} P_-^{i\beta_1} & P_-^{i\beta_2} & \cdots \end{bmatrix} \tag{4.20}$$

which can then be represented using the decomposed covariance terms stored on node i,

$$p_{k|k}^{ij} = (I - K^i H^i) p_{k|k-1}^{ij} \qquad \forall j \in \beta \tag{4.21}$$

Note how the update to p^{ij} in (4.21) is a pre-multiplication, similar to the prediction step (4.17c). Thus, no communication is required between nodes, as all calculations can be performed with information already known to node i. A graphical representation of this update is shown in Figure 4.6 on Page 66.

4.2.3 The Approximated Schmidt-Kalman Filter

Luft, Schubert *et al.* [98] aim to reduce the amount of information transmitted between nodes during the update step. As seen above, if the measurement is partitioned such that $|\alpha| = 1$, then the Schmidt-Kalman update can be performed without any communication. Luft, Schubert *et al.* propose an approximation for the case where $|\alpha| = 2$, which allows an approximated Schmidt-Kalman update to be performed only with communication between the two nodes in α.

For the SKF in the case where $|\alpha| = 2$, (4.18d) expands to

$$P_+^{\alpha\beta} = \begin{bmatrix} I - K^{\alpha_1} H^{\alpha_1} & -K^{\alpha_1} H^{\alpha_2} \\ -K^{\alpha_2} H^{\alpha_1} & I - K^{\alpha_2} H^{\alpha_2} \end{bmatrix} \begin{bmatrix} P_-^{\alpha_1\beta_1} & P_-^{\alpha_1\beta_2} & \cdots \\ P_-^{\alpha_2\beta_1} & P_-^{\alpha_2\beta_2} & \cdots \end{bmatrix}. \tag{4.22}$$

The approximation proposed by Luft, Schubert *et al.* [98] is

$$P^{jk} \approx P^{ji} (P^i)^{-1} P^{ik} \tag{4.23}$$

for any $i, j, k \in N$. If we expand out the first element of $P_+^{\alpha\beta}$ from (4.22) and substitute in the approximation $P_-^{\alpha_2\beta_1} \approx P_-^{\alpha_2\alpha_1}(P_-^{\alpha_1})^{-1} P_-^{\alpha_1\beta_1}$, we have

$$\begin{aligned} P_+^{\alpha_1\beta_1} &= (I - K^{\alpha_1} H^{\alpha_1}) P_-^{\alpha_1\beta_1} - K^{\alpha_1} H^{\alpha_2} P_-^{\alpha_2\beta_1} \\ &\approx (I - K^{\alpha_1} H^{\alpha_1}) P_-^{\alpha_1\beta_1} - K^{\alpha_1} H^{\alpha_2} P_-^{\alpha_2\alpha_1} (P_-^{\alpha_1})^{-1} P_-^{\alpha_1\beta_1} \\ &= \left((I - K^{\alpha_1} H^{\alpha_1}) P_-^{\alpha_1} - K^{\alpha_1} H^{\alpha_2} P_-^{\alpha_2\alpha_1}\right) (P_-^{\alpha_1})^{-1} P_-^{\alpha_1\beta_1} \\ &= P_+^{\alpha_1} (P_-^{\alpha_1})^{-1} P_-^{\alpha_1\beta_1}. \end{aligned}$$

Similar approximations can be made for each element of (4.22), which yields the following result

$$P_+^{\alpha\beta} \approx \begin{bmatrix} P_+^{\alpha_1}(P_-^{\alpha_1})^{-1} & 0 \\ 0 & P_+^{\alpha_2}(P_-^{\alpha_2})^{-1} \end{bmatrix} P_-^{\alpha\beta}. \tag{4.24}$$

As the approximated update is now a pre-multiplication with a block-diagonal matrix, the update can be performed without having to reconstruct the values of $P_-^{\alpha\beta}$. Now, the update step can be performed in the same way as the distributed Schmidt-Kalman filter in (4.18), but with (4.18d) replaced by

$$p_{k|k}^{\alpha_1 j} = P_+^{\alpha_1}(P_-^{\alpha_1})^{-1} p_{k|k-1}^{\alpha_1 j} \qquad \forall j \in \beta \tag{4.25a}$$
$$p_{k|k}^{\alpha_2 j} = P_+^{\alpha_2}(P_-^{\alpha_2})^{-1} p_{k|k-1}^{\alpha_2 j} \qquad \forall j \in \beta. \tag{4.25b}$$

Algorithm 8: Distributed Approximate SKF[98]

Predict: (Identical to Distributed SKF (4.17))
On each node $i \in N$:

$$\hat{x}_-^i = f^i(\hat{x}_{k-1|k-1}^i, u^i) \tag{4.26a}$$
$$P_-^i = F^i P_{k-1|k-1}^i {F^i}^\top + Q^i \tag{4.26b}$$
$$p_{k|k-1}^{ij} = F^i p_{k-1|k-1}^{ij} \quad \forall j \in 1 \ldots n \setminus i \tag{4.26c}$$

Update for $|\alpha| = 1$:

$$\hat{x}_+^\alpha = \hat{x}_-^\alpha + K^\alpha(z_k - h(\hat{x}_-^\alpha)) \tag{4.27a}$$
$$\hat{x}_+^\beta = \hat{x}_-^\beta \tag{4.27b}$$
$$P_+^\alpha = (I - K^\alpha H^\alpha) P_-^\alpha \tag{4.27c}$$
$$p_+^{\alpha j} = (I - K^\alpha H^\alpha) p_-^{\alpha j} \quad \forall j \in \beta \tag{4.27d}$$
$$P_+^\beta = P_-^\beta \tag{4.27e}$$

Update for $|\alpha| = 2$:
Let $\alpha = \{\alpha_1, \alpha_2\}$

$$\hat{x}_+^\alpha = \hat{x}_-^\alpha + K^\alpha(z_k - h(\hat{x}_-^\alpha)) \tag{4.28a}$$
$$\hat{x}_+^\beta = \hat{x}_-^\beta \tag{4.28b}$$
$$P_+^\alpha = (I - K^\alpha H^\alpha) P_-^\alpha \tag{4.28c}$$
$$p_{k|k}^{\alpha_1 j} = P_+^{\alpha_1}(P_-^{\alpha_1})^{-1} p_{k|k-1}^{\alpha_1 j} \qquad \forall j \in \beta \tag{4.28d}$$
$$p_{k|k}^{\alpha_2 j} = P_+^{\alpha_2}(P_-^{\alpha_2})^{-1} p_{k|k-1}^{\alpha_2 j} \qquad \forall j \in \beta. \tag{4.28e}$$
$$P_+^\beta = P_-^\beta \tag{4.28f}$$

We will denote this version of the filter as the Distributed Approximate Schmidt-Kalman Filter (ASKF). A graphical representation of this update step of the ASKF with $\alpha = \{1, 2\}$ is shown in Figure 4.7 on Page 67, where we can observe that only the nodes in α are required to communicate. This algorithm makes a trade-off between communication requirements and filter performance. Two factors contribute to the reduction in performance; one is the SKF update strategy which discards update terms for nodes in β, and the other is the approximation introduced in (4.23). The benefit of these modifications is that it allows the computation to be performed completely locally within α, as long as $|\alpha| \leq 2$.

This constraint fits quite well with the collaborative localisation problem when we consider the typical measurements that are made. Sensors that measure a vehicle's position relative to the environment, such as GNSS, landmark sensors, magnetometers etc. all only depend on the state of a single vehicle. Thus, the system can be partitioned such that $|\alpha| = 1$, and the SKF update can be applied. For sensors that measure other vehicles, such as a camera measuring a relative bearing or a time-of-flight sensor measuring relative distance, the measurement model will depend on the state of both the vehicle being measured and the vehicle performing the measurement. In this case, the system can be partitioned such that $|\alpha| = 2$, and the ASKF update can be applied. A sensor would rarely measure some property of the system that is a function of 3 or more vehicles.

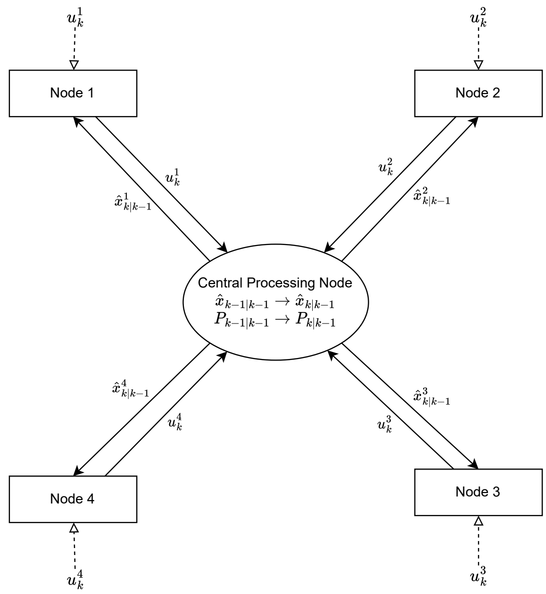


FIGURE 4.1: Information flows for prediction step for Central EKF for $n = 4$. Dotted lines indicate measurement information and solid lines indicate the transmission of data from one node to another. Symbols inside nodes indicate information storage and computation.

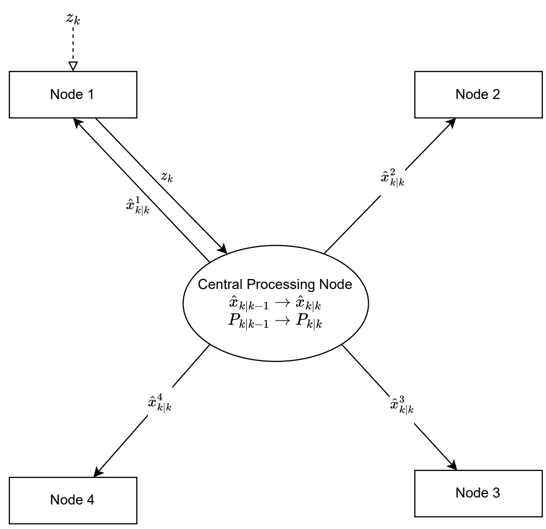


FIGURE 4.2: Information flows for update step for Central EKF

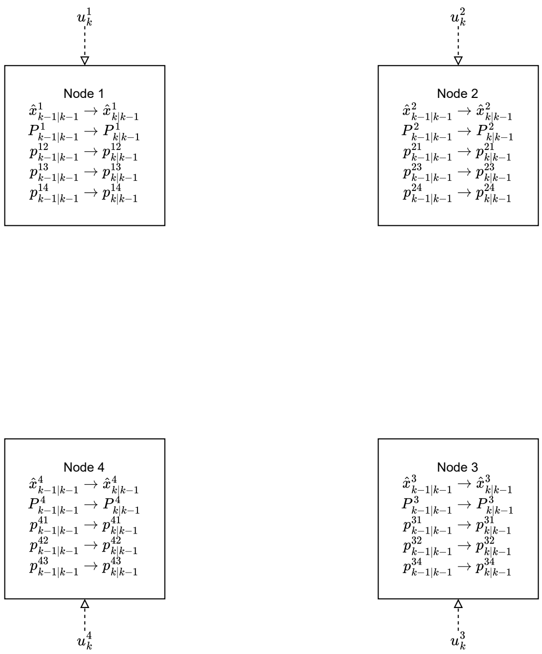
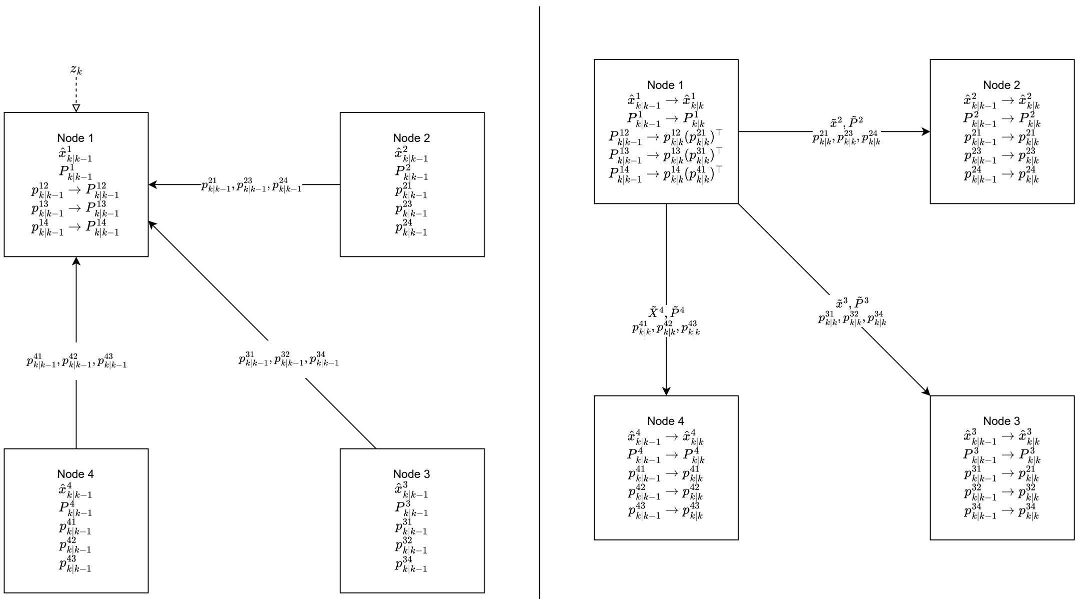


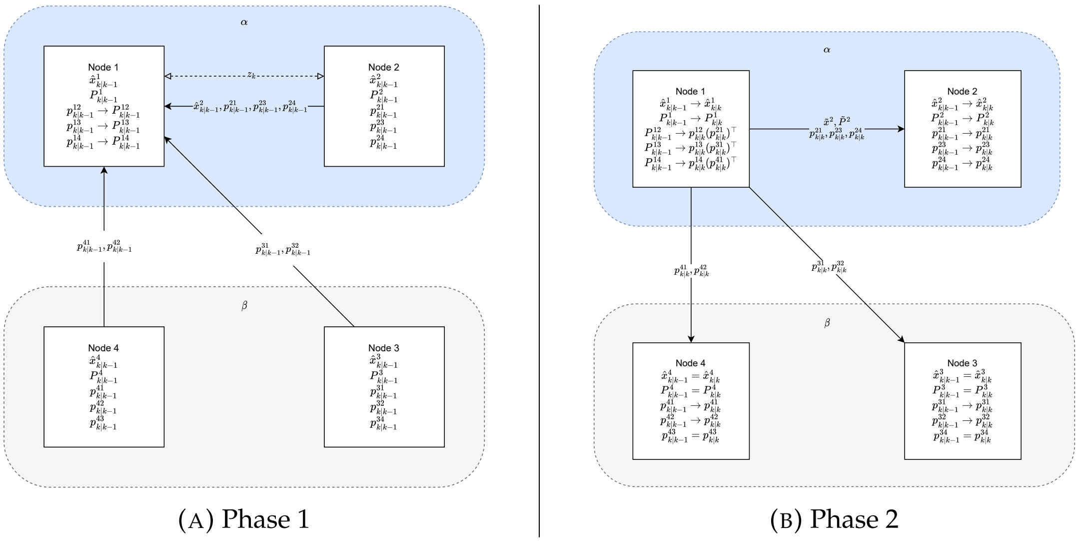
FIGURE 4.3: Information flows for prediction step of Distributed EKF. Note that no communication between nodes is required to perform the prediction step.



(A) Phase 1

(B) Phase 2

FIGURE 4.4: Information flows for the update step of Distributed EKF for a measurement made at node 1. Here, $\tilde{x}^i = K^i(z_k - h(\hat{x}^1_-))$ and $\tilde{P}^i = -K^i H^1 P^{1i}_- \; \forall i \in \{2, 3, 4\}$.



(A) Phase 1

(B) Phase 2

FIGURE 4.5: Information flows for the update step of the SKF where $\alpha = \{1, 2\}$ and the measurement model is β-independent. The shaded regions indicate the boundaries of the α and β sets.

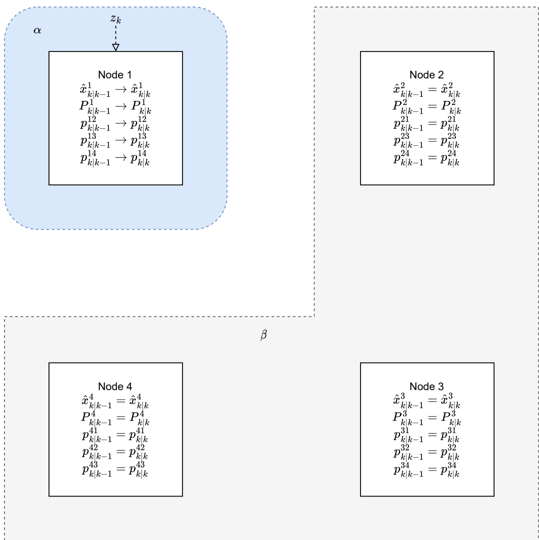
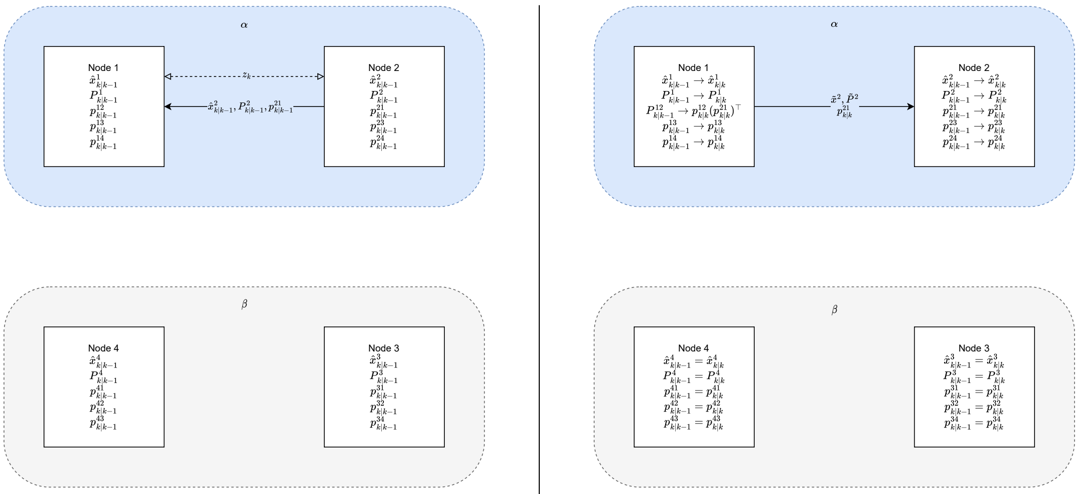


FIGURE 4.6: Information flows for the update step of the SKF where $\alpha = \{1\}$ and the measurement model is β-independent. Observe that no communication between any nodes is required.



(A) Phase 1 (B) Phase 2

FIGURE 4.7: Information flows for the update step of the ASKF where $\alpha = \{1, 2\}$. Note that communication is only required between the nodes in α, compared with the communication requirements of the SKF in Figure 4.5.

4.3 Derivation of a Centralised Minimum Energy Filter

The analysis in the preceding section shows how the equations of the extended Kalman filter (EKF) can be distributed across a network of nodes in order to remove the single point of failure and reduce the communication requirements of the network. Further to this, the Schmidt Kalman filter (SKF) and the Approximate Schmidt Kalman filter (ASKF) show how certain approximations can result in even further reductions in communication at the expense of a small reduction in filter performance. One will note, however, that it is the structure of the EKF equations and some particular matrix properties allow for this decomposition. It is not immediately obvious if the techniques applied here also work for filter algorithms other than the EKF.

Recalling the previous discussion in Chapter 2 on the EKF and its major weaknesses and limitations, there is significant interest in exploring collaborative localisation algorithms using more advanced filter designs. To address this, we investigate whether we can apply a similar methodology to a minimum-energy filter to create a distributed collaborative localisation algorithm. The first step in this process is to derive a centralised collaborative localisation filter.

4.3.1 Problem Formulation

The problem we will study here is an extension of the single-vehicle localisation problem formulated in Section 3.1. Rather than considering just a single vehicle, we consider the problem of determining the pose of a network of n vehicles that are free to move in 3-dimensional space. The set of vehicles in the network is defined as $\boldsymbol{V} := \{1, \ldots, n\}$, and the indices i and $j \in \boldsymbol{V}$ are used throughout this chapter to refer to particular vehicles in the network[4]. These vehicles could be unmanned aerial vehicles (UAV), unmanned ground vehicles (UGV), or a combination of both.

As in the previous problem, the vehicles move freely through a known environment where there are a set of n_L fixed landmark points, indexed by the set $\boldsymbol{L} := \{1, \ldots, n_L\}$. For a given landmark point, $l \in \boldsymbol{L}$, the known position of the landmark with respect to the inertial frame is $p_l \in \mathbb{R}^3$. Similarly, each vehicle is equipped with a strap-down inertial measurement unit (IMU), which measures the linear acceleration and angular velocity of the vehicle with respect to the inertial frame, as well as a landmark sensor which measures the relative position of the vehicle with respect to each landmark.

[4]The colour-coding of the indices i, j, and l throughout this chapter is intended to assist the reader in more easily identifying terms relevant to each vehicle or landmark

Similar to 3.1.1, the state of a single vehicle $i \in \boldsymbol{V}$ is represented by X^i as

$$X^i := \left(\begin{bmatrix} R^i & x^i & v^i \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \theta^i, \phi^i \right) \in \mathrm{SE}_2(3) \times \mathbb{R}^3 \times \mathbb{R}^3 = \mathrm{G}$$

where the orientation R^i, the position x^i, and the velocity v^i of the body-fixed frame are with respect to the inertial frame and are expressed in the coordinates of the inertial frame. The biases θ^i and ϕ^i respectively represent the angular and linear biases of the IMU onboard vehicle i.

Combining the states of all the vehicles in the network, the global state, X is represented by the direct product group

$$X := \left(X^1, X^2, \dots, X^n \right) \in \mathrm{G}^n$$

Using the same kinematic model and IMU measurement model as Section 3.1.2, the state of each vehicle evolves according to

$$\dot{X}^i = X^i \left(\lambda^i(X^i, u^i) + B^i(\delta^i) \right) \tag{4.29}$$

where $u^i \in \mathbb{R}^6$ is the IMU measurement, $\lambda^i : \mathrm{G} \times \mathbb{R}^6 \to \mathfrak{g}$ is given by

$$\lambda^i(X^i, u^i) := \begin{bmatrix} u^i_\omega - \theta^i \\ {R^i}^\top v^i \\ u^i_a - \phi^i - {R^i}^\top g \\ 0 \\ 0 \end{bmatrix}^\wedge, \tag{4.30}$$

$\delta^i \in \mathbb{R}^{12}$ is the IMU measurement error signal, and the linear map $B^i : \mathbb{R}^{12} \to \mathfrak{g}$ is given by

$$B^i(\delta^i) := (\check{B}^i \delta^i)^\wedge, \quad \delta^i := \begin{bmatrix} \delta^i_\omega \\ \delta^i_a \\ \delta^i_\theta \\ \delta^i_\phi \end{bmatrix}, \quad \check{B}^i := \begin{bmatrix} -B^i_\omega & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & -B^i_a & 0 & 0 \\ 0 & 0 & B^i_\theta & 0 \\ 0 & 0 & 0 & B^i_\phi \end{bmatrix}. \tag{4.31}$$

Combining the state evolution of all the vehicles in the network, the global state evolution can be represented by

$$\dot{X} = X \left(\lambda(X, u) + B(\delta) \right) \tag{4.32}$$

where

$$\lambda(X, u) := \left(\lambda^1(X^1, u^1), \cdots, \lambda^n(X^n, u^n) \right)$$

$$\begin{aligned} B(\delta) &:= (\check{B}\delta)^\wedge \\ \check{B} &:= \operatorname{blkdiag}(\check{B}^1, \dots, \check{B}^n) \\ \delta &:= \begin{bmatrix} \delta^1 \\ \vdots \\ \delta^n \end{bmatrix}. \end{aligned}$$

The landmark measurement $y_l^i \in \mathbb{R}^3$ represents a measurement made by vehicle $i \in \boldsymbol{V}$ to a landmark $l \in \boldsymbol{L}$, and is modelled by

$$y_l^i = h_l^i(X(t)) + D\epsilon_l^i(t) \tag{4.33}$$

where $\epsilon_l^i \in \mathbb{R}^3$ is the unknown sensor error, $D \in \mathbb{R}^{3\times3}$ is invertible, and the measurement model, $h_l^i : \mathrm{G}^n \to \mathbb{R}^3$ is given by

$$h_l^i(X) = {R^i}^\top (p_l - x^i) \tag{4.34}$$

With each vehicle just equipped with a landmark sensor and an IMU, each vehicle would have no way to detect or perceive other vehicles in the network, reducing the problem to a set of single-vehicle localisation problems. Thus, we will extend the problem, and introduce a new inter-vehicle sensor, which measures the relative position of a vehicle with respect to another. We can model the inter-vehicle measurements similarly to the landmark measurement model. The landmark, l, is substituted by a marker point $m_j \in \mathbb{R}^3$ located at a known fixed point in the body-fixed frame of vehicle $j \in \boldsymbol{V}$. A measurement made at time t of the marker on vehicle j, received by vehicle i is denoted by $y_j^i(t) \in \mathbb{R}^3$,

$$y_j^i(t) := h_j^i(X(t)) + D\epsilon_j^i(t), \tag{4.35}$$

where $\epsilon_j^i \in \mathbb{R}^3$ is the unknown sensor error, $D \in \mathbb{R}^{3\times3}$ is invertible, and the measurement model, $h_j^i : \mathrm{G}^n \to \mathbb{R}^3$ is given by

$$h_j^i(X) := (R^i)^\top (R^j m_j + x^j - x^i). \tag{4.36}$$

Given that each vehicle may make a relative measurement to every vehicle other than itself, we define $\tilde{\boldsymbol{V}}^2$ as the set of all possible indices for the relative measurement, given by

$$\tilde{\boldsymbol{V}}^2 := \{(i, j) \mid i \in \boldsymbol{V}, j \in \boldsymbol{V}, i \neq j\}$$

An illustration of the multi-vehicle collaborative localisation problem is shown in Figure 4.8.

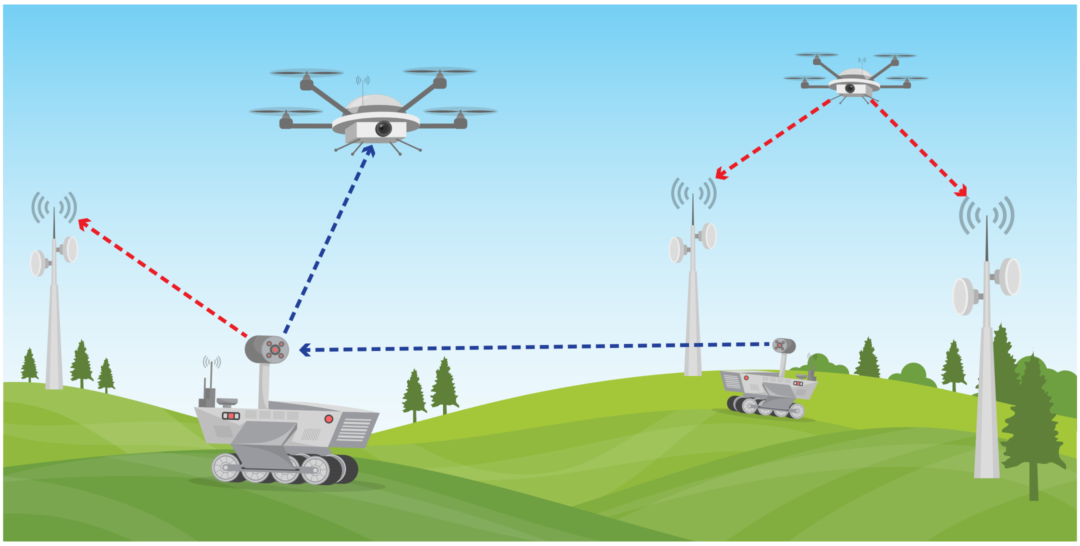


FIGURE 4.8: Illustration of the multi-vehicle collaborative localisation problem showing an example network of 4 vehicles, 3 landmarks. A selection of measurements are illustrated with dashed lines, where red lines indicate landmark measurements made by vehicles and blue lines indicate inter-vehicle measurements.

4.3.2 Minimum Energy Filter Construction

Following the same procedure as the previous problem, we define a cost function J_t on the error signals, which is given by

$$\begin{aligned} J_t(\delta_{[t_0,t]}, \epsilon_{[t_0,t]}, X_0) :=& \\ \frac{1}{2} J_0(X_0) + \frac{1}{2}\int_{t_0}^{t} \|\delta(\tau)\|_W^2 + \sum_{i,l \in \mathbf{V}\times\mathbf{L}} \left\|\epsilon_l^i(\tau)\right\|_{Q_l^i}^2 + \sum_{i,j\in\tilde{\mathbf{V}}^2} \left\|\epsilon_j^i(\tau)\right\|_{Q_j^i}^2 \mathrm{d}\tau \end{aligned} \tag{4.37}$$

where $J_0 : \mathrm{G}^n \to \mathbb{R}$ is some cost on the initial state, X_0, with a unique global minimum. $Q_l^i, Q_j^i \succ 0 \in \mathbb{R}^{3\times 3}$ and $W \succ 0 \in \mathbb{R}^{12n\times 12n}$ are used to weight the norms of the respective error signals, with

$$W := \mathrm{blkdiag}(W^1, \dots, W^n).$$

The optimal minimum-energy estimate is then defined as the terminal point of the trajectory which minimises J_t, i.e.

$$\hat{X}(t) := X^*_{[t_0,t]}(t).$$

We can observe that this problem is also an instance of the general problem formulation in [77], and thus we can adopt the abstract second-order minimum energy solution.

The second-order minimum energy estimate for the joint state of the network is defined by a pair of ODEs which describe the time evolution of the state estimate, $\hat{X}(t) \in \mathrm{G}^n$, and the associated gain operator, $K(t) : (\mathfrak{g}^n)^* \to \mathfrak{g}^n$. They are given by

$$\dot{\hat{X}} = \hat{X}\left(\lambda(\hat{X}, u) + K \circ r\right) \tag{4.38}$$

$$\dot{K} = A \circ K + K \circ A^* - K \circ E \circ K + B \circ W^{-1} \circ B^* - \Lambda_{K \circ r} \circ K - K \circ \Lambda^*_{K \circ r} \tag{4.39}$$

with initial conditions

$$\hat{X}(t_0) = \arg\min_X J_0(X)$$
$$K(t_0) = (T_e L^*_{\hat{X}_0} \circ \operatorname{Hess} J_0(\hat{X}_0) \circ T_e L_{\hat{X}_0})^{-1}.$$

The operator Λ is the connection function (see A.2.2), and the residual, $r = r(\hat{X}, y) \in (\mathfrak{g}^n)^*$, is given by

$$r := \sum_{i,l \in \boldsymbol{V} \times \boldsymbol{L}} T_e L^*_{\hat{X}} \circ \left((M^i_l \circ (y^i_l - \hat{y}^i_l)) \circ \mathbf{d}\hat{y}^i_l\right) + \sum_{i,j \in \tilde{\boldsymbol{V}}^2} T_e L^*_{\hat{X}} \circ \left(\left(M^i_j \circ (y^i_j - \hat{y}^i_j)\right) \circ \mathbf{d}\hat{y}^i_j\right) \tag{4.40}$$

where

$$\hat{y}^i_l := h^i_l(\hat{X})$$
$$\hat{y}^i_j := h^i_j(\hat{X})$$
$$M^i_l := (D^{-1})^\top Q^i_l D^{-1}$$
$$M^i_j := (D^{-1})^\top Q^i_j D^{-1}.$$

The operators $A = A(\hat{X}, u)$ and $E = E(\hat{X}, y)$ are defined by

$$A := \mathbf{d}_1 \lambda(\hat{X}, u) \circ T_e L_{\hat{X}} - \operatorname{ad}_{\lambda(\hat{X}, u)} - T_{\lambda(\hat{X}, u)}, \tag{4.41}$$

$$E := -T_e L^*_{\hat{X}} \circ \left(\sum_{i,l \in \boldsymbol{V} \times \boldsymbol{L}} E^i_l + \sum_{i,j \in \tilde{\boldsymbol{V}}^2} E^i_j \right) \circ T_e L_{\hat{X}}, \tag{4.42}$$

where

$$E^i_l := \left(M^i_l \circ (y^i_l - \hat{y}^i_l)\right)^{T_{\hat{X}} \mathrm{G}^n} \circ \operatorname{Hess} \hat{y}^i_l - (\mathbf{d}\hat{y}^i_l)^* \circ M^i_l \circ \mathbf{d}\hat{y}^i_l$$
$$E^i_j := \left(M^i_j \circ (y^i_j - \hat{y}^i_j)\right)^{T_{\hat{X}} \mathrm{G}^n} \circ \operatorname{Hess} \hat{y}^i_j - (\mathbf{d}\hat{y}^i_j)^* \circ M^i_j \circ \mathbf{d}\hat{y}^i_j$$

4.3.3 Centralised Filter Derivation

As with the single-vehicle filter, the equations in the previous section only describe the filter in terms of abstract operators. In order to be able to evaluate the filter equations on a computer, we must define an explicit matrix representation for each term above. In this section, we introduce a series of definitions and lemmas describing how these explicit representations relate to the abstract operators.

Definition 4.1. *Consider the operator $A : \mathfrak{g}^n \to \mathfrak{g}^n$ defined in (4.41). The operator $\check{A} \in \mathbb{R}^{15n\times 15n}$ is the matrix representation of A, defined by*

$$(A \circ \Gamma)^\vee = \check{A}\Gamma^\vee$$

for all $\Gamma \in \mathfrak{g}^n$.

By a simple extension of the result of Lemma 3.7, the matrix representation $\check{A}$ is given by

$$\check{A} = \operatorname{blkdiag}(\check{A}^1, \dots, \check{A}^n)$$

where

$$\check{A}^i = -\begin{bmatrix} (u_\omega^i - \hat{\theta}^i)_\times & \mathbf{0} & \mathbf{0} & \boldsymbol{I} & \mathbf{0} \\ \mathbf{0} & (u_\omega^i - \hat{\theta}^i)_\times & -\boldsymbol{I} & \mathbf{0} & \mathbf{0} \\ (u_a^i - \hat{\phi}^i)_\times & \mathbf{0} & (u_\omega^i - \hat{\theta}^i)_\times & \mathbf{0} & \boldsymbol{I} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \forall i \in \boldsymbol{V}.$$

Definition 4.2. *Recall the model of a landmark measurement, $h_l^i : \mathrm{G}^n \to \mathbb{R}^3$, as defined in (4.34). The matrix representation, $H_l^i \in \mathbb{R}^{3\times 15n}$, of the derivative of the measurement model is defined by*

$$\mathbf{d}h_l^i(X) \circ X\Gamma = H_l^i(X)\Gamma^\vee$$

for all $\Gamma \in \mathfrak{g}^n$.

Recalling Lemma 3.4, the matrix representation H_l^i is given by

$$H_l^i(X) = \begin{bmatrix} \bar{H}_l^i(1) & \dots & \bar{H}_l^i(n) \end{bmatrix}$$

where $\bar{H}_l^i \in \mathbb{R}^{3\times 15}$ is defined as

$$\bar{H}_l^i(k) := \begin{cases} \begin{bmatrix} h_l^i(X)_\times & -\boldsymbol{I}_3 & \mathbf{0}_{3\times 9} \end{bmatrix} & k = i \\ \mathbf{0}_{3\times 15} & \text{otherwise} \end{cases}$$

Definition 4.3. *Recall the model of an inter-vehicle measurement, $h_j^i : \mathrm{G}^n \to \mathbb{R}^3$, between two vehicles $i, j \in \tilde{\boldsymbol{V}}^2$ as defined in (4.36). The matrix representation, $H_j^i \in \mathbb{R}^{3\times 15n}$, of the derivative of the measurement model is defined by*

$$\mathbf{d}h_j^i(X) \circ X\Gamma = H_j^i(X)\Gamma^\vee$$

for all $\Gamma \in \mathfrak{g}^n$.

Lemma 4.4. *The matrix representation, $H^i_j \in \mathbb{R}^{3\times 15n}$ from Definition 4.3 is given by*

$$H^i_j(X) = \begin{bmatrix} \bar{H}^i_j(1) & \dots & \bar{H}^i_j(n) \end{bmatrix}$$

where $\bar{H}^i_j \in \mathbb{R}^{3\times 15}$ is defined as

$$\bar{H}^i_j(k) := \begin{cases} \begin{bmatrix} h^i_j(X)_\times & -\boldsymbol{I}_3 & \boldsymbol{0}_{3\times 9} \end{bmatrix} & k = i \\ \begin{bmatrix} -R_{ij}m_{j\times} & R_{ij} & \boldsymbol{0}_{3\times 9} \end{bmatrix} & k = j \\ \boldsymbol{0}_{3\times 15} & \textit{otherwise} \end{cases}$$

Proof. Consider the derivative of the measurement function in an arbitrary direction $X\Gamma \in T_X G$

$$\mathbf{d}h^i_j(X) \circ X\Gamma = \mathbf{d}(R^i)^\top (R^j m_j + x^j - x^i) \circ X\Gamma$$

by applying the product rule and the identities from Appendix A.4, we have

$$\begin{aligned}
&\mathbf{d}h^i_j(X) \circ X\Gamma && \text{(4.43a)}\\
&\quad = (\mathbf{d}R^i \circ X\Gamma)^\top (R^j m_j + x^j - x^i) + (R^i)^\top \mathbf{d}(R^j m_j + x^j - x^i) \circ X\Gamma && \text{(4.43b)}\\
&\quad = (R^i \Gamma^i_{R\times})^\top (R^j m_j + x^j - x^i) + (R^i)^\top (R^j \Gamma^j_{R\times} m_j + R^j \Gamma^j_x - R^i \Gamma^i_x) && \text{(4.43c)}\\
&\quad = -\Gamma^i_{R\times} h^i_j(X) + (R^i)^\top R^j (\Gamma^j_{R\times} m_j + \Gamma^j_x) - \Gamma^i_x && \text{(4.43d)}\\
&\quad = h^i_j(X)_\times \Gamma^i_R + (R^i)^\top R^j (-m_{j\times} \Gamma^j_R + \Gamma^j_x) - \Gamma^i_x && \text{(4.43e)}\\
&\quad = \bar{H}^i_j(i) {\Gamma^i}^\vee + \bar{H}^i_j(j) {\Gamma^j}^\vee && \text{(4.43f)}
\end{aligned}$$

and the lemma follows. □

Lemma 4.5. *Consider the inter-vehicle measurement model defined in (4.36) for a measurement of vehicle j made by vehicle i. The Hessian of the measurement function in two arbitrary direction $X\Gamma, X\Psi \in T_X G^n$ is given by*

$$\begin{aligned}
\operatorname{Hess} h^i_j(X)(X\Gamma)(X\Psi) = \\
-\frac{1}{2}\left(\Gamma^i_{R\times} H^i_j(X) + \Gamma^i_{R\times} R_{ij} L_j - R_{ij} \Gamma^j_{R\times} L_j\right) \Psi^\vee \\
-\frac{1}{2}\left(\Psi^i_{R\times} H^i_j(X) + \Psi^i_{R\times} R_{ij} L_j - R_{ij} \Psi^j_{R\times} L_j\right) \Gamma^\vee
\end{aligned}$$

where $L_j \in \mathbb{R}^{3\times 15n}$ is defined by

$$L_j := \begin{bmatrix} \bar{L}_j(1) & \dots & \bar{L}_j(n) \end{bmatrix}$$

$$\bar{L}_j(k) := \begin{cases} \begin{bmatrix} -m_{j\times} & \boldsymbol{I}_3 & \boldsymbol{0}_{3\times 9} \end{bmatrix} & k = j \\ \boldsymbol{0}_{3\times 15} & \textit{otherwise} \end{cases}$$

Proof. The Hessian of the measurement function is given by

$$\operatorname{Hess} h_j^i(X)(X\Gamma)(X\Psi) = \mathbf{d}(\mathbf{d}h_j^i(X) \circ X\Psi) \circ X\Gamma - \mathbf{d}h_j^i(X) \circ X\Lambda_\Gamma(\Psi) \tag{4.44}$$

The first term in the Hessian is

$$\mathbf{d}(\mathbf{d}h_j^i(X) \circ X\Psi) \circ X\Gamma$$

Substituting in (4.43d),

$$\begin{aligned}
&\mathbf{d}(\mathbf{d}h_j^i(X) \circ X\Psi) \circ X\Gamma \\
&\qquad = \mathbf{d}(-\Psi^i_{R\times} h_j^i(X) + R_{ij}(\Psi^j_{R\times} m_j + \Psi^j_x) - \Psi^i_x) \circ X\Gamma \\
&\qquad = -\Psi^i_{R\times}\mathbf{d}h_j^i(X) \circ X\Gamma + \mathbf{d}R_{ij} \circ X\Gamma \cdot (\Psi^j_{R\times} m_j + \Psi^j_x) \\
&\qquad = -\Psi^i_{R\times} H_j^i(X)\Gamma^\vee + (-\Gamma^i_{R\times} R_{ij} + R_{ij}\Gamma^j_{R\times})(-m_{j\times}\Psi^j_R + \Psi^j_x) \\
&\qquad = -\Psi^i_{R\times} H_j^i(X)\Gamma^\vee - \Gamma^i_{R\times} R_{ij}(-m_{j\times}\Psi^j_R + \Psi^j_x) + R_{ij}\Gamma^j_{R\times}(-m_{j\times}\Psi^j_R + \Psi^j_x) \\
&\qquad = -\Psi^i_{R\times} H_j^i(X)\Gamma^\vee - \Gamma^i_{R\times} R_{ij} L_j \Psi^\vee + R_{ij}\Gamma^j_{R\times} L_j \Psi^\vee
\end{aligned}$$

Considering the second term of (4.44), and using the (0)-connection function, Λ_0, from (A.3), we have

$$\mathbf{d}h_j^i(X) \circ X\Lambda_\Gamma(\Psi) = \frac{1}{2}\mathbf{d}h_j^i(X) \circ X[\Gamma, \Psi]$$

Then, substituting in the result from (4.43d),

$$\begin{aligned}
\mathbf{d}h_j^i(X) \circ X\Lambda_\Gamma(\Psi) = \frac{1}{2}\left(-[\Gamma^i_{R\times}, \Psi^i_{R\times}]h_j^i(X) - \Gamma^i_{R\times}\Psi^i_x + \Psi^i_{R\times}\Gamma^i_x\right) \\
+ \frac{1}{2}R_{ij}\left([\Gamma^j_{R\times}, \Psi^j_{R\times}]m_j + \Gamma^j_{R\times}\Psi^j_x - \Psi^j_{R\times}\Gamma^j_x\right),
\end{aligned}$$

expanding the Lie bracket operators,

$$\begin{aligned}
\mathbf{d}h_j^i(X) \circ X\Lambda_\Gamma(\Psi) = \frac{1}{2}\left(-(\Gamma^i_{R\times}\Psi^i_{R\times} - \Psi^i_{R\times}\Gamma^i_{R\times})h_j^i(X) - \Gamma^i_{R\times}\Psi^i_x + \Psi^i_{R\times}\Gamma^i_x\right) \\
+ \frac{1}{2}R_{ij}\left((\Gamma^j_{R\times}\Psi^j_{R\times} - \Psi^j_{R\times}\Gamma^j_{R\times})m_j + \Gamma^j_{R\times}\Psi^j_x - \Psi^j_{R\times}\Gamma^j_x\right),
\end{aligned}$$

applying the cross-product rules,

$$\begin{aligned}
\mathbf{d}h_j^i(X) \circ X\Lambda_\Gamma(\Psi) = \frac{1}{2}\left(\Gamma^i_{R\times} h_j^i(X)_\times \Psi^i_R - \Psi^i_{R\times} h_j^i(X)_\times \Gamma^i_R - \Gamma^i_{R\times}\Psi^i_x + \Psi^i_{R\times}\Gamma^i_x\right) \\
+ \frac{1}{2}R_{ij}\left(-\Gamma^j_{R\times} m_{j\times}\Psi^j_R + \Psi^j_{R\times} m_{j\times}\Gamma^j_R + \Gamma^j_{R\times}\Psi^j_x - \Psi^j_{R\times}\Gamma^j_x\right),
\end{aligned}$$

substituting L_j and $\bar{H}_j^i(i)$,

$$\mathbf{d}h^i_j(X) \circ X\Lambda_\Gamma(\Psi) = \frac{1}{2}\left(\Gamma^i_{R\times}\bar{H}^i_j(i)(\Psi^i)^\vee - \Psi^i_{R\times}\bar{H}^i_j(i)(\Gamma^i)^\vee\right) + \frac{1}{2}R_{ij}\left(\Gamma^j_{R\times}L_j\Psi^\vee - \Psi^j_{R\times}L_j\Gamma^\vee\right),$$

and, after factoring out common terms, we have

$$\mathbf{d}h^i_j(X) \circ X\Lambda_\Gamma(\Psi) = \frac{1}{2}\left(\Gamma^i_{R\times}H^i_j(X) - \Gamma^i_{R\times}R_{ij}L_j + R_{ij}\Gamma^j_{R\times}L_j\right)\Psi^\vee - \frac{1}{2}\left(\Psi^i_{R\times}H^i_j(X) - \Psi^i_{R\times}R_{ij}L_j + R_{ij}\Psi^j_{R\times}L_j\right)\Gamma^\vee.$$

Substituting these results back into (4.44) gives

$$\begin{aligned}\operatorname{Hess} h^i_j(X)(X\Gamma)(X\Psi) = \\ &- \Psi^i_{R\times}H^i_j(X)\Gamma^\vee - \Gamma^i_{R\times}R_{ij}L_j\Psi^\vee + R_{ij}\Gamma^j_{R\times}L_j\Psi^\vee \\ &- \frac{1}{2}\left(\Gamma^i_{R\times}H^i_j(X) - \Gamma^i_{R\times}R_{ij}L_j + R_{ij}\Gamma^j_{R\times}L_j\right)\Psi^\vee \\ &+ \frac{1}{2}\left(\Psi^i_{R\times}H^i_j(X) - \Psi^i_{R\times}R_{ij}L_j + R_{ij}\Psi^j_{R\times}L_j\right)\Gamma^\vee,\end{aligned}$$

which simplifies to

$$\operatorname{Hess} h^i_j(X)(X\Gamma)(X\Psi) = -\frac{1}{2}\left(\Gamma^i_{R\times}H^i_j(X) + \Gamma^i_{R\times}R_{ij}L_j - R_{ij}\Gamma^j_{R\times}L_j\right)\Psi^\vee - \frac{1}{2}\left(\Psi^i_{R\times}H^i_j(X) + \Psi^i_{R\times}R_{ij}L_j - R_{ij}\Psi^j_{R\times}L_j\right)\Gamma^\vee.$$

□

Definition 4.6. *Consider the operator $E : \mathfrak{g}^n \to (\mathfrak{g}^n)^*$ as defined in (4.42). The operator $\check{E} \in \mathbb{R}^{15n\times 15n}$ is the matrix representation of E, defined by*

$$(E \circ \Gamma) \circ \Psi = (\Psi^\vee)^\top \check{E}\Gamma^\vee$$

for all $\Gamma, \Psi \in \mathfrak{g}^n$.

Lemma 4.7. *The matrix operator $\check{E}$, from Definition 4.6 is given by*

$$\begin{aligned}\check{E} &:= \sum_{i,l\in \boldsymbol{V}\times\boldsymbol{L}} \check{E}^i_l + \sum_{i,j\in\tilde{\boldsymbol{V}}^2} \check{E}^i_j \\ \check{E}^i_l &:= \mathbb{P}_s\left(F_i(s^i_l)^\top H^i_l(\hat{X})\right) + H^i_l(\hat{X})^\top M^i_l H^i_l(\hat{X}) \\ \check{E}^i_j &:= \mathbb{P}_s\left(F_i(s^i_j)^\top H^i_j(\hat{X}) + F_i(s^i_j)^\top R_{ij}L_j - F_j(R_{ij}{}^\top s^i_j)^\top L_j\right) \\ &\quad + H^i_j(\hat{X})^\top M^i_j H^i_j(\hat{X})\end{aligned}$$

where $s_j^i := M_j^i(y_j^i - \hat{y}_j^i)$, *and* $F_i : \mathbb{R}^3 \to \mathbb{R}^{3\times 15n}$ *is defined by*

$$F_i(s) := \begin{bmatrix} \bar{F}_i(s,1) & \dots & \bar{F}_i(s,n) \end{bmatrix}$$
$$\bar{F}_i(s,k) := \begin{cases} \begin{bmatrix} s_\times & \mathbf{0}_{3\times 12} \end{bmatrix} & k = i \\ \mathbf{0}_{3\times 15} & \textit{otherwise} \end{cases}$$

Proof. The first term, $\check{E}_l^i$, is given by Lemma 3.9 while the second term, $\check{E}_j^i$, is derived similarly as follows. To aid in the proof, we split the term $\check{E}_j^i$, from (4.42) into two components, E_1 and E_2, given by

$$E_1 = -T_eL_{\hat{X}}^* \circ \left(\left(M_j^i \circ (y_j^i - \hat{y}_j^i) \right)^{T_{\hat{X}}G^n} \circ \operatorname{Hess} \hat{y}_j^i \right) \circ T_eL_{\hat{X}}$$
$$E_2 = T_eL_{\hat{X}}^* \circ \left((\mathbf{d}\hat{y}_j^i)^* \circ M_j^i \circ \mathbf{d}\hat{y}_j^i \right) \circ T_eL_{\hat{X}}.$$

Applying the E_1 operator to two arbitrary elements $\Gamma, \Psi \in \mathfrak{g}^n$, we have

$$\begin{aligned}(E_1 \circ \Gamma) \circ \Psi &= -\left\langle T_eL_{\hat{X}}^* \circ (M_j^i \circ (y_j^i - \hat{y}_j^i))^{T_{\hat{X}}G^n} \circ \operatorname{Hess} \hat{y}_j^i \circ T_eL_{\hat{X}} \circ \Gamma, \Psi \right\rangle \\ &= -\left\langle \left((M_j^i \circ (y_j^i - \hat{y}_j^i))^{T_{\hat{X}}G^n} \circ \operatorname{Hess} \hat{y}_j^i \right) \circ \hat{X}\Gamma, \hat{X}\Psi \right\rangle\end{aligned}$$

Using the exponential functor identity (A.5) and substituting $s_j^i := M_j^i(y_j^i - \hat{y}_j^i)$, we get

$$(E_1 \circ \Gamma) \circ \Psi = -s_j^i \circ \left(\operatorname{Hess} \hat{y}_j^i \circ \hat{X}\Gamma \right) \circ \hat{X}\Psi$$

Substituting in the result from Lemma 4.5 gives

$$\begin{aligned}(E_1 \circ \Gamma) \circ \Psi = \frac{1}{2}(s_j^i)^\top &\left(\Gamma^i_{R\times} H_j^i(\hat{X}) + \Gamma^i_{R\times} R_{ij} L_j - R_{ij} \Gamma^j_{R\times} L_j \right) \Psi^\vee \\ &+ \frac{1}{2}(s_j^i)^\top \left(\Psi^i_{R\times} H_j^i(\hat{X}) + \Psi^i_{R\times} R_{ij} L_j - R_{ij} \Psi^j_{R\times} L_j \right) \Gamma^\vee\end{aligned}$$

Applying the identities from Appendix A.4 and introducing the F operator gives

$$\begin{aligned}(E_1 \circ \Gamma) \circ \Psi = \frac{1}{2}(\Gamma^\vee)^\top &\left(F_i(s_j^i) H_j^i(\hat{X}) + F_i(s_j^i) R_{ij} L_j - F_j(R_{ij}^\top s_j^i) L_j \right) \Psi^\vee \\ &+ \frac{1}{2}(\Psi^\vee)^\top \left(F_i(s_j^i) H_j^i(\hat{X}) + F_i(s_j^i) R_{ij} L_j - F_j(R_{ij}^\top s_j^i) L_j \right) \Gamma^\vee\end{aligned}$$

Using (A.1), we have the result

$$E_1 = \mathbb{P}_s \left(F_i(s_j^i)^\top H_j^i(\hat{X}) + F_i(s_j^i)^\top R_{ij} L_j - F_j({R_{ij}}^\top s_j^i)^\top L_j \right)$$

The second term E_2 is given by

$$E_2 = H_j^i(\hat{X})^\top M_j^i H_j^i(\hat{X})$$

using the result from Lemma 4.4. The result then follows from combining E_1, E_2 and $\check{E}^i_l$. □

Definition 4.8. *Consider the operator $r \in (\mathfrak{g}^n)^*$, defined in (4.40). The operator $\check{r} \in \mathbb{R}^{15n}$ is the vector representation of r, defined by*

$$r \circ \Gamma = \langle \check{r}, \Gamma^\vee \rangle = \check{r}^\top \Gamma^\vee$$

for all $\Gamma \in \mathfrak{g}^n$.

Following the same reasoning as Lemma 3.11, we can see that the operator $\check{r}$ from Definition 4.8 is given by

$$\check{r} = \sum_{i,l \in \boldsymbol{V} \times \boldsymbol{L}} H^i_l(\hat{X})^\top M^i_l (y^i_l - \hat{y}^i_l) + \sum_{i,j \in \tilde{\boldsymbol{V}}^2} H^i_j(\hat{X})^\top M^i_j (y^i_j - \hat{y}^i_j).$$

Definition 4.9. *Consider the gain operator, $K(t) : (\mathfrak{g}^n)^* \to \mathfrak{g}^n$, defined in (4.39). The matrix operator, $\check{K} \in \mathbb{R}^{15n \times 15n}$, is defined by*

$$K \circ \mu = (\check{K}\mu^\vee)^\wedge$$

for all $\mu \in (\mathfrak{g}^n)^*$

We now have all the required pieces to construct the matrix representation of the centralised continuous-time minimum energy filter. Using Definitions 4.1, 4.6, 4.8, and 4.9, the filter equations (4.38) and (4.39) can be equivalently represented by the pair of matrix differential equations as follows

Algorithm 9: Continuous-Time Minimum Energy Filter for Collaborative Localisation

$$\dot{\hat{X}} = \hat{X}\left(\lambda^\vee(\hat{X}, u) + \check{K}\check{r}\right)^\wedge \tag{4.45a}$$

$$\dot{\check{K}} = \check{A}\check{K} + \check{K}\check{A}^\top - \check{K}\check{E}\check{K} + \check{B}\check{W}^{-1}\check{B}^\top - \frac{1}{2}\check{\mathrm{ad}}_{Kr}\check{K} - \frac{1}{2}\check{K}(\check{\mathrm{ad}}_{Kr})^\top \tag{4.45b}$$

where

$$\check{A} = \mathrm{blkdiag}(\check{A}^1, \ldots, \check{A}^n) \tag{4.46a}$$

$$\check{A}^i = -\begin{bmatrix} (u^i_\omega - \hat{\theta}^i)_\times & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{0} & (u^i_\omega - \hat{\theta}^i)_\times & -\boldsymbol{I} & \boldsymbol{0} & \boldsymbol{0} \\ (u^i_a - \hat{\varphi}^i)_\times & \boldsymbol{0} & (u^i_\omega - \hat{\theta}^i)_\times & \boldsymbol{0} & \boldsymbol{I} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \end{bmatrix} \tag{4.46b}$$

$$\check{E} = \sum_{i,l \in \boldsymbol{V} \times \boldsymbol{L}} \check{E}^i_l + \sum_{i,j \in \tilde{\boldsymbol{V}}^2} \check{E}^i_j \tag{4.46c}$$

$$\check{E}^i_l = \mathbb{P}_s\left(F_i(s^i_l)^\top H^i_l(\hat{X})\right) + H^i_l(\hat{X})^\top M^i_l H^i_l(\hat{X}) \tag{4.46d}$$

$$\begin{aligned}\check{E}^i_j = \mathbb{P}_s\left(F_i(s^i_j)^\top H^i_j(\hat{X}) + F_i(s^i_j)^\top R_{ij} L_j - F_j(R_{ij}{}^\top s^i_j)^\top L_j\right) \\ + H^i_j(\hat{X})^\top M^i_j H^i_j(\hat{X})\end{aligned} \tag{4.46e}$$

$$\check{r} = \sum_{i,l \in \boldsymbol{V}\times\boldsymbol{L}} H^i_l(\hat{X})^\top M^i_l(y^i_l - \hat{y}^i_l) + \sum_{i,j\in\tilde{\boldsymbol{V}}^2} H^i_j(\hat{X})^\top M^i_j(y^i_j - \hat{y}^i_j) \tag{4.46f}$$

$$\hat{y}^i_l = h^i_l(\hat{X}) \tag{4.46g}$$

$$\hat{y}^i_j = h^i_j(\hat{X}) \tag{4.46h}$$

$$M^i_l = (D^{-1})^\top Q^i_l D^{-1} \tag{4.46i}$$

$$M^i_j = (D^{-1})^\top Q^i_j D^{-1} \tag{4.46j}$$

$$s^i_j = M^i_j(y^i_j - \hat{y}^i_j) \tag{4.46k}$$

$$F_i(s) = \begin{bmatrix}\bar{F}_i(s,1) & \dots & \bar{F}_i(s,n)\end{bmatrix} \tag{4.46l}$$

$$\bar{F}_i(s,k) = \begin{cases}\begin{bmatrix} s_\times & \mathbf{0}_{3\times12}\end{bmatrix} & k = i \\ \mathbf{0}_{3\times15} & \text{otherwise}\end{cases} \tag{4.46m}$$

$$H^i_l(X) = \begin{bmatrix}\bar{H}^i_l(1) & \dots & \bar{H}^i_l(n)\end{bmatrix} \tag{4.46n}$$

$$\bar{H}^i_l(k) = \begin{cases}\begin{bmatrix} h^i_l(X)_\times & -\boldsymbol{I}_3 & \mathbf{0}_{3\times9}\end{bmatrix} & k = i \\ \mathbf{0}_{3\times15} & \text{otherwise}\end{cases} \tag{4.46o}$$

$$H^i_j(X) = \begin{bmatrix}\bar{H}^i_j(1) & \dots & \bar{H}^i_j(n)\end{bmatrix} \tag{4.46p}$$

$$\bar{H}^i_j(k) = \begin{cases}\begin{bmatrix} h^i_j(X)_\times & -\boldsymbol{I}_3 & \mathbf{0}_{3\times9}\end{bmatrix} & k = i \\ \begin{bmatrix} -R_{ij} m_{j\times} & R_{ij} & \mathbf{0}_{3\times9}\end{bmatrix} & k = j \\ \mathbf{0}_{3\times15} & \text{otherwise}\end{cases} \tag{4.46q}$$

$$L_j := \begin{bmatrix}\bar{L}_j(1) & \dots & \bar{L}_j(n)\end{bmatrix} \tag{4.46r}$$

$$\bar{L}_j(k) := \begin{cases}\begin{bmatrix} -m_{j\times} & \boldsymbol{I}_3 & \mathbf{0}_{3\times9}\end{bmatrix} & k = j \\ \mathbf{0}_{3\times15} & \text{otherwise}\end{cases} \tag{4.46s}$$

4.4 Creating a Distributed Minimum Energy Filter

The remaining question is how to transform the continuous-time filter equations (4.45) into a discrete-time distributed collaborative localisation filter. We will take a similar approach as Section 3.4 to transform the continuous-time filter equations into a discrete-time set of prediction and update steps but with careful consideration for how the resulting equations will be distributed among the nodes.

Firstly, we split the differential equations into two parts, one containing terms relating to the IMU measurements, and the other relating to the external measurements (the landmark and inter-vehicle measurements). Just considering the terms relating to the IMU measurements in (4.45), we have

$$\dot{\hat{X}} = \hat{X}\lambda(\hat{X}, u),$$
$$\dot{\check{K}} = \check{A}\check{K} + \check{K}\check{A}^\top + \check{B}\check{W}^{-1}\check{B}^\top.$$

Recalling the definition of λ from (4.32), it is possible to integrate the state estimate of each vehicle independently as

$$\dot{\hat{X}}^i = \hat{X}^i\lambda(\hat{X}^i, u^i),$$

which has an explicit solution

$$\hat{X}^i(t + \Delta t_u) = \hat{X}^i(t) \cdot \exp\Big(\Delta t_u \cdot \lambda^i(\hat{X}^i(t), u^i)\Big).$$

Here, Δt_u is the sampling period between consecutive IMU measurements, and we are using a sample-and-hold strategy to assume u is constant in the time interval from t to $t + \Delta t_u$.

For $\check{K}$, we will consider a block-wise solution, similarly to how the covariance matrix P in Section 4.2 is decomposed into multiple blocks to enable distribution among the network. Given that W and B have a block-diagonal structure, the on-diagonal blocks of $\check{K}$, represented by $\check{K}^i$, have the following form

$$\dot{\check{K}}^i = \check{A}^i\check{K}^i + \check{K}^i(\check{A}^i)^\top + \check{B}^i(\check{W}^i)^{-1}(\check{B}^i)^\top,$$

while the off-diagonal blocks, represented by $\check{K}^{ij}$, have the form

$$\dot{\check{K}}^{ij} = \check{A}^i\check{K}^{ij} + \check{K}^{ij}(\check{A}^j)^\top.$$

This yields a number of interesting properties. Firstly, observe that the term $\dot{\check{K}}^i$ can be calculated with terms known locally to vehicle i. It is not trivial to find an explicit solution to the differential equation, but it can be approximated by an Euler integration

$$\check{K}^i(t + \Delta t_u) = \check{K}^i(t) + \Delta t_u \left(\check{A}^i\check{K}^i + \check{K}^i(\check{A}^i)^\top + \check{B}^i(\check{W}^i)^{-1}(\check{B}^i)^\top\right)$$

or numerically integrated using another method.

On the other hand, the term $\dot{\check{K}}^{ij}$ requires information known locally by vehicle i as well as vehicle j, however this term does admit an explicit solution, namely

$$\check{K}^{ij}(t + \Delta t_u) = \exp\Big(\check{A}^i(t)\Delta t_u\Big) \cdot \check{K}^{ij}(t) \cdot \exp\Big(\check{A}^j(t)\Delta t_u\Big)^\top.$$

Notice that this equation has a similar form to the prediction step of the partitioned EKF with independent state transition model (4.8d). This means we can apply the same decomposition strategy as Roumeliotis and Bekey [79] did to the EKF by separating the calculation of $\check{K}^{ij}$ into two parts, each calculated on a separate vehicle. On vehicle i, the term $\check{k}^{ij}$ is stored, and is updated according to

$$\check{k}^{ij}(t+\Delta t_u) = \exp\left(\check{A}^i(t)\Delta t_u\right) \cdot \check{k}^{ij}(t)$$

while on vehicle j, the term $\check{k}^{ji}$, with the corresponding update

$$\check{k}^{ji}(t+\Delta t_u) = \exp\left(\check{A}^j(t)\Delta t_u\right) \cdot \check{k}^{ji}(t).$$

Note how, in the absence of any landmark or inter-vehicle measurements, an arbitrary number of consecutive IMU measurements can be recursively incorporated by continuing to left-multiply each term. When an external measurement is made, and the full term K^{ij} is required, it can be reconstructed by communicating between vehicles i and j to calculate

$$\check{K}^{ij} = \check{k}^{ii}(\check{k}^{ji})^\top.$$

Now, we turn our attention back to (4.45a) and (4.45b), considering the terms related to the external measurements. For a single landmark measurement y_l^i, we have

$$\dot{\hat{X}} = \hat{X}\left(\check{K}\check{r}_l^i\right)^\wedge,$$
$$\dot{\check{K}} = -\check{K}\check{E}_l^i\check{K} - \frac{1}{2}\check{\mathrm{ad}}_{Kr_l^i}\check{K} - \frac{1}{2}\check{K}(\check{\mathrm{ad}}_{Kr_l^i})^\top = -\check{K}\check{E}_l^i\check{K} - \mathbb{P}_s\left(\check{\mathrm{ad}}_{Kr_l^i}\check{K}\right).$$

The first equation lends itself to the simple explicit solution of

$$\hat{X}(t^+) = \hat{X}(t)\cdot \mathrm{Exp}\left(\Delta t_l^i \cdot \check{K}(t^+)\check{r}_l^i\right)$$

while an explicit solution for the second equation is not feasible to compute. As in the case of the single-vehicle filter, we use the Euler method to numerically integrate $\dot{\check{K}}$, which gives

$$\begin{aligned}\check{K}(t^+) &= \check{K}(t) + \Delta t_l^i\left(-\check{K}(t)\check{E}_l^i\check{K}(t) - \mathbb{P}_s\left(\check{\mathrm{ad}}_{K(t)r_l^i}\check{K}(t)\right)\right)\\ &= \left(I - \Delta t_l^i\check{K}(t)\check{E}_l^i - \Delta t_l^i\,\mathbb{P}_s\left(\check{\mathrm{ad}}_{K(t)r_l^i}\check{K}(t)\right)\check{K}(t)^{-1}\right)\check{K}(t)\end{aligned}$$

However, in simulations, we find that numerical stability is improved when using the inverse form (derived by numerically integrating $\dot{\check{K}}^{-1}$ and then rearranging in terms of $\check{K}$), which gives

$$\check{K}(t^+) = \left(I + \Delta t_l^i\check{K}(t)\check{E}_l^i + \Delta t_l^i\,\mathbb{P}_s\left(\check{\mathrm{ad}}_{K(t)r_l^i}\check{K}(t)\right)\check{K}(t)^{-1}\right)^{-1}\check{K}(t)$$

The same approach can be applied to the inter-vehicle measurement terms, y_j^i. This gives us a completely distributed minimum energy filter for collaborative localisation;

Algorithm 10: Distributed Minimum Energy Filter for Collaborative Localisation

Predict:
On each vehicle $i \in \boldsymbol{V}$:

$$\hat{X}^i(t + \Delta t_u) = \hat{X}^i(t) \cdot \exp\Big(\Delta t_u \cdot \lambda^i(\hat{X}^i(t), u^i(t))\Big) \tag{4.47a}$$

$$\check{K}^i(t + \Delta t_u) = \check{K}^i(t) + \Delta t_u \left(\check{A}^i\check{K}^i + \check{K}^i(\check{A}^i)^\top + \check{B}^i(\check{W}^i)^{-1}(\check{B}^i)^\top\right) \tag{4.47b}$$

$$\check{k}^{ij}(t + \Delta t_u) = \exp\Big(\check{A}^i(t)\Delta t_u\Big) \cdot \check{k}^{ij}(t) \qquad \forall j \in \boldsymbol{V} \setminus i \tag{4.47c}$$

Update: For a measurement y^i_l

$$\hat{X}(t^+) = \hat{X}(t) \cdot \mathrm{Exp}\left(\Delta t^i_l \cdot \check{K}(t^+)\check{r}^i_l\right) \tag{4.48a}$$

$$\check{K}(t^+) = \Big(I + \Delta t^i_l \check{K}(t)\check{E}^i_l + \underbrace{\Delta t^i_l\, \mathbb{P}_s\big(\check{\mathrm{ad}}_{K(t)r^i_l}\check{K}(t)\big)\check{K}(t)^{-1}}_{\text{Curvature Term}}\Big)^{-1}\check{K}(t) \tag{4.48b}$$

Update: For a measurement y^i_j

$$\hat{X}(t^+) = \hat{X}(t) \cdot \mathrm{Exp}\left(\Delta t^i_j \cdot \check{K}(t^+)\check{r}^i_j\right) \tag{4.49a}$$

$$\check{K}(t^+) = \Big(I + \Delta t^i_j \check{K}(t)\check{E}^i_j + \underbrace{\Delta t^i_j\, \mathbb{P}_s\big(\check{\mathrm{ad}}_{K(t)r^i_j}\check{K}(t)\big)\check{K}(t)^{-1}}_{\text{Curvature Term}}\Big)^{-1}\check{K}(t) \tag{4.49b}$$

where the remaining terms are given in (4.46).

This filter has the same structure as Roumeliotis and Bekey's Distributed EKF described in Section 4.2.1, wherein the prediction step can be performed independently by each vehicle without any communication. As with the Distributed EKF, this filter also requires all-to-all communication at times when landmark or inter-vehicle measurements are made, and so has a similar communication structure to the one shown in Figures 4.3 and 4.4 on Page 65. This shows that it is possible to create a distributed minimum energy filter for collaborative localisation, and Roumeliotis and Bekey's technique is not limited only to Kalman filter-based algorithms.

4.4.1 Extending Beyond the Distributed Filter

While we have shown that we can construct a distributed minimum energy collaborative localisation algorithm, it suffers from the same issue that the distributed EKF does. Namely, it requires all-to-all communication every time a landmark or inter-vehicle measurement is made. In Section 4.2, we have seen how the Schmidt-Kalman filter (SKF) and the Approximated Schmidt-Kalman filter (ASKF) have been applied to negate the need for additional communication by exploiting the particular structure of the EKF filter equations and discarding certain terms. The question that remains is whether these same techniques can be applied to the minimum energy filter we have derived above.

The key to this is in the structure of the update step of the filter. As it stands, the update step of the minimum energy filter is not in a form that allows it to be decomposed into separate updates for each block of $\check{K}$. The main impediment to this is in calculating the curvature terms in (4.48b) and (4.49b), respectively

$$\Delta t_l^i \, \mathbb{P}_s \left(\check{\mathrm{ad}}_{K(t) r_l^i} \check{K}(t) \right) \check{K}(t)^{-1} \qquad \text{and} \qquad \Delta t_j^i \, \mathbb{P}_s \big(\check{\mathrm{ad}}_{K(t) r_j^i} \check{K}(t) \big) \check{K}(t)^{-1},$$

which require inverting the entire $\check{K}$ matrix. Under the distribution scheme from the previous section, this would only be possible if all vehicles communicated the required k terms to a single agent, who would then be able to reconstruct the full $\check{K}$ matrix and perform the inversion. Analysis from simulations, presented in Section 5.2.3, indicate that these terms have negligible impact on filter performance in most situations. Thus, we choose to discard the curvature terms entirely from the update process in order to facilitate the further decoupling of the filter.

Once the terms are discarded, the landmark update step for the $\check{K}$ matrix becomes

$$\check{K}(t^+) = \left(I + \Delta t_l^i \check{K}(t) \check{E}_l^i \right)^{-1} \check{K}(t). \tag{4.50}$$

We can now use the same partitioning strategy that we used for the EKF in Section 4.1.1. For a landmark measurement, y_l^i, we have $\alpha = \{i\}$ and $\beta = \boldsymbol{V} \setminus i$ and $\check{K}$ can be represented as

$$\check{K} = \begin{bmatrix} \check{K}^{\alpha} & \check{K}^{\alpha\beta} \\ \check{K}^{\beta\alpha} & \check{K}^{\beta} \end{bmatrix}$$

The update equation (4.50) can then be represented in the expanded form

$$\check{K}(t^+) = \left(\begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix} + \Delta t_l^i \begin{bmatrix} \check{K}^{\alpha} & \check{K}^{\alpha\beta} \\ \check{K}^{\beta\alpha} & \check{K}^{\beta} \end{bmatrix} \begin{bmatrix} [\check{E}_l^i]^{\alpha} & [\check{E}_l^i]^{\alpha\beta} \\ [\check{E}_l^i]^{\beta\alpha} & [\check{E}_l^i]^{\beta} \end{bmatrix} \right)^{-1} \check{K}$$

By expanding out $\check{E}_l^i$ from Lemma 4.7, we observe that it is sparse, with nonzero entries only in the term $[\check{E}_l^i]^{\alpha}$. This simplifies the update to

$$\begin{aligned} \check{K}(t^+) &= \left(\begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix} + \Delta t_l^i \begin{bmatrix} \check{K}^{\alpha} & \check{K}^{\alpha\beta} \\ \check{K}^{\beta\alpha} & \check{K}^{\beta} \end{bmatrix} \begin{bmatrix} [\check{E}_l^i]^{\alpha} & 0 \\ 0 & 0 \end{bmatrix} \right)^{-1} \check{K} \\ &= \begin{bmatrix} I + \Delta t_l^i \check{K}^{\alpha} [\check{E}_l^i]^{\alpha} & 0 \\ \Delta t_l^i \check{K}^{\beta\alpha} [\check{E}_l^i]^{\alpha} & I \end{bmatrix}^{-1} \check{K} \end{aligned}$$

Using the block matrix inversion formula (A.2), we have

$$\begin{aligned}\check{K}(t^+) &= \begin{bmatrix} (I+\Delta t_l^i \check{K}^\alpha [\check{E}_l^i]^\alpha)^{-1} & 0 \\ -\Delta t_l^i \check{K}^{\beta\alpha}[\check{E}_l^i]^\alpha (I+\Delta t_l^i \check{K}^\alpha [\check{E}_l^i]^\alpha)^{-1} & I \end{bmatrix} \check{K} \\ &= \begin{bmatrix} A\check{K}^\alpha & A\check{K}^{\alpha\beta} \\ \check{K}^{\beta\alpha} - \Delta t_l^i \check{K}^{\beta\alpha}[\check{E}_l^i]^\alpha A\check{K}^\alpha & \check{K}^\beta - \Delta t_l^i \check{K}^{\beta\alpha}[\check{E}_l^i]^\alpha A\check{K}^{\alpha\beta} \end{bmatrix}\end{aligned}$$

where $A = (I+\Delta t_l^i \check{K}^\alpha [\check{E}_l^i]^\alpha)^{-1}$. With the Woodbury matrix identity (A.3), we can show that

$$\check{K}(t^+) = \begin{bmatrix} A\check{K}^\alpha & A\check{K}^{\alpha\beta} \\ (A\check{K}^{\alpha\beta})^\top & \check{K}^\beta - \Delta t_l^i \check{K}^{\beta\alpha}[\check{E}_l^i]^\alpha A\check{K}^{\alpha\beta} \end{bmatrix}.$$

This gives us 3 distinct equations for each of the partitions of $\check{K}$,

$$\begin{aligned}\check{K}^\alpha(t^+) &= (I+\Delta t_l^i \check{K}^\alpha [\check{E}_l^i]^\alpha)^{-1}\check{K}^\alpha(t) \\ \check{K}^{\alpha\beta}(t^+) &= (I+\Delta t_l^i \check{K}^\alpha [\check{E}_l^i]^\alpha)^{-1}\check{K}^{\alpha\beta}(t) \\ \check{K}^\beta(t^+) &= \check{K}^\beta - \Delta t_l^i \check{K}^{\beta\alpha}[\check{E}_l^i]^\alpha (I+\Delta t_l^i \check{K}^\alpha [\check{E}_l^i]^\alpha)^{-1}\check{K}^{\alpha\beta}.\end{aligned}$$

The attentive reader will notice the similarities of these equations compared to the partitioned EKF update equations (4.9) on Page 54. Similarly, the update process for the state estimate can also be partitioned such that,

$$\begin{aligned}\hat{X}^\alpha(t^+) &= \hat{X}^\alpha(t)\cdot \operatorname{Exp}\left(\Delta t_l^i \cdot \check{K}^\alpha(t^+)[\check{r}_l^i]^\alpha\right) \\ \hat{X}^\beta(t^+) &= \hat{X}^\beta(t)\cdot \operatorname{Exp}\left(\Delta t_l^i \cdot \check{K}^{\beta\alpha}(t^+)[\check{r}_l^i]^\alpha\right)\end{aligned}$$

noting that the residual $\check{r}_l^i$ is also sparse, and only the α partition, $[\check{r}_l^i]^\alpha$, is nonzero. The same process also applies to the inter-vehicle measurements, y_j^i, but with $\alpha = \{i, j\}$ and $\beta = \boldsymbol{V} \setminus \alpha$.

The structure of these update equations now allows us to apply the same technique as the Schmidt-Kalman filter to discard the landmark update terms for all vehicles in β. This results in the Schmidt Minimum Energy filter as below.

Algorithm 11: Distributed Schmidt Minimum Energy Filter for CL

Predict: Identical to (4.47)
On each vehicle $i \in \boldsymbol{V}$:

$$\hat{X}^i(t+\Delta t_u) = \hat{X}^i(t)\cdot \exp\left(\Delta t_u \cdot \lambda^i(\hat{X}^i, u^i)\right) \tag{4.51a}$$

$$\check{K}^i(t+\Delta t_u) = \check{K}^i(t) + \Delta t_u \left(\check{A}^i\check{K}^i + \check{K}^i(\check{A}^i)^\top + \check{B}^i(\check{W}^i)^{-1}(\check{B}^i)^\top\right) \tag{4.51b}$$

$$\check{k}^{ij}(t+\Delta t_u) = \exp\left(\check{A}^i(t)\Delta t_u\right)\cdot \check{k}^{ij}(t) \tag{4.51c}$$

Update: For a measurement y^i_l, $\alpha = \{i\}$, $\beta = \boldsymbol{V} \setminus \alpha$

$$\hat{X}^i(t^+) = \hat{X}^i(t) \cdot \operatorname{Exp}\left(\Delta t^i_l \cdot \check{K}^i(t^+)[\check{r}^i_l]^i\right) \tag{4.52a}$$
$$\hat{X}^\beta(t^+) = \hat{X}^\beta(t) \tag{4.52b}$$
$$\check{K}^i(t^+) = (I + \Delta t^i_l \check{K}^i(t)[\check{E}^i_l]^i)^{-1}\check{K}^i(t) \tag{4.52c}$$
$$\check{k}^{ij}(t^+) = (I + \Delta t^i_l \check{K}^i(t)[\check{E}^i_l]^i)^{-1}\check{k}^{ij}(t) \qquad \forall j \in \beta \tag{4.52d}$$
$$\check{K}^\beta(t^+) = \check{K}^\beta(t) \tag{4.52e}$$

Update: For a measurement y^i_j, $\alpha = \{i, j\}$, $\beta = \boldsymbol{V} \setminus \alpha$

$$\hat{X}^i(t^+) = \hat{X}^i(t)\operatorname{Exp}\left(\Delta t^i_j \check{K}^i(t^+)[\check{r}^i_j]^i + \Delta t^i_j \check{K}^{ij}(t^+)[\check{r}^i_j]^j\right) \tag{4.53a}$$
$$\hat{X}^j(t^+) = \hat{X}^i(t)\operatorname{Exp}\left(\Delta t^i_j \check{K}^{ji}(t^+)[\check{r}^i_j]^i + \Delta t^i_j \check{K}^j(t^+)[\check{r}^i_j]^j\right) \tag{4.53b}$$
$$\hat{X}^\beta(t^+) = \hat{X}^\beta(t) \tag{4.53c}$$
$$\check{K}^\alpha(t^+) = (I + \Delta t^i_j \check{K}^\alpha(t)[\check{E}^i_j]^\alpha)^{-1}\check{K}^\alpha(t) \tag{4.53d}$$
$$\check{K}^{\alpha\beta}(t^+) = (I + \Delta t^i_j \check{K}^\alpha(t)[\check{E}^i_j]^\alpha)^{-1}\check{K}^{\alpha\beta}(t) \tag{4.53e}$$
$$\check{K}^\beta(t^+) = \check{K}^\beta(t) \tag{4.53f}$$

See (4.46) for the remaining definitions.

This shows that we can build a filter similar to the Schmidt-Kalman filter, but using the minimum energy filtering framework instead. The topology of the communication requirements will be the same as that of the SKF, which is shown in Figures 4.5 and 4.6 on Page 66, albeit with different information being stored and transmitted.

As is true for the SKF, it is possible to perform the landmark measurement update step of the Schmidt minimum energy filter without any communication between vehicles, as $|\alpha| = 1$. But, again, it is not possible to perform the inter-vehicle measurement update step without communicating additional terms. We suffer from exactly the same problem, in that the update step for the terms in $\check{K}^{\alpha\beta}$ can not be written as a simple recursive pre-multiplication.

Recall the inter-vehicle measurement update step for $\check{K}^{\alpha\beta}$ (4.53e),

$$\check{K}^{\alpha\beta}(t^+) = (I + \Delta t^i_j \check{K}^\alpha(t)[\check{E}^i_j]^\alpha)^{-1}\check{K}^{\alpha\beta}(t).$$

The term $(I + \Delta t^i_j \check{K}^\alpha(t)[\check{E}^i_j]^\alpha)^{-1}$ can be computed using the information known locally to vehicles i and j. Substituting to this term as Ξ, the above equation simplifies to

$$\check{K}^{\alpha\beta}(t^+) = \Xi\check{K}^{\alpha\beta}(t).$$

Expanding out the set $\alpha = \{i, j\}$, and $\beta = \{\beta_1, \beta_2, \dots\}$ we have

$$\check{K}^{\alpha\beta}(t^+) = \begin{bmatrix} \Xi^i & \Xi^{ij} \\ \Xi^{ji} & \Xi^j \end{bmatrix} \begin{bmatrix} \check{K}^{i\beta_1} & \check{K}^{i\beta_2} & \dots \\ \check{K}^{j\beta_1} & \check{K}^{j\beta_2} & \dots \end{bmatrix}.$$

Considering a single element, $\beta_1 \in \beta$, we have the two update equations

$$\begin{aligned} \check{K}^{i\beta_1}(t^+) &= \Xi^i \check{K}^{i\beta_1}(t) + \Xi^{ij} \check{K}^{j\beta_1}(t) \\ \check{K}^{j\beta_1}(t^+) &= \Xi^{ji} \check{K}^{i\beta_1}(t) + \Xi^j \check{K}^{j\beta_1}(t) \end{aligned}$$

As we discussed in Section 4.2.3, the fact that these two equations can't be written as a single recursive pre-multiplication means that communication between all vehicles is required to reconstruct the terms $\check{K}^{i\beta_1}$ and $\check{K}^{j\beta_1}$.

If we follow the same line of reasoning from Luft, Schubert *et al.* [98] as we did in Section 4.2.3, then we can approximate terms in the $\check{K}$ matrix by

$$\check{K}^{jk} \approx \check{K}^{ji} (\check{K}^i)^{-1} \check{K}^{ik}$$

for every $i, j, k \in \boldsymbol{V}$. Taking the term $\check{K}^{i\beta_1}$ as an example, this leads to the approximated version of the update equation

$$\begin{aligned} \check{K}^{i\beta_1}(t^+) &= \Xi^i \check{K}^{i\beta_1}(t) + \Xi^{ij} \check{K}^{j\beta_1}(t) \\ &\approx \Xi^i \check{K}^{i\beta_1}(t) + \Xi^{ij} \check{K}^{ji} \check{K}^i(t)^{-1} \check{K}^{i\beta_1}(t) \\ &= (\Xi^i \check{K}^i + \Xi^{ij} \check{K}^{ji}) \check{K}^i(t)^{-1} \check{K}^{i\beta_1}(t) \\ &= \check{K}^i(t^+) \check{K}^i(t)^{-1} \check{K}^{i\beta_1}(t) \end{aligned}$$

Applying this same approximation to all other elements of $\check{K}^{\alpha\beta}$ results in

$$\check{K}^{\alpha\beta}(t^+) \approx \begin{bmatrix} \check{K}^i(t^+) \check{K}^i(t)^{-1} & 0 \\ 0 & \check{K}^j(t^+) \check{K}^j(t)^{-1} \end{bmatrix} \begin{bmatrix} \check{K}^{i\beta_1} & \check{K}^{i\beta_2} & \dots \\ \check{K}^{j\beta_1} & \check{K}^{j\beta_2} & \dots \end{bmatrix}.$$

This approximated update is of the same form as the ASKF, which means that we can perform the inter-vehicle update step with only communication between vehicles i and j. This gives our final filter, the approximate Schmidt minimum energy filter;

Algorithm 12: Distributed Approximate Schmidt Minimum Energy Filter for CL

Predict: Identical to (4.47)

On each vehicle $i \in \boldsymbol{V}$:

$$\hat{X}^i(t+\Delta t_u) = \hat{X}^i(t) \cdot \exp\Big(\Delta t_u \cdot \lambda^i(\hat{X}^i, u^i)\Big) \tag{4.54a}$$

$$\check{K}^i(t+\Delta t_u) = \check{K}^i(t) + \Delta t_u \left(\check{A}^i\check{K}^i + \check{K}^i(\check{A}^i)^\top + \check{B}^i(\check{W}^i)^{-1}(\check{B}^i)^\top\right) \tag{4.54b}$$

$$k^{ij}(t+\Delta t_u) = \exp\Big(\check{A}^i(t)\Delta t_u\Big) \cdot \check{k}^{ij}(t) \tag{4.54c}$$

Update: For a measurement y^i_l, $\alpha = \{i\}$, $\beta = \boldsymbol{V} \setminus \alpha$

$$\hat{X}^i(t^+) = \hat{X}^i(t) \cdot \mathrm{Exp}\left(\Delta t^i_l \cdot \check{K}^i(t^+)[\check{r}^i_l]^i\right) \tag{4.55a}$$

$$\hat{X}^\beta(t^+) = \hat{X}^\beta(t) \tag{4.55b}$$

$$\check{K}^i(t^+) = (I + \Delta t^i_l \check{K}^i(t)[\check{E}^i_l]^i)^{-1}\check{K}^i(t) \tag{4.55c}$$

$$k^{ij}(t^+) = (I + \Delta t^i_l \check{K}^i(t)[\check{E}^i_l]^i)^{-1}k^{ij}(t) \qquad \forall j \in \beta \tag{4.55d}$$

$$\check{K}^\beta(t^+) = \check{K}^\beta(t) \tag{4.55e}$$

Update: For a measurement y^i_j, $\alpha = \{i, j\}$, $\beta = \boldsymbol{V} \setminus \alpha$

$$\hat{X}^i(t^+) = \hat{X}^i(t)\,\mathrm{Exp}\left(\Delta t^i_j \check{K}^i(t^+)[\check{r}^i_j]^i + \Delta t^i_j \check{K}^{ij}(t^+)[\check{r}^i_j]^j\right) \tag{4.56a}$$

$$\hat{X}^j(t^+) = \hat{X}^i(t)\,\mathrm{Exp}\left(\Delta t^i_j \check{K}^{ji}(t^+)[\check{r}^i_j]^i + \Delta t^i_j \check{K}^j(t^+)[\check{r}^i_j]^j\right) \tag{4.56b}$$

$$\hat{X}^\beta(t^+) = \hat{X}^\beta(t) \tag{4.56c}$$

$$\check{K}^\alpha(t^+) = (I + \Delta t^i_j \check{K}^\alpha(t)[\check{E}^i_j]^\alpha)^{-1}\check{K}^\alpha(t) \tag{4.56d}$$

$$k^{ik}(t^+) = \check{K}^i(t^+)\check{K}^i(t)^{-1}k^{ik}(t) \tag{4.56e}$$

$$k^{jk}(t^+) = \check{K}^j(t^+)\check{K}^j(t)^{-1}k^{jk}(t) \tag{4.56f}$$

$$\check{K}^\beta(t^+) = \check{K}^\beta(t) \tag{4.56g}$$

See (4.46) for the remaining definitions.

4.5 Possible extensions to the ASKF

As an aside, we undertake a brief investigation into possible extensions to the Approximate Schmidt-Kalman filter. Both the SKF and the ASKF trade-off filter performance with communication requirements. The ASKF filter only requires pairwise communication between nodes when a relative measurement is made between them. This enables very sparse and infrequent communications within the network. However, there may be times when nodes have connectivity and spare bandwidth to communicate. Given this, we explore if it is possible to improve performance beyond the ASKF approach by taking advantage of additional optional communication.

Consider a partitioned update, similar to the update described above but with α containing an arbitrary number of nodes and where communication is freely available between all nodes in α. In this general case, the update to the covariance term $P^{\alpha\beta}$ is given by (4.12d), which is

$$P_+^{\alpha\beta} = (I - K^\alpha H^\alpha) P_-^{\alpha\beta} \tag{4.57}$$

$$= \begin{bmatrix} I - K^{\alpha_1} H^{\alpha 1} & -K^{\alpha_1} H^{\alpha_2} & \cdots \\ -K^{\alpha_2} H^{\alpha 1} & I - K^{\alpha_2} H^{\alpha_2} & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix} \begin{bmatrix} P_-^{\alpha_1\beta_1} & P_-^{\alpha_1\beta_2} & \cdots \\ P_-^{\alpha_2\beta_1} & P_-^{\alpha_2\beta_2} & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix} \tag{4.58}$$

Under the same scheme as described in Section 4.2.1, where the covariance, P is decomposed into components, p, the key to performing this update without needing to communicate with any node in β is to find a block diagonal matrix, A, such that

$$P_+^{\alpha\beta} \approx A P_-^{\alpha\beta}. \tag{4.59}$$

Note however, that this is not the same problem as finding a block-diagonal approximation for the term $(I - K^\alpha H^\alpha)$, as $P_-^{\alpha\beta}$ is not an arbitrary matrix and may have additional properties which can be exploited. This is how Luft, Schubert *et al.* derives the approximation for the ASKF — by taking advantage of relationships between elements of the covariance matrix (namely through the conditional covariance and the Schur complement) — not just by finding an approximation for A in isolation.

If we use the same approximation presented in Luft, the natural extension of the approximation becomes

$$A = \begin{bmatrix} P_+^{\alpha_1\alpha_1}(P_-^{\alpha_1\alpha_1})^{-1} & 0 & 0 & \cdots \\ 0 & P_+^{\alpha_2\alpha_2}(P_-^{\alpha_2\alpha_2})^{-1} & 0 & \cdots \\ 0 & 0 & P_+^{\alpha_3\alpha_3}(P_-^{\alpha_3\alpha_3})^{-1} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} \tag{4.60}$$

However, we find that the performance of the filter using this approximation is poor. One would expect that, as the size of α increases, the performance would start at that of the ASKF and improve towards that of the EKF (The EKF is equivalent to the SKF/ASKF with $\alpha = N, \beta = \emptyset$). In simulations on the MRCLAM dataset, we observe that the performance of the proposed filter generally behaves as expected. However, there are a number of situations in which the proposed filter performs worse than both the ASKF and the EKF. It seems unintuitive that allowing additional communication between nodes results in worse performance than the ASKF, and it suggests that the approximation made does not extend well to instances where $|\alpha| > 2$.

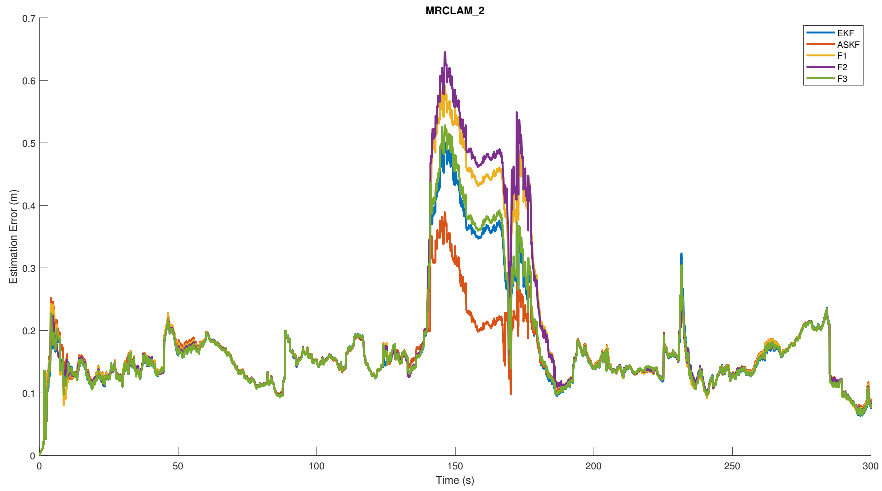


FIGURE 4.9: Position estimate error of filters run on a sample from MRCLAM dataset [101]. F1, F2, F3 represent the proposed filter based on (4.60) where α contains an additional 1,2, and 3 randomly selected nodes above the minimal β-independent set. For context, F0 would be equivalent to the ASKF and F4 would be equivalent to the EKF

An example simulation is shown in Figure 4.9, which compares the ASKF, the EKF, and the proposed extension with various sizes of α. We observe at approximately 140s, the error of all estimates increases, possibly due to erroneous measurement data, however, the F1, F2, and F3 filters all perform worse than both the EKF and the ASKF during this period. Interestingly, the ASKF performs better than the EKF in this situation, and this is speculated to be because the erroneous measurement updates do not propagate as widely in the ASKF, and the estimate is not as strongly affected. The fact that this behaviour doesn't result in similar performance for the proposed filters suggests that the approximation errors play a significant role.

Based on these results, we considered a different approach to applying the approximation (4.23). Rather than consider i as a single node in N, we use a set of nodes. Recalling again the first element of $P_+^{\alpha\beta}$, we have

$$P_+^{\alpha_1\beta_1} = (I - K^{\alpha_1}H^{\alpha_1})P_-^{\alpha_1\beta_1} - K^{\alpha_1}H^{\alpha_2}P_-^{\alpha_2\beta_1} \tag{4.61}$$

in which we make an approximation of the term $P_-^{\alpha_2\beta_1}$. Instead of approximating with $i = \alpha_1$, we make the approximation with $i = \{\alpha_1, \alpha_3\}$. The reasoning is that by incorporating more information into the approximation, it should yield more accurate results. Applying the approximation from (4.23) in this context gives

$$P_-^{\alpha_2\beta_1} \approx \begin{bmatrix} P_-^{\alpha_2\alpha_1} & P_-^{\alpha_2\alpha_3} \end{bmatrix} \begin{bmatrix} P_-^{\alpha_1\alpha_1} & P_-^{\alpha_1\alpha_3} \\ P_-^{\alpha_3\alpha_1} & P_-^{\alpha_3\alpha_3} \end{bmatrix}^{-1} \begin{bmatrix} P_-^{\alpha_1\beta_1} \\ P_-^{\alpha_3\beta_1} \end{bmatrix}. \tag{4.62}$$

While this value can be calculated with the information available on the α nodes, the decomposition strategy only works if $P_+^{\alpha_1\beta_1}$ can be expressed as a single pre-multiplication of $P_-^{\alpha_1\beta_1}$. In order to do this, we must further approximate

$$P_-^{\alpha_3\beta_1} \approx P_-^{\alpha_3\alpha_1}(P_-^{\alpha_1\alpha_1})^{-1}P_-^{\alpha_1\beta_1}. \tag{4.63}$$

Substituting this approximation into the above equation, and performing the block matrix inversion, we find that many of the terms cancel out, which gives the final result of

$$P_-^{\alpha_2\beta_1} \approx P_-^{\alpha_2\alpha_1}(P_-^{\alpha_1\alpha_1})^{-1}P_-^{\alpha_1\beta_1}, \tag{4.64}$$

which is no different to the original approximation (4.23) with $i = \alpha_1$.

4.6 Summary

This chapter has covered a wide range of material related to collaborative localisation. We have shown in detail the exact process that is used to transform the centralised extended Kalman filter to the distributed version. We then showed how the different forms of collaborative filters, namely the Schmidt Kalman filter and the Approximate Schmidt Kalman filter build on the EKF structure, and how they can reduce communication overhead at the expense of introducing approximations. Given that the aforementioned filters are all based on the EKF, we explored whether the same techniques could also be applied to more recent filter designs, such as the minimum energy filter. We showed that, under a particular choice of discretisation, and by disregarding a minor term, we can formulate a minimum energy filter with a compatible structure that allows the filter equations to be distributed among the network nodes. Further, we demonstrated that the approximations that lead to the SKF and the ASKF can also be applied to the minimum energy filter.

This work provides a new perspective on top-down collaborative localisation algorithms, providing a better understanding of the techniques for transforming centralised filters into decentralised ones, and demonstrating the wider applicability of these techniques outside the standard EKF. In the following chapters, we demonstrate the application of these different filter designs through the use of simulation and real-world data.

Chapter 5

Filter Demonstration in Simulation

In this chapter, we demonstrate how the algorithms derived in the previous chapter can be implemented on a computer, give some insight into their performance through the use of a simulation, and explore the trade-offs between communication overhead and filtering performance. We use data from the EuRoC MAV dataset [105] to construct a set of different collaborative localisation scenarios and run the newly-proposed minimum energy filtering algorithms on these scenarios. By analysing the filter estimates and comparing them to the ground-truth data, we can evaluate the performance of the different filtering algorithms and examine their behaviour under different conditions. Initially, we start with a baseline scenario to demonstrate that the implementation of the algorithms is correct and that the behaviour is as expected. Then we introduce two more challenging scenarios with limited sensor information to demonstrate the advantages of using collaborative localisation over independent localisation. This also allows us to investigate the performance penalty that is incurred when using reduced communication algorithms, including the distributed Schmidt filter and the distributed approximate Schmidt filter.

It is important to clarify that it is not the goal of this chapter to make statements about the absolute performance of the minimum-energy filter or provide comparisons to other filters in the literature. This is for several reasons. Firstly, the author is not aware of an existing comparable filter in the literature for the distributed inertial collaborative localisation problem. The closest example of such a filter is found in Jung, Brommer and Weiss [100], however, there are a number of differences in the sensor models that make this filter incompatible with our work. Furthermore, any discussion on performance comparisons between filters inevitably leads to discussions on filter tuning and other implementation details such as numerical stability, with each algorithm requiring detailed knowledge in order to extract the optimum performance. While there is a clear need for such work, this goes beyond the scope of this thesis.

5.1 Constructing a Semi-Synthetic dataset

As discussed in Chapter 2, one of the challenges in evaluating collaborative localisation algorithms is the lack of publicly available datasets on which to run the algorithms. The Multi-Robot Collaborative Localisation and Mapping

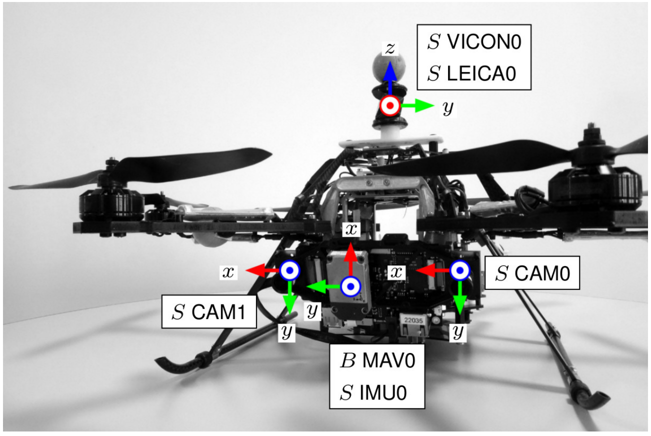


FIGURE 5.1: The vehicle used for data collection in the EuRoC dataset. [105]

(MRCLAM) dataset [101] is the most popular of these, but it only consists of ground-based robots with SE(2) pose information and wheel odometry, making it unsuitable for evaluating SE(3) collaborative localisation with IMU measurements. The creation of a real-world experimental dataset for inertial collaborative localisation on SE(3) is considered in the proceeding chapters of this thesis. In the meantime, we turn our attention to constructing a synthetic dataset in order to provide some initial insights into the filter design and performance.

Despite a lack of multi-vehicle collaborative localisation datasets, there are many high-quality single-vehicle datasets with SE(3) pose information and inertial measurement data, including the European Robotics Challenge (EuRoC) dataset [105], and Technical University of Munich Visual-Inertial (TUM VI) dataset [106]. Rather than constructing a synthetic dataset from scratch, we can reuse elements of existing datasets to create a more realistic semi-synthetic dataset. We will use the EuRoC dataset, as it is well-documented and popular in the visual odometry community.

The EuRoC dataset contains several sequences, each of a single UAV, pictured in Figure 5.1, flying in an indoor environment with recordings of IMU data and stereo video imagery. For 6 of the datasets, ground truth pose information is recorded using a motion capture system, and is time-aligned to the IMU data. For our purposes, we are not interested in the video imagery recorded by the UAV, but the IMU and ground truth pose information provides a collection of realistic vehicle trajectories as well as real-world IMU sensor data, capturing a number of real-world effects including biases and vibrations.

To construct a multi-vehicle collaborative localisation dataset, we adjust the timestamps of each of the 6 EuRoC sequences so that they have a common start time as if they were all flying at the same time. Each sequence is a different length, and so to ensure there is complete data for all vehicles, each sequence is trimmed to the length of the shortest sequence, which is approximately 83 seconds. A plot of the vehicle trajectories is shown in Figure 5.2, which illustrates the extent of the operational space, as well as the complexity of the individual trajectories.

As the EuRoC dataset was originally intended for visual-inertial odometry, the data provided by the EuRoC sequences only contains ground-truth pose, IMU measurements and vision data. Our filter algorithms were designed around the use of relative position measurement to known landmarks and inter-vehicle relative position measurements. Thus, we add a collection of 'virtual' landmarks and generate synthetic relative measurements between the vehicles and the landmarks. We use the provided ground truth poses of the vehicles to determine the true relative position and add Gaussian noise to generate the synthetic measurement. We also perform a similar process to generate synthetic inter-vehicle measurements.

To model interruptions to the relative position measurements, for example, due to range limitations or sensor interference, we define the time-varying visibility matrices. We use $\Theta_{\boldsymbol{L}} : \mathbb{R}^+ \to \{0,1\}^{|\boldsymbol{V}|\times|\boldsymbol{L}|}$ and $\Theta_{\boldsymbol{V}} : \mathbb{R}^+ \to \{0,1\}^{|\boldsymbol{V}|\times|\boldsymbol{V}|}$ to represent the landmark and inter-vehicle visibility respectively (recalling the definitions of $\boldsymbol{L}$ and $\boldsymbol{V}$ from Section 4.3.1). Each entry in the visibility matrix $\Theta_{\boldsymbol{L}}(t)$ corresponds to whether a vehicle can make a relative position measurement of a landmark at time t. Similarly, each entry in $\Theta_{\boldsymbol{V}}(t)$ represents whether a vehicle can make an inter-vehicle relative position measurement of another vehicle in the network at time t.

For the simulation results presented below, we added 4 virtual landmarks, located at the following points

$$p_1 = \begin{bmatrix} 3 \\ 3 \\ 0 \end{bmatrix} \qquad p_2 = \begin{bmatrix} -3 \\ 3 \\ 0 \end{bmatrix} \qquad p_3 = \begin{bmatrix} 2 \\ -2 \\ 3 \end{bmatrix} \qquad p_4 = \begin{bmatrix} 0 \\ 0 \\ 5 \end{bmatrix}$$

in metres with respect to the global reference frame. If a given landmark or vehicle is visible, the sensor produces measurements at a rate of 10 Hz with additive measurement error sampled from a normal distribution, $\mathcal{N}(0, 0.5)$.

The resulting dataset, which we will refer to as the EuRoC CL dataset, is a 6-vehicle collaborative localisation dataset using realistic trajectories and IMU sensor data, with customisable landmark and inter-vehicle measurements.

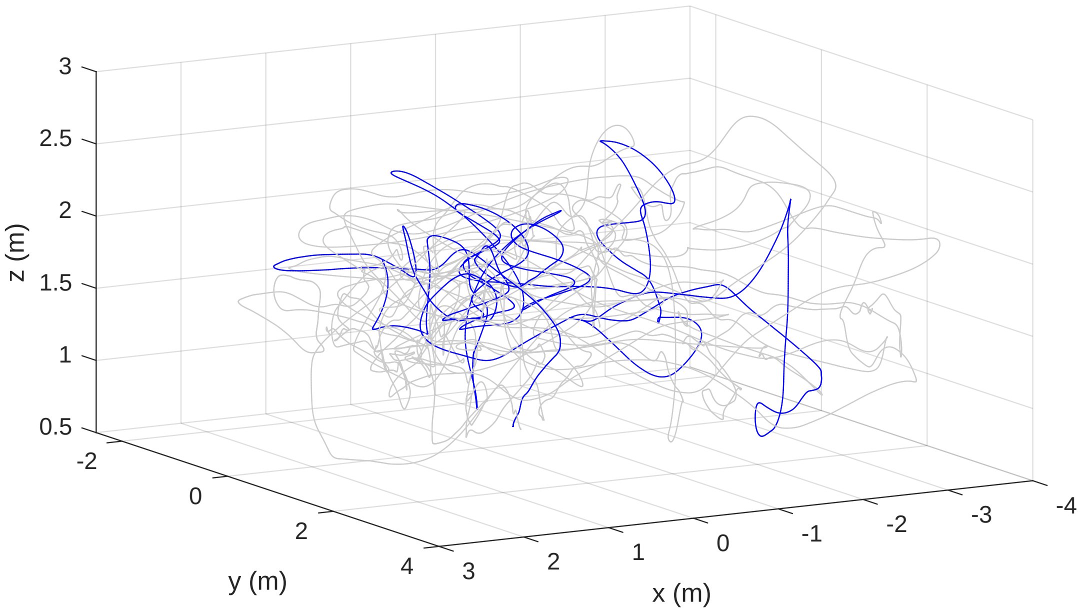


FIGURE 5.2: Illustration of the 6 different vehicle trajectories in the semi-synthetic EuRoC CL dataset. An example trajectory of a single vehicle has been shown in blue, while all others are shown in grey.

5.2 Simulation 1: Baseline Simulation

The first simulation represents a baseline scenario that contains an abundance of sensor measurements which should yield accurate, stable position estimates. The purpose of the baseline simulation is to demonstrate the correct implementation of the filter algorithms and establish an expected level of performance under ideal conditions. We use the EuRoC CL dataset and set the visibility matrices to

$$\Theta_V(t) \equiv \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 \end{bmatrix}, \qquad \Theta_L(t) \equiv \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

indicating that all vehicles make measurements of all 4 landmarks and all other vehicles in the network at a constant rate of 10 Hz.

5.2.1 Filter Tuning Parameters

Before we can test the different minimum-energy collaborative localisation algorithms, including the one detailed in Algorithm 10, several tuning parameters must be specified, including $\check{B}$, $\check{D}$, $\check{W}$, $\check{Q}$, as well as the cost on the initial state, J_0. As discussed in Section 3.4.1, tuning can be a long and complex process. Consequently, an in-depth analysis of the optimal tuning parameters

TABLE 5.1: Parameter values used in the simulations for the Distributed Minimum Energy Filter.

Symbol	Value
$\check{W}^i$	$\frac{1}{\Delta t_{u^i}} \boldsymbol{I}$
B^i_ω	$1.0 \times 10^{-2} \boldsymbol{I}$
B^i_a	$2.0 \times 10^{-3} \boldsymbol{I}$
B^i_θ	$1.94 \times 10^{-5} \boldsymbol{I}$
B^i_ϕ	$3.0 \times 10^{-3} \boldsymbol{I}$
$\check{Q}^i_l$	$\frac{1}{\Delta t^i_l} \boldsymbol{I}$
$\check{Q}^i_j$	$\frac{1}{\Delta t^i_j} \boldsymbol{I}$
$\check{D}^i_l$	$\sqrt{0.5} \boldsymbol{I}$
$\check{D}^i_j$	$\sqrt{0.5} \boldsymbol{I}$

of the minimum energy filter is beyond the scope of this work. Instead, we select the parameters based on measured sensor properties and empirical observations which we find provide satisfactory, but admittedly non-optimal, results. As we are more concerned with evaluating the relative performance between different collaborative localisation algorithms under the same tuning parameters this does not present an impediment to the analysis.

The EuRoC dataset contains reported noise parameters of the IMU which were determined by recording samples from the IMU at rest. However, as acknowledged in the report [105], this may not account for dynamic effects that may introduce additional noise into IMU measurements during flight. We used these reported values for the corresponding components of the B matrix as a starting point but found it necessary to increase some of these values to provide satisfactory results. For the D matrix, we use the known value of the synthetic additive measurement noise, which is 0.5 m. As proposed in Section 3.4.1, we use a value of $\frac{1}{\Delta t}$ for the corresponding values of the W and Q matrices. The values selected for each parameter are shown in Table 5.1.

The cost on the initial state, J_0, in the cost functional (4.37) is implicitly defined by initialising the state estimate $\hat{X}$ and the $\check{K}$ matrix to some value. For each vehicle $i \in \boldsymbol{V}$, state estimate is initialised to

$$\hat{R}^i(0) = \mathrm{Exp}(\xi^i_R) R^i(0) \qquad \xi^i_R \sim \mathcal{N}(\mathbf{0}, 0.3\boldsymbol{I}) \tag{5.1a}$$
$$\hat{x}^i(0) = x^i(0) + \xi^i_x \qquad \xi^i_x \sim \mathcal{N}(\mathbf{0}, 2\boldsymbol{I}) \tag{5.1b}$$
$$\hat{v}^i(0) = \mathbf{0} \tag{5.1c}$$
$$\hat{\theta}^i(0) = \mathbf{0} \tag{5.1d}$$
$$\hat{\phi}^i(0) = \mathbf{0} \tag{5.1e}$$

indicating that the pose estimate is initialised to a random point near the true pose, and the remainder of the state variables are set to zero. For each vehicle $i \in \boldsymbol{V}$, the $\check{K}^i$ matrix is initialised to[1]

$$\check{K}^i(0) = \text{blkdiag}(5\boldsymbol{I}_3, 0.01\boldsymbol{I}_3, 3\boldsymbol{I}_3, \boldsymbol{I}_3, \boldsymbol{I}_3) \tag{5.2}$$

and the complete $\check{K}$ matrix is initialised as

$$\check{K}(0) = \text{blkdiag}(\check{K}^0(0), \check{K}^1(0), \ldots, \check{K}^6(0)). \tag{5.3}$$

We will use the parameter values specified in this section for all simulations and filter algorithms unless otherwise stated.

5.2.2 Performance of Distributed Minimum Energy Filter

Using the baseline scenario, we will first demonstrate the performance of the Distributed Minimum Energy Filter (D-MEF) described in Algorithm 10 on Page 82. Using the reported ground-truth data reported in the EuRoC dataset, we can compare the state estimate generated by the filter with the true state of each vehicle. The ground-truth data includes not only the pose of the vehicle but also the linear velocity, and the IMU biases. However, because the ground-truth data was collected from a VICON motion capture system which only records an object's pose at discrete points in time, the vehicle's velocity is determined by interpolating between poses. Similarly, the IMU biases are not directly measurable and are inferred by comparing the IMU measurements against the true acceleration and angular velocities interpolated from the ground truth and then smoothing over the entire trajectory. Thus, one must be careful about drawing conclusions as to the accuracy of particular state estimates of these variables.

The Distributed Minimum Energy Filter was run on Simulation 1 and the results are shown in Figures 5.3 and 5.4 and summary statistics are shown in Table 5.2. Figure 5.3 shows the error between the filter state estimates and the ground truth as a function of time for the vehicle states of position, orientation, and velocity. The mean error value over all 6 vehicles is shown as the solid red line, while the range between the vehicle with the highest and lowest estimate error is shown as the shaded region. Taken together, the three sub-figures show the filter converges rapidly from the initial estimate to track the vehicles' poses and velocities accurately. The filter continues to provide accurate estimates and does not diverge the full duration of the scenario, demonstrating that the filter can remain stable over long periods. Even with only a basic tuning procedure, the mean position error over all vehicles is 13 cm, and the shaded regions in Figure 5.3 show that there are no significant outliers in the individual vehicle estimates.

Figure 5.4 shows the estimation errors over time for the IMU bias states. Here, we show each of the individual vehicle errors as separate lines rather than just the range. From the figure we can observe that the bias estimates show

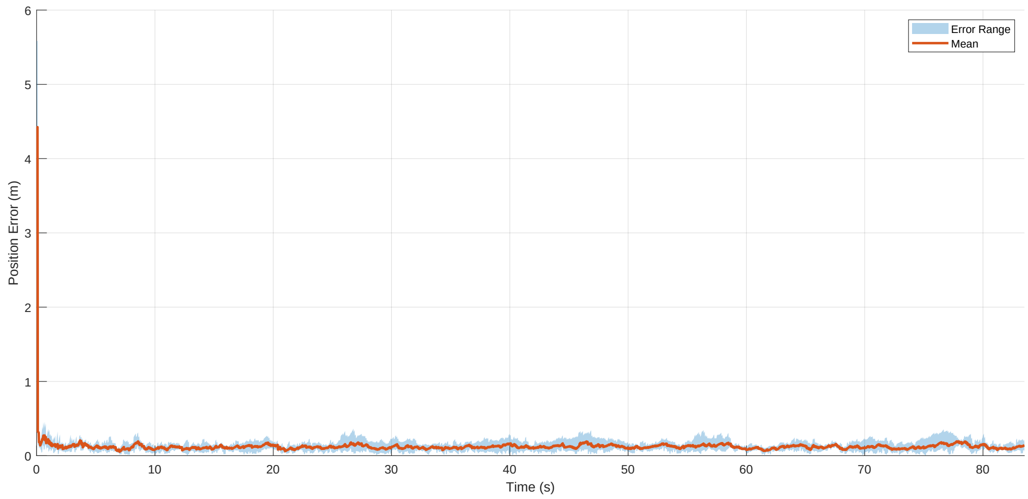
[1] See Appendix A.1.2 for definition of blkdiag.

TABLE 5.2: Mean estimation error of Distributed Minimum Energy Filter on baseline simulation.

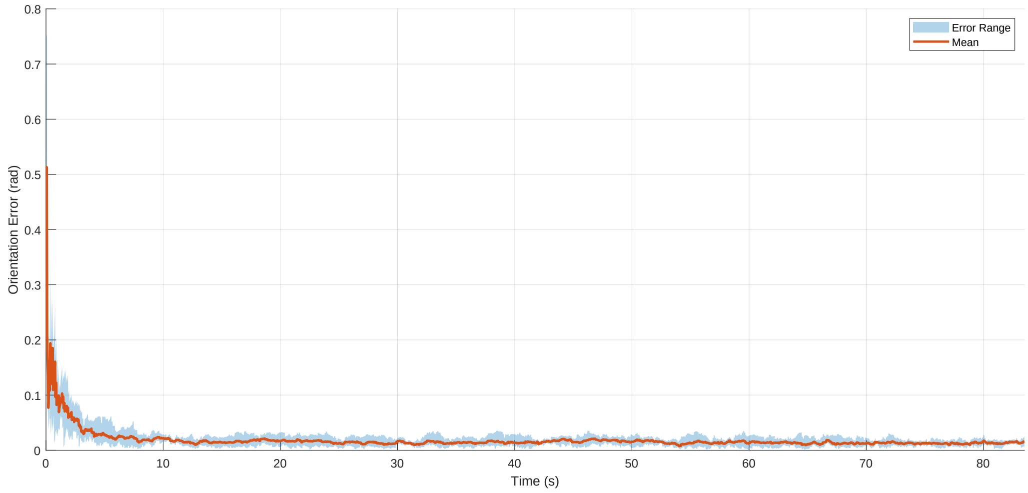
	Position m	Orientation rad	Velocity $\mathrm{m\,s^{-1}}$	Gyro Bias $\mathrm{rad\,s^{-1}}$	Accel. Bias $\mathrm{m\,s^{-2}}$
Vehicle 1	0.140	0.026	0.152	0.016	0.148
Vehicle 2	0.125	0.020	0.135	0.017	0.073
Vehicle 3	0.118	0.019	0.135	0.016	0.065
Vehicle 4	0.142	0.024	0.153	0.011	0.132
Vehicle 5	0.140	0.025	0.153	0.020	0.146
Vehicle 6	0.133	0.019	0.150	0.010	0.058
Mean	**0.133**	**0.022**	**0.146**	**0.015**	**0.104**

an initial increase in error, before slowly converging. This is likely due to the initial estimation errors in the other components of the state resulting in erroneous updates to the bias terms. For the IMU angular bias, shown in Figure 5.4a, all the vehicles have similar error trajectories, however, for the IMU linear bias shown in Figure 5.4b (i.e. the bias in the accelerometer), the error trajectory for each vehicle is significantly different. In some cases, there is more than an order of magnitude difference between the vehicles with the lowest and highest estimate errors. Given that the scenario dataset was constructed by merging 6 individual EuRoC sequences into a collaborative dataset, and that the vehicle used in each of those individual sequences was the same, one would expect broadly similar performance from each of the vehicles. The fact that this difference is significant suggests that there is some long-term variation of the sensor properties, perhaps due to environmental fluctuations that occurred between the recording of each sequence. More complex IMU models, such as the one proposed by Ramalingam, Anitha and Shanmugam [121], incorporate additional effects such as scale factors and temperature-based variations. This may assist in reducing estimation errors but comes at the cost of introducing additional states into the system, further complicating the algorithm. As discussed above, the ground-truth values for the IMU biases were not directly measured but were estimated from available data, making it difficult to determine where the true source of error lies.

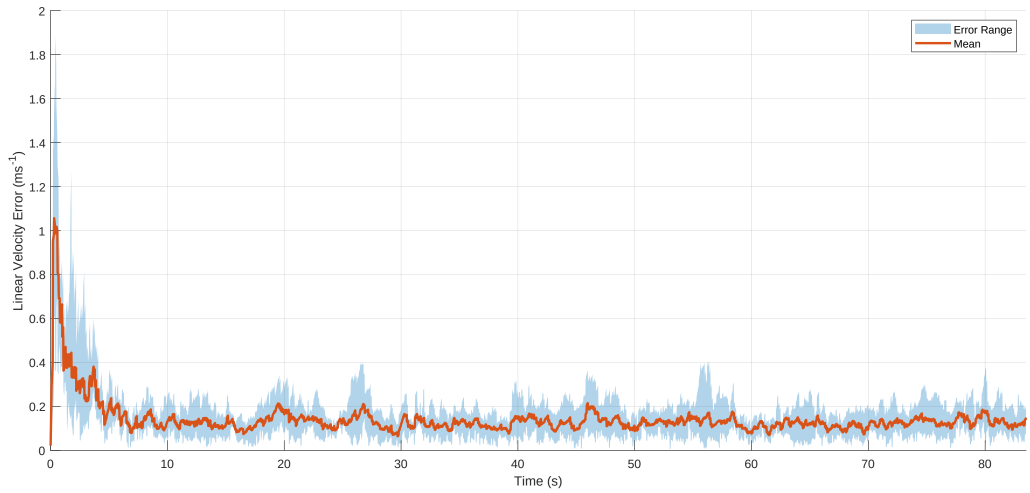
Overall, the initial results of the Distributed Minimum Energy filter (D-MEF) are promising. We have shown that all the filter state estimates converge to within a small tolerance of their respective ground-truth values, and remain stable for the remainder of the simulation. While a single simulation can not prove or disprove particular properties of the filter, the strength of these initial results adds significant weight to the argument that the system model accurately represents the true system, the filter derivation does not contain errors, the discretisation approach was sound, and software implementation is correct. One should not underestimate the number of different ways throughout the entire process in which errors or mistakes may be introduced which would significantly degrade filter performance or result in complete failure of the filter.



(A) Position Error

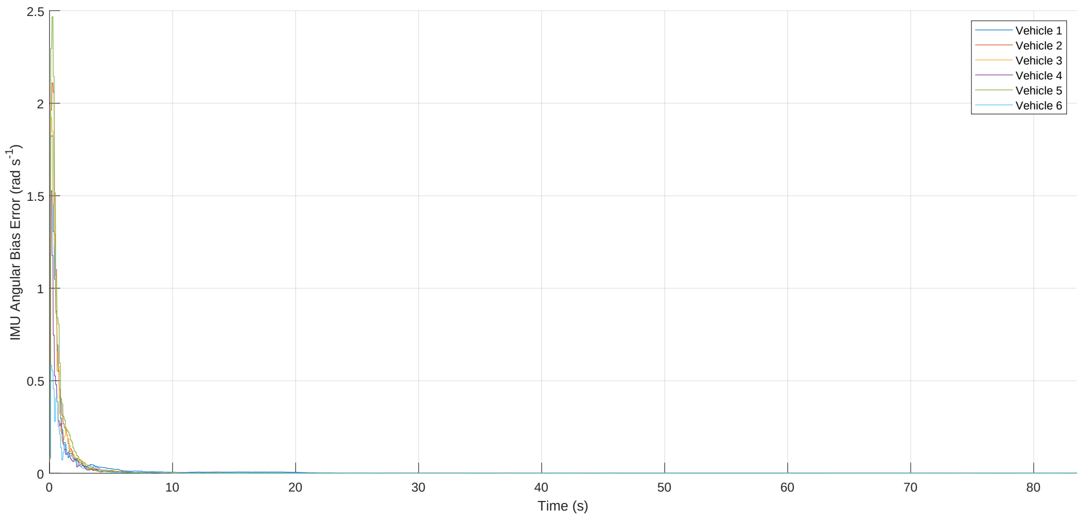


(B) Orientation Error

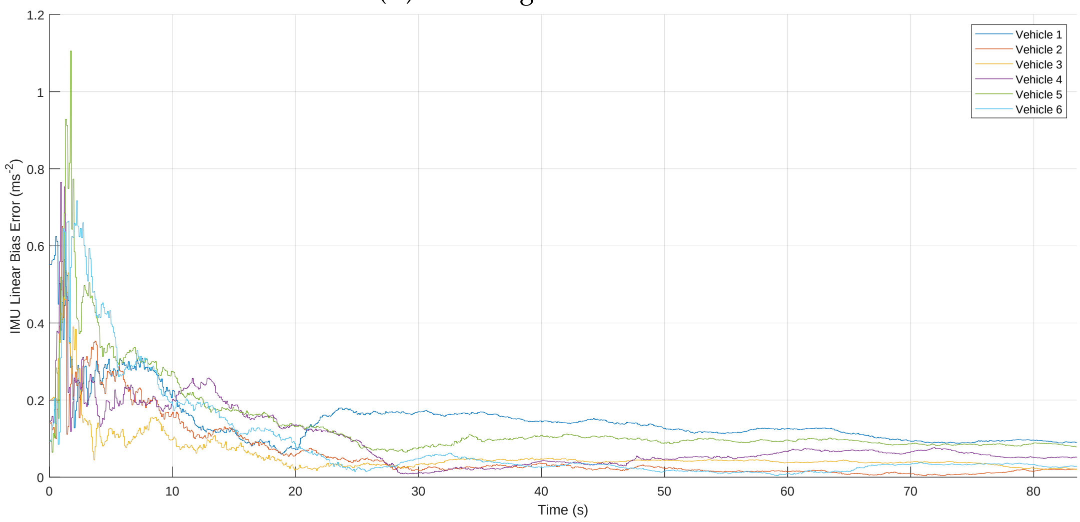


(C) Linear Velocity Error

FIGURE 5.3: Estimate errors for each of the primary state variables in the Distributed Minimum Energy collaborative localisation filter on Baseline EuRoC CL dataset. The mean error is shown in red and the range between the smallest and largest individual vehicle error is shown as the blue shaded area.



(A) IMU Angular Bias Error



(B) IMU Linear Bias Error

FIGURE 5.4: Individual bias estimate errors for each vehicle in the distributed minimum energy collaborative localisation filter on the Baseline EuRoC CL Dataset.

5.2.3 Analysis of Curvature Terms

In Section 4.4.1, we assumed that the impact of the curvature terms in the update equations of the $\check{K}$ matrix was negligible, and could thus be removed from the update equations. The update equation for the landmark measurement y_l^i, repeated here for reference, is

$$\check{K}(t^+) = \Big(I + \Delta t_l^i \check{K}(t)\check{E}_l^i + \underbrace{\Delta t_l^i\, \mathbb{P}_s\big(\check{\mathrm{ad}}_{K(t)r_l^i}\check{K}(t)\big)\check{K}(t)^{-1}}_{\text{Curvature Term}}\Big)^{-1}\check{K}(t) \tag{5.4}$$

and similarly for the inter-vehicle measurement y_j^i,

$$\check{K}(t^+) = \Big(I + \Delta t_j^i \check{K}(t)\check{E}_j^i + \underbrace{\Delta t_j^i\, \mathbb{P}_s\big(\check{\mathrm{ad}}_{K(t)r_j^i}\check{K}(t)\big)\check{K}(t)^{-1}}_{\text{Curvature Term}}\Big)^{-1}\check{K}(t) \tag{5.5}$$

In order to demonstrate the effect that these curvature terms have on filter performance, we simulated two instances of the Distributed Minimum Energy Collaborative Localisation filter (Algorithm 10), one with the curvature terms included, and one without. The mean position error over all vehicles is shown in Figure 5.5, and indicates that there is very little difference between the two versions of the filter. Figure 5.6 shows the norm of the difference between the two position estimates and indicates that the difference is on the order of cm to mm, which is approximately one to two orders of magnitude smaller than the position estimate error. The largest difference between the two filters is in the transient response in the first 10 seconds of the simulation, and the difference continues to decrease throughout the length of the simulation. These simulation results support the argument that the curvature terms have a negligible impact on filter performance, especially in the steady-state response, where the filter estimate is close to the true system state. The curvature terms do appear to have some impact on the transient behaviour of the filter, where the state estimates are further from the true state and thus update steps result in bigger jumps in the estimate. Though, overall, this does not result in significantly different performance.

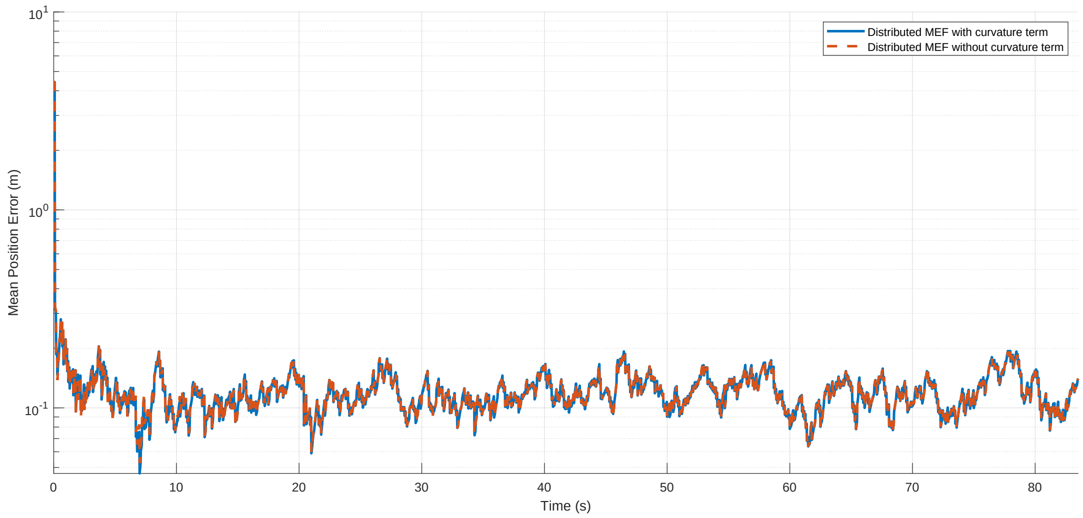


FIGURE 5.5: Mean Position Error of Distributed Minimum Energy Filter with and without the curvature terms in the update equations (5.4) and (5.5)

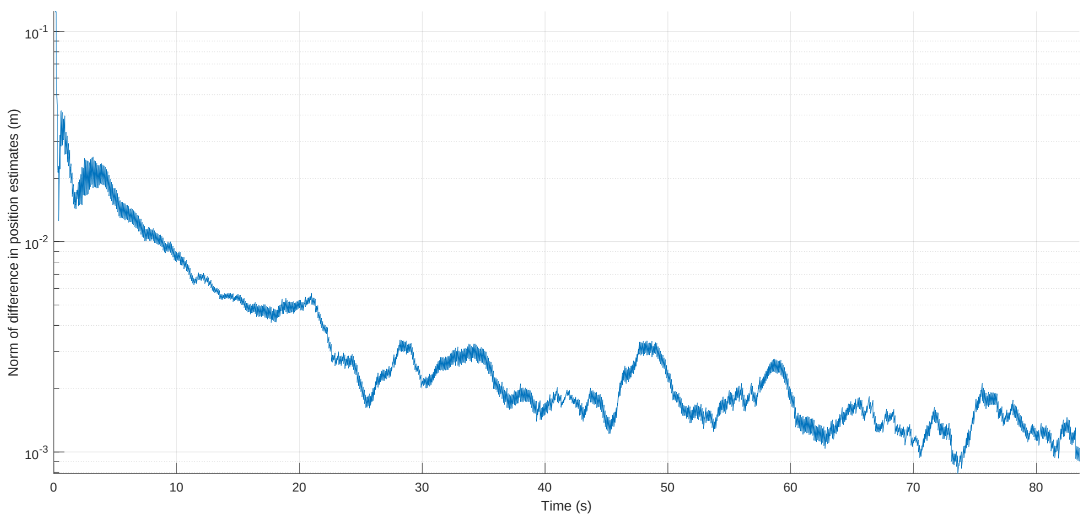


FIGURE 5.6: Difference in position estimates of Distributed Minimum Energy Filter with and without the curvature terms in the update equations (5.4) and (5.5)

5.2.4 Comparison of Distributed Filter Algorithms

Having established the baseline performance of the Distributed Minimum Energy filter, we now consider the different variations of the distributed filter proposed in Section 4.4.1. Using the same EuRoC baseline CL dataset, we present simulation results for the following five minimum energy filters

1. **Distributed Minimum Energy Filter (D-MEF)**
   Implementation of Algorithm 10
2. **Distributed Minimum Energy Filter without Curvature (D-MEF*)**
   Implementation of Algorithm 10 with the curvature term discarded, as in Section 5.2.3
3. **Non-Collaborative Minimum Energy Filter (NC-MEF)**
   Implementation of Algorithm 1. Each vehicle operates independently using only IMU and landmark sensors.
4. **Distributed Schmidt Minimum Energy Filter (DS-MEF)**
   Implementation of Algorithm 11
5. **Distributed Approximate Schmidt Minimum Energy Filter (DAS-MEF)**
   Implementation of Algorithm 12

The mean position and orientation errors for each filter are shown in Figures 5.7a and 5.7b respectively. However, due to the number of filters being examined and the high variation in errors between the filters, these figures can make it difficult to draw definitive conclusions about the relative performance of each filter. To aid in the analysis, we also present the cumulative position and orientation error for each filter in Figures 5.8a and 5.8b respectively. These figures show a clearer indication of the relative performance of each filter.

The simulation results show that there is a clear difference in the estimation error of the different filters. The results also match the expectations and intuition of the author, based on the methodology in which each filter was constructed. Unsurprisingly, the worst-performing filter is the Non-Collaborative Minimum Energy Filter (NC-MEF). Figure 5.7 shows the NC-MEF consistently, but not always, has the highest estimation error of all the filters. This is supported by Figure 5.8, which shows that the NC-MEF accumulates the largest

TABLE 5.3: Mean estimation error of baseline simulation

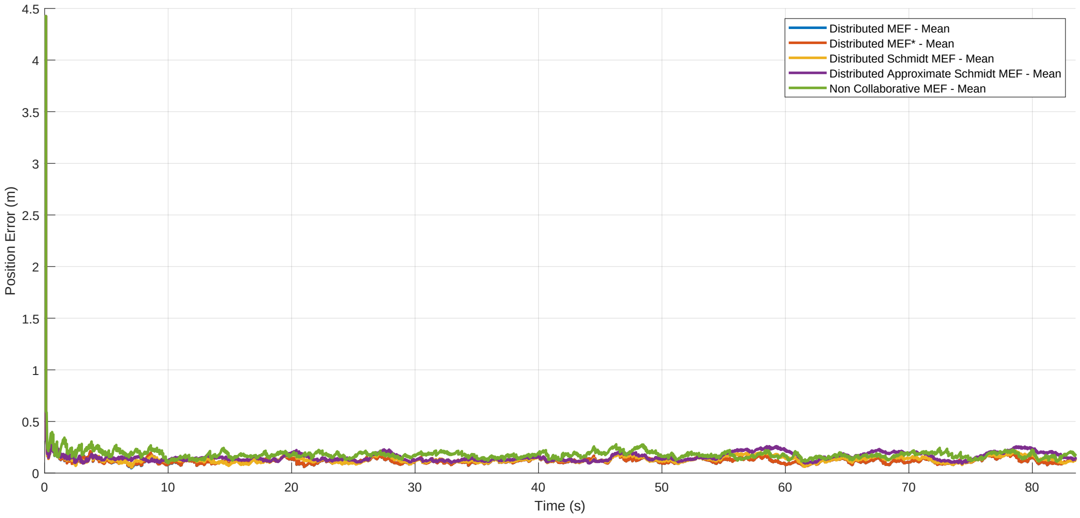
	Position m	Orientation rad	Velocity $\mathrm{m\,s^{-1}}$	Gyro Bias $\mathrm{rad\,s^{-1}}$	Accel. Bias $\mathrm{m\,s^{-2}}$
D-MEF	**0.126**	**0.018**	**0.143**	**0.014**	**0.095**
D-MEF*	**0.126**	0.019	**0.143**	**0.014**	0.096
DS-MEF	0.138	0.019	0.146	0.015	0.098
DAS-MEF	0.162	0.019	0.156	0.015	0.100
NC-MEF	0.184	0.022	0.181	0.016	0.112

errors. This gives support to the argument that collaborative localisation improves localisation performance, even though there are sufficient landmark measurements to localise each vehicle individually.

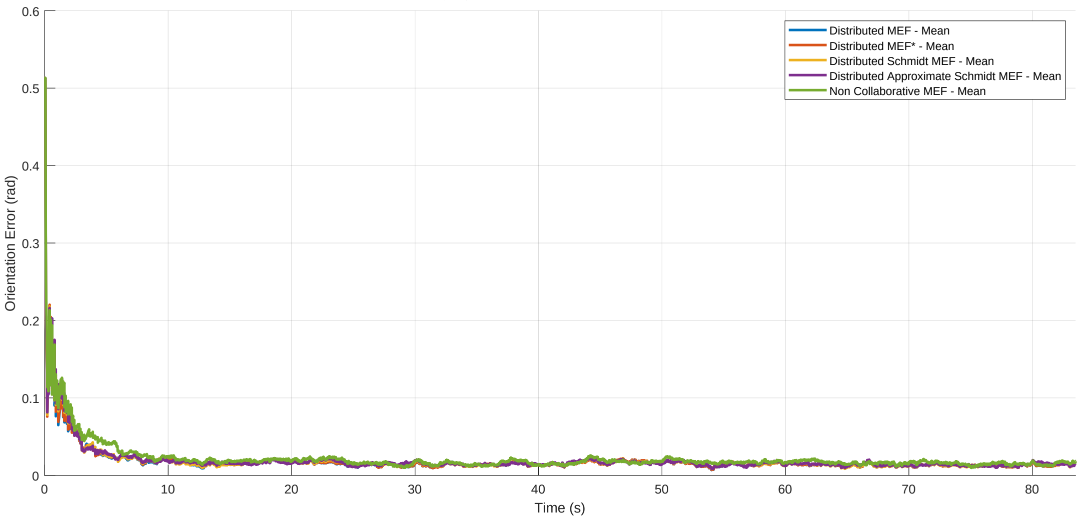
The minor difference between the Distributed Minimum Energy Filter (D-MEF) with and without the curvature term is also shown in these results, most clearly in Figure 5.8. Compared with the other filters, the difference between these two filters is small and concentrated in the transient region.

Comparing the different collaborative localisation methods, we can observe a clear relationship between the level of communication overhead and filter performance. As a refresher of the discussions in Chapter 4, the Distributed Minimum Energy Filter (D-MEF) has the highest communication overhead, requiring information to be sent and received between every agent every time an exteroceptive measurement (i.e. landmark or inter-vehicle measurement) is made. The Distributed Schmidt Minimum Energy Filter (DS-MEF) only requires all-to-all communication every time inter-vehicle measurements are made, but vehicles can process landmark measurements independently without communication. The Distributed Approximate Schmidt Minimum Energy Filter (DAS-MEF) further reduces communication so that only the vehicles involved in the inter-vehicle measurement (i.e. the one making the measurement and the one being measured) need to communicate. These reductions in communication requirements are possible due to approximations made in the filter update equations, and it is expected that this will impact localisation performance.

Referring back to Figures 5.7 and 5.8, the simulation results show that the performance of the DS-MEF, as expected, is worse than the D-MEF. By discarding updates to other vehicles during the landmark measurement step, the DS-MEF significantly reduces communication overhead at the expense of moderately worse estimation errors. The DAS-MEF follows this trend, further reducing communication requirements and resulting in diminished filter performance in the form of increased estimation errors. As shown in Figure 5.8, the DAS-MEF gives an overall smaller estimation error than the NC-MEF. However, one can observe in Figure 5.7a that there are several instances where the position estimation error of the DAS-MEF is significantly higher than the NC-MEF, for example during the time interval $[55, 60]$s. This suggests that the approximation errors introduced into the filter may have detrimental effects in some cases. We will show a more significant example of this in Section 5.4.

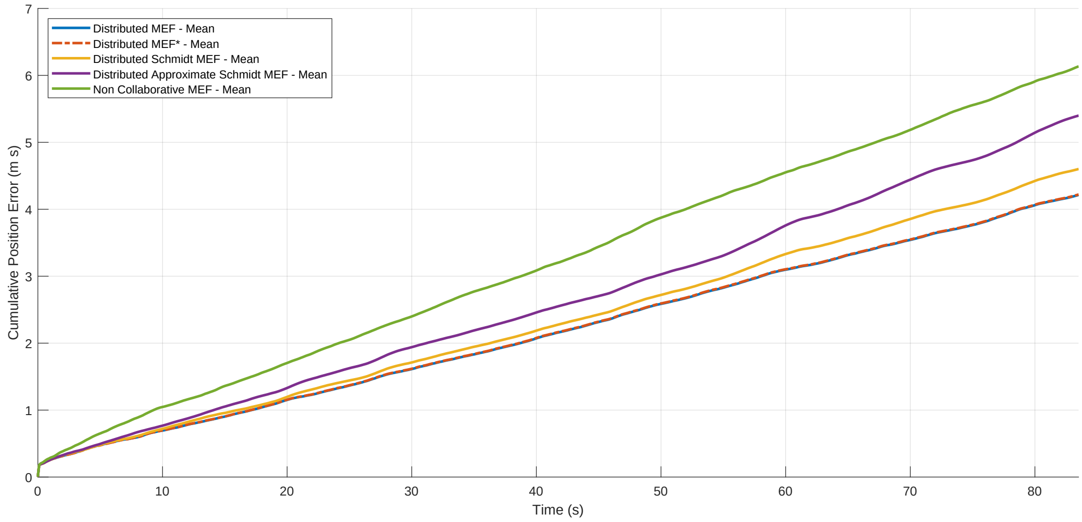


(A) Position Error

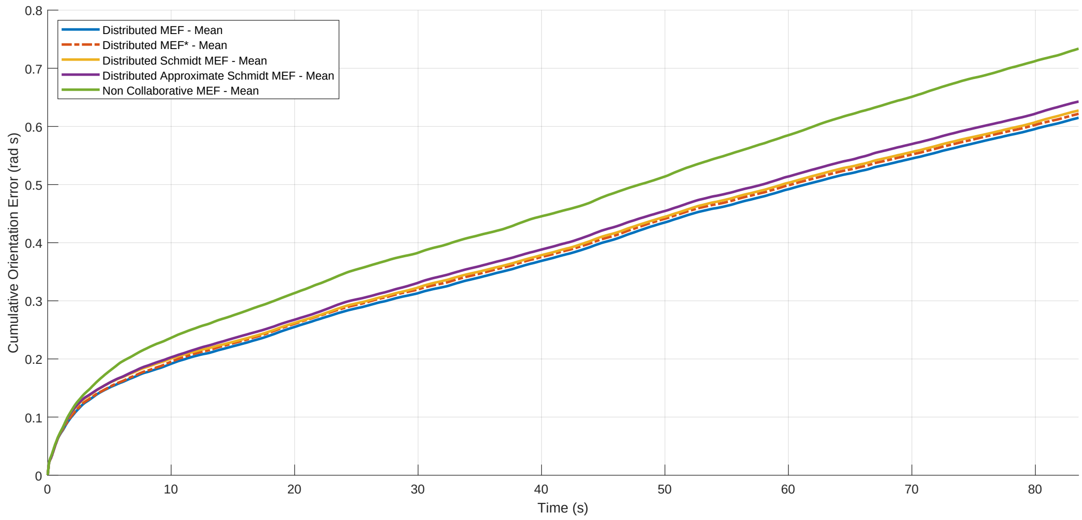


(B) Orientation Error

FIGURE 5.7: Estimate errors for different filters on the Baseline EuRoC CL Dataset.



(A) Cumulative Position Error



(B) Cumulative Orientation Error

FIGURE 5.8: Cumulative estimate errors for different filters on the Baseline EuRoC CL Dataset.

5.3 Simulation 2: Limited Sensor Data

In this section, we present a modified version of the baseline simulation to examine the performance of the different localisation algorithms under conditions where each vehicle is unable to individually localise itself using only landmark measurements. The dataset for Simulation 2 is constructed in the same way as the baseline EuRoC CL dataset, however, we use the following set of visibility matrices

$$\Theta_V(t)\big|_{t\in[0,2)} = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 \end{bmatrix}, \quad \Theta_L(t)\big|_{t\in[0,2)} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix},$$

$$\Theta_V(t)\big|_{t\in[2,\infty)} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}, \quad \Theta_L(t)\big|_{t\in[2,\infty)} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

This represents a scenario where all vehicles initially have full access to all measurements for the first 2 seconds of the simulation. Then, for the remainder of the simulation, each vehicle can only make measurements of one of the landmarks and two other vehicles. This scenario aims to highlight an area where collaborative localisation can be highly beneficial. With visibility of only one landmark, each vehicle will not be able to individually localise as there is not a sufficient amount of sensor information to make the vehicle's pose observable. However, with the addition of inter-vehicle measurements, vehicles may be able to utilise the information obtained by other vehicles in the network to localise.

We run the same five minimum energy filtering algorithms as in the previous simulation and present the mean estimation errors for position and orientation for each filter in Figure 5.9. To aid in the analysis, we also present the cumulative estimation errors for position and orientation in Figure 5.10.

5.3.1 Analysis

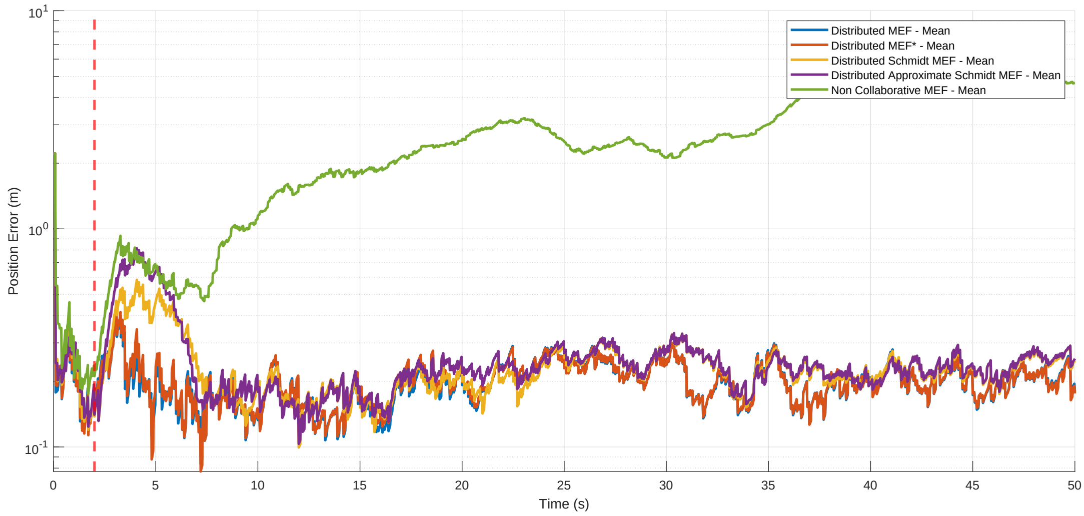
These results clearly highlight the effectiveness of the collaborative localisation methods in this limited-information scenario. The initial two seconds of the simulation, when all vehicles have visibility to all landmarks, provide the filters time to converge towards a steady state estimate. Then, at $t = 2$, when the visibility matrix changes and sensor information becomes limited, the estimates of all filters become significantly worse. However, after only a few seconds, the collaborative filters re-localise whereas the non-collaborative filter continues to diverge. The long-term behaviour shows that

TABLE 5.4: Mean estimation error for Simulation 2

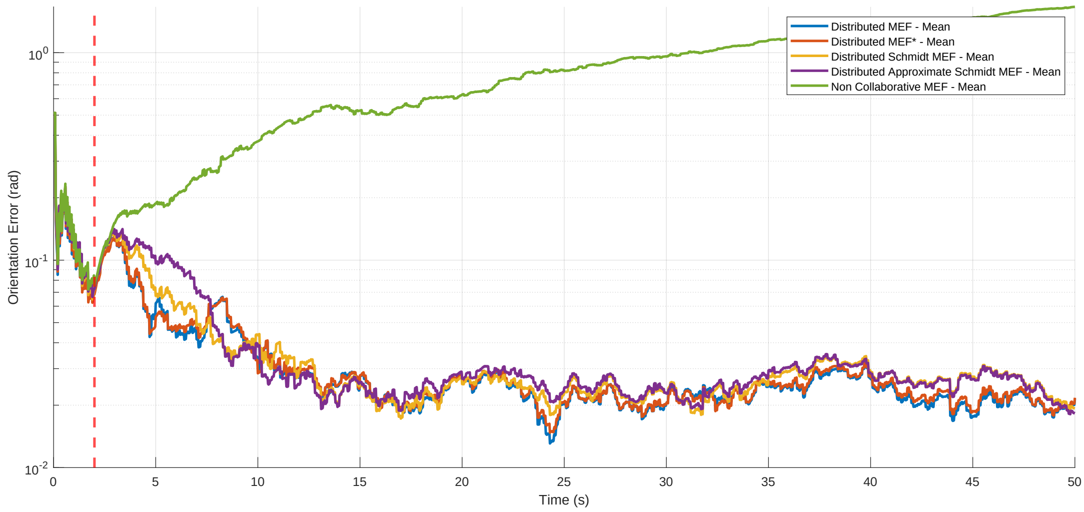
	Position m	Orientation rad	Velocity $\mathrm{m\,s^{-1}}$	Gyro Bias $\mathrm{rad\,s^{-1}}$	Accel. Bias $\mathrm{m\,s^{-2}}$
D-MEF	**0.204**	**0.034**	0.212	0.026	0.150
D-MEF*	0.206	0.035	**0.211**	**0.025**	**0.148**
DS-MEF	0.239	0.038	0.232	0.026	0.154
DAS-MEF	0.255	0.040	0.240	0.027	0.179
NC-MEF	2.748	0.831	0.675	0.058	0.236

the non-collaborative filter never regains a good position estimate while the collaborative filters maintain a position error of under 35 cm.

Amongst the different collaborative localisation algorithms, the results are broadly in line with the baseline simulation. Looking at Figure 5.10, we again see that the D-MEF has the lowest cumulative error. The DS-MEF and DAS-MEF have higher errors than the D-MEF but still manage to successfully maintain good estimates despite the limited sensor information. The largest difference between these filters and the D-MEF can be seen in Figure 5.9 during the time interval $[3, 7]$ seconds. During this interval, the D-MEF has a significantly smaller estimation error for both position and orientation, and the DAS-MEF is on par with the NC-MEF. Again this highlights the trade-off that is made between localisation accuracy and communication requirements.

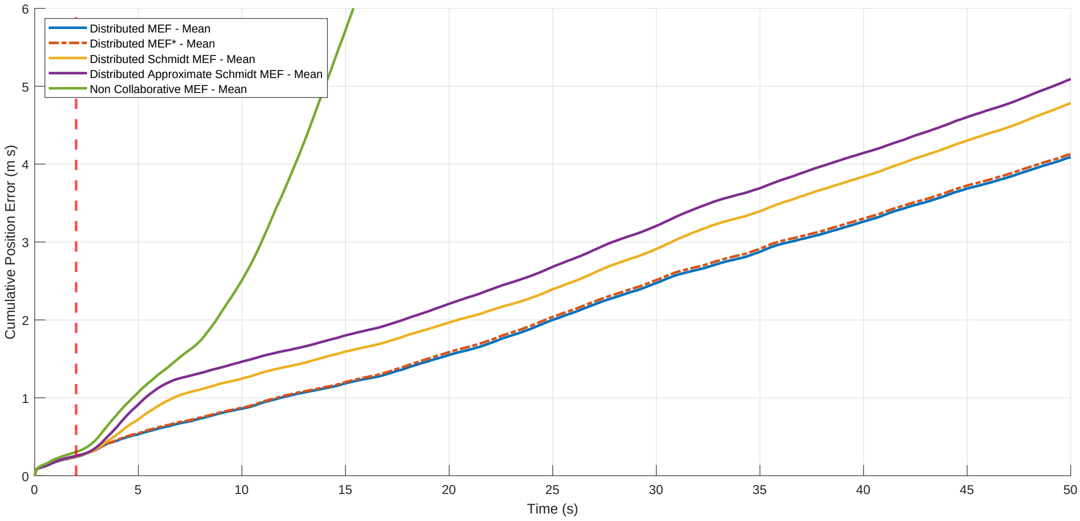


(A) Position Error

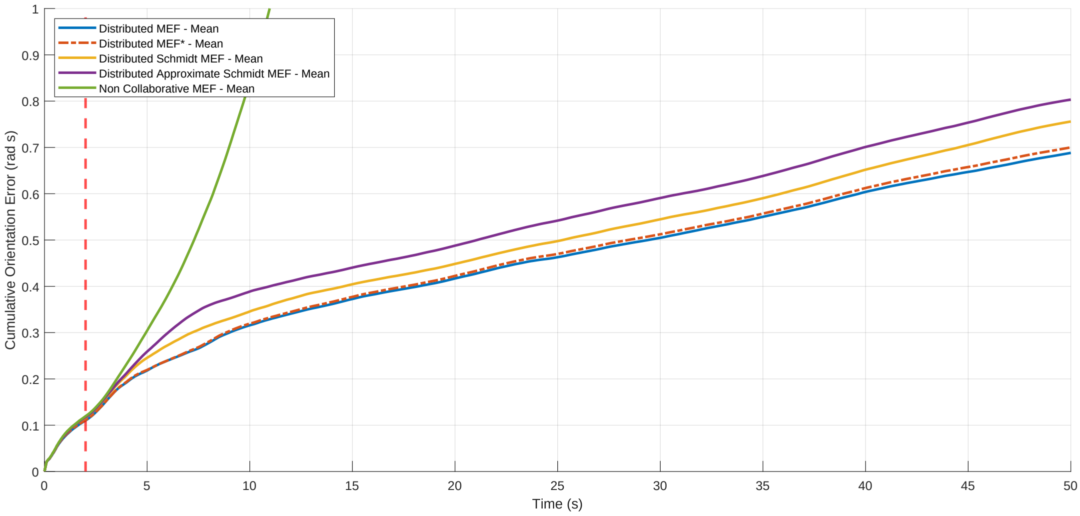


(B) Orientation Error

FIGURE 5.9: Mean estimation errors for each filter for Simulation 2. Red dashed vertical lines indicate changes in the visibility matrices.



(A) Cumulative Position Error



(B) Cumulative Orientation Error

FIGURE 5.10: Cumulative estimation errors for each filter for Simulation 2. Red dashed vertical lines indicate changes in the visibility matrices.

5.4 Simulation 3: Landmark Dropouts

The final simulation we present emulates a scenario where multiple vehicles in the network encounter a loss of absolute positioning information. This might occur, for example, if vehicles are operating in a contested environment where there is GNSS interference or when vehicles are operating beyond the line-of-sight to ground-based landmark beacons. We use the baseline EuRoC CL dataset to construct a dataset where all vehicles initially start with full access to all landmark and inter-vehicle measurements. The visibility matrices start at $t = 0$ as

$$\Theta_{\boldsymbol{V}}(t)\big|_{t\in[0,5)} = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 \end{bmatrix}, \quad \Theta_{\boldsymbol{L}}(t)\big|_{t\in[0,5)} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}. \tag{5.6}$$

Then, at 5 seconds into the simulation, the landmark visibility matrix is changed such that only one vehicle in the network has access to landmark measurements, simulating that the remaining vehicles' sensors are disrupted or out of range. The inter-vehicle visibility matrix remains unchanged, and so the visibility matrices at $t = 5$ are given by

$$\Theta_{\boldsymbol{V}}(t)\big|_{t\in[5,20)} = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 \end{bmatrix}, \quad \Theta_{\boldsymbol{L}}(t)\big|_{t\in[5,20)} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}. \tag{5.7}$$

Finally, at 20 seconds into the simulation, we restore the landmark measurements to all vehicles, simulating that the disruption has ceased, and normal operations can resume. The visibility matrices then return to

$$\Theta_{\boldsymbol{V}}(t)\big|_{t\in[20,\infty)} = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 \end{bmatrix}, \quad \Theta_{\boldsymbol{L}}(t)\big|_{t\in[20,\infty)} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}. \tag{5.8}$$

Similarly to previous simulations, we test five different minimum energy localisation algorithms, and the results can be seen in Figures 5.11 and 5.12.

TABLE 5.5: Mean estimation error for Simulation 3

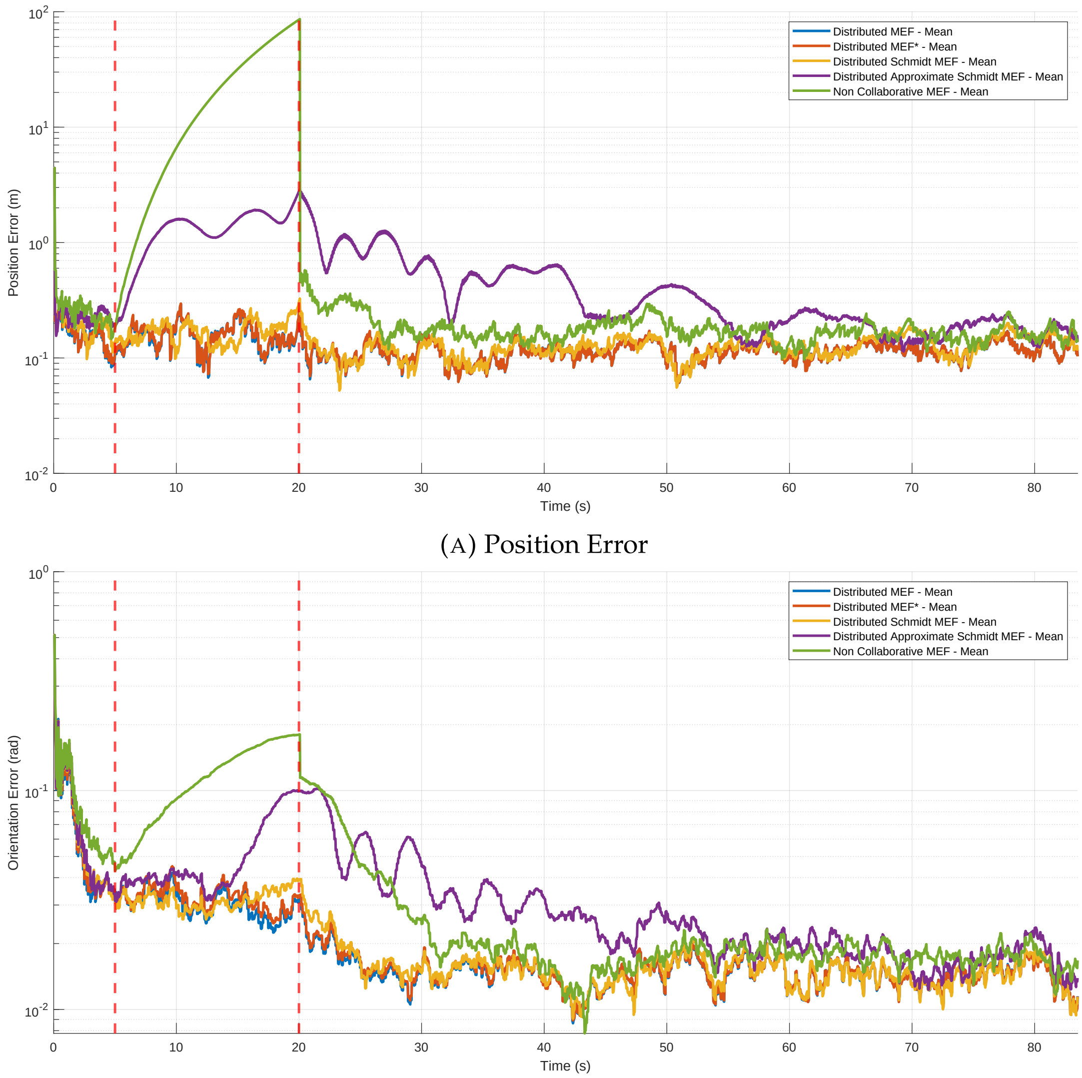
	Position m	Orientation rad	Velocity $\mathrm{m\,s^{-1}}$	Gyro Bias $\mathrm{rad\,s^{-1}}$	Accel. Bias $\mathrm{m\,s^{-2}}$
D-MEF	**0.132**	**0.022**	**0.146**	**0.015**	**0.101**
D-MEF*	0.133	**0.022**	**0.146**	**0.015**	0.104
DS-MEF	0.144	0.023	0.151	**0.015**	0.114
DAS-MEF	0.575	0.036	0.417	0.017	0.202
NC-MEF	4.780	0.045	1.230	0.019	0.183

5.4.1 Analysis

For the first 5 seconds of the simulation, the filters perform as expected, as each vehicle has full visibility of all the landmarks. Then as disruption occurs at $t = 5$, we can observe that the NC-MEF begins to diverge. With 5 of 6 vehicles having no landmark measurements, the non-collaborative filter for these vehicles can only use the IMU to perform dead-reckoning. Inevitably, due to the high noise levels in the IMU, this results in a rapid increase in estimation errors for the NC-MEF, reaching a mean error of 86 m after only 15 seconds of disrupted measurements. However, at $t = 20$, as measurements resume, the NC-MEF rapidly re-localises and returns to its expected level of performance that was previously observed in the baseline simulation.

On the other hand, the D-MEF demonstrates that accurate localisation of all vehicles can still be achieved despite the limited amount of absolute position information. In terms of position error, shown in Figure 5.11a, the difference in the D-MEF's performance in the intervals where landmark measurements are disrupted and when they are restored is not significantly different. The orientation error, shown in Figure 5.11b, does show some difference, with the mean estimation error in the range of 20 mrad to 30 mrad (1.1° to 1.7°) during the landmark sensor disruption and 10 mrad to 20 mrad (0.57° to 1.1°) after measurements are restored. The fact that the D-MEF maintains accurate pose estimates for all 6 vehicles in the network despite only one of the vehicles having access to landmark measurement information again highlights the effectiveness of collaborative localisation techniques in challenging environments. Similar observations can be made for both the D-MEF without curvature and the DS-MEF.

However, the performance of the DAS-MEF suffers significantly, during both the period of disrupted measurements and well after measurements are restored. Almost immediately from $t = 5$ when landmark sensor information is disrupted, we can observe the position error of the DAS-MEF begin to increase. While the rate of increase is not as high as the NC-MEF, it does not appear to show any signs of plateauing and reaches a maximum mean error of over 20 m within 15 seconds. At $t = 20$, as landmark measurements are restored, the DAS-MEF slowly begins to re-localise, but there is a significant delay before performance returns to the baseline level. During the period between $t = 20$ and $t = 50$, even the NC-MEF has both lower position and

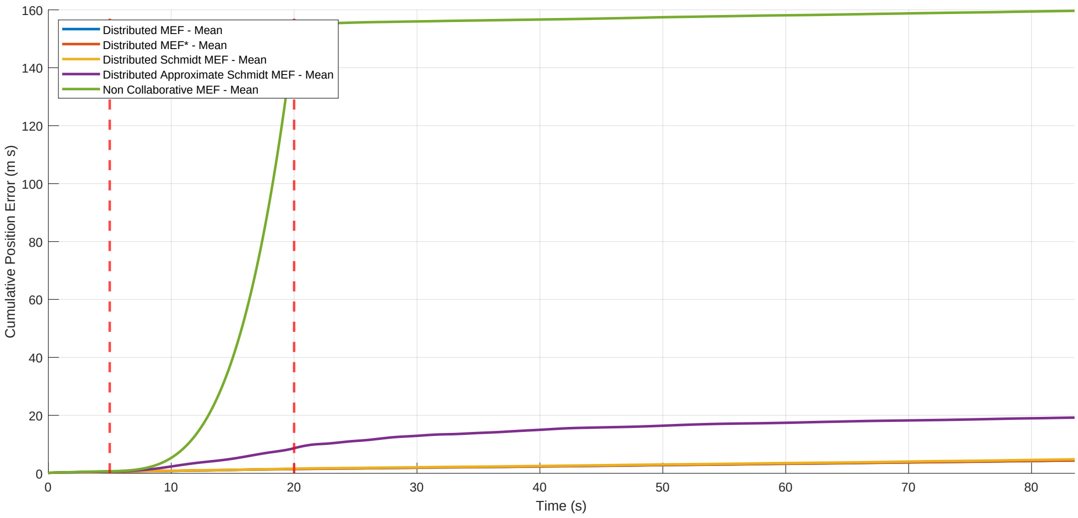


(A) Position Error

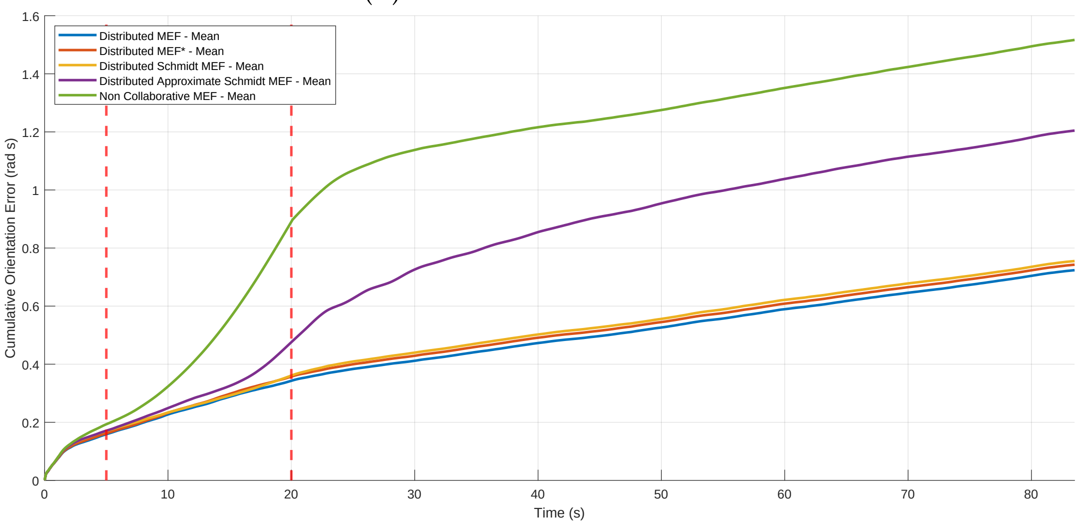
(B) Orientation Error

FIGURE 5.11: Mean estimation errors for each filter for Simulation 3. Red dashed vertical lines indicate changes in the visibility matrices.

orientation estimation errors, demonstrating that there are some instances where collaborative localisation methods, the DAS-MEF in particular, can produce worse estimates than having no collaboration at all. The long delay in returning to the baseline performance level, especially when comparing the DAS-MEF with the DS-MEF, suggests that the approximations made in the DAS-MEF update steps during the period of disrupted landmark measurements result in significant errors in the $\check{K}$ matrix. The comparatively better performance of the NC-MEF also suggests that a better strategy may be to completely re-initialise the $\check{K}$ matrix of the DAS-MEF at points where the visibility matrix changes.



(A) Cumulative Position Error



(B) Cumulative Orientation Error

FIGURE 5.12: Cumulative estimation errors for each filter for Simulation 3. Red dashed vertical lines indicate changes in the visibility matrices.

5.5 Summary

Through the use of simulation using a hybrid dataset containing real-world trajectories and IMU sensor data, and synthetically generated relative position measurements, we have examined the performance of a collection of minimum energy localisation algorithms. The simulation results provide strong evidence supporting the concept of minimum energy filtering as a viable option for estimation systems with complex nonlinear dynamics, although further tuning is required to maximise performance.

The three simulations presented provide a cross-section of the typical performance under different conditions and reveal several interesting behaviours. The relationship between localisation performance and communication overhead is clear in all three simulations, showing that algorithms that reduce communication overhead suffer a performance penalty as a result. Also, the benefits that collaborative localisation provides over non-collaborative algorithms are clearly evident. In all simulations, the Distributed Minimum Energy Filter provides lower estimate errors across all state variables. It remains stable during periods where sensor information is limited while the non-collaborative filter estimates diverge.

The simulations also highlight some examples of the Distributed Approximate Schmidt Minimum Energy Filter performing measurably worse than all other filters, including the non-collaborative MEF. This shows that, in some cases, the approximations made in the DAS-MEF algorithm can be detrimental to performance compared with not incorporating the measurements at all. This requires some further analysis to explore what conditions invoke this behaviour and whether there are any ways to avoid it.

The results presented in this chapter provide a solid foundation to evaluate the performance of minimum energy collaborative localisation algorithms using real-world data. However, as discussed at the beginning of this chapter, the semi-synthetic construction of the dataset limits the extent of the conclusions that can be drawn. In order to truly evaluate real-world performance, a proper collaborative localisation dataset is needed — one with real inter-vehicle measurements as well as landmark measurements rather than the synthetically generated measurements used in this chapter. Creating such a dataset and using it to evaluate filter performance will be the focus of the remainder of this thesis.

Chapter 6

Building a Heterogeneous Fleet of Collaborative Robots

FIGURE 6.1: The fleet of three Uncrewed Aerial Vehicles (UAV) and three Uncrewed Ground Vehicles (UGV) designed and built to conduct collaborative localisation experiments.

As we have discussed many times in the previous chapters of this thesis, the lack of a high-quality multi-vehicle inertial collaborative localisation dataset presents a significant limitation not only in the evaluation and verification of this work but also for many others. Such a dataset would allow for the testing of collaborative localisation algorithms under real-world circumstances and also enable meaningful evaluation of the performance of different algorithms using a common benchmark dataset. However, with an ever-growing interest in collaborative robotics, and in collaborative localisation specifically, the fact that such a dataset does not already exist gives some hints as to the level of cost, complexity and time required to create one.

In this chapter, we present our work on constructing a fleet of ground-based and aerial robots, pictured in Figure 6.1, to support the creation of collaborative localisation datasets. We start with an exploration of existing datasets published in the literature to identify the features that distinguish high-quality datasets and make them useful to others. From there, we develop a set of high-level requirements to specify the properties of an ideal collaborative localisation dataset and then use these requirements to inform the design, build and evaluation process of the physical robots. We will detail the many technical decisions, component choices, and subsystem designs throughout the process and how they relate to achieving the high-level system specification.

6.1 Analysis of Existing Public Datasets

Before creating a new dataset, it can be helpful to examine existing datasets in the literature in order to gain some understanding of the different elements of a good localisation dataset and also identify any deficiencies that can be addressed. In Chapter 2, we identified the MRCLAM [101] dataset as the most popular multi-robot collaborative localisation dataset in use. We also identified EuRoC [105] and TUM VI [106] as more general robotics datasets that are not strictly used for collaborative localisation. All three datasets will serve as examples of popular high-quality datasets currently in use in the literature.

The structure of each of these datasets is broadly similar. Each dataset contains a collection of data recorded from different experiments, which are often individually referred to as 'datasets'. To avoid ambiguity, we will refer to each of these sequences as 'missions'. There are 9 missions In the MRCLAM dataset, 11 missions in the EuRoC dataset, and 28 in the TUM VI dataset. Each mission has a typical duration of several minutes during which the robots move through the environment and collect sensor data. The data available for each mission has three components, the sensor data, the ground-truth data and the metadata.

6.1.1 Ground-Truth Data

The ground-truth information is perhaps the most important element of a dataset, as it provides the pose of each robot which is used to evaluate filter estimates. It must be noted that it is not possible to exactly determine the 'true' pose of the robot, as it is typically measured with a motion-capture system such as VICON [25] or OptiTrack [26], which itself has some level of uncertainty. However, these systems are typically accurate to the order of 1 to 10 mm, which is often sufficient when evaluating estimation errors that are at least an order of magnitude larger. In the MRCLAM dataset and some EuRoC missions, a VICON system is used to provide the ground-truth pose, while the TUM VI dataset uses OptiTrack. However, in the Machine Hall EuRoC missions and some TUM VI missions, the experimental area is too large for the motion capture system to cover, and thus full pose ground truth cannot be provided.

The limited coverage of motion capture systems is a major issue for large-scale missions, especially those outdoors. Additionally, motion capture systems cannot directly measure other state variables, including vehicle velocity and IMU biases.

The EuRoC dataset attempts to provide full ground-truth data for all state variables, including velocity and biases, by performing a maximum likelihood estimation over the interpolated motion capture data and the IMU sensor data. This provides a valuable reference for evaluation, but one must be careful when labelling this data as 'ground truth', as it is not independent of the IMU sensor data.

6.1.2 Sensor Data

In order to support localisation, both proprioceptive and exteroceptive sensor data are provided in each dataset. In the MRCLAM dataset, the motor velocity commands are recorded to provide velocity information, while in the TUM VI and EuRoC data is recorded from the IMU to give inertial measurements. Both EuRoC and TUM VI record IMU data at 200 Hz, while the MRCLAM velocity commands are recorded at 67 Hz which is appropriate given the lower velocities of the ground-based robots.

As the MRCLAM dataset is the only multi-robot dataset of the three, it is the only one to provide relative position measurements between robots, and because it is the only localisation dataset (as opposed to a visual odometry dataset), it is the only one to provide relative position measurements to known landmarks in the environment. Both of these measurements are made by the same sensor, which is a forward-looking camera mounted to each robot, which detects barcode-like markers on other robots and the landmarks, shown in Figure 6.2. Computer vision is used to detect the barcode markers in the image and identify the unique ID for each robot or landmark. The relative size and position of the barcode in the image determine the range and bearing measurement which is then converted into a relative position measurement.

6.1.3 Metadata

Each dataset includes metadata that is necessary to run relevant algorithms and evaluate the performance. This can include such information as landmark positions, sensor offsets, calibration parameters, and mission descriptions. Good quality metadata makes implementing algorithms using the dataset significantly easier. For example, both EuRoC and TUM VI provide pre-calibrated camera intrinsic and extrinsic parameters, but also provide additional raw data in case the user wishes to perform their own calibration. The IMU noise

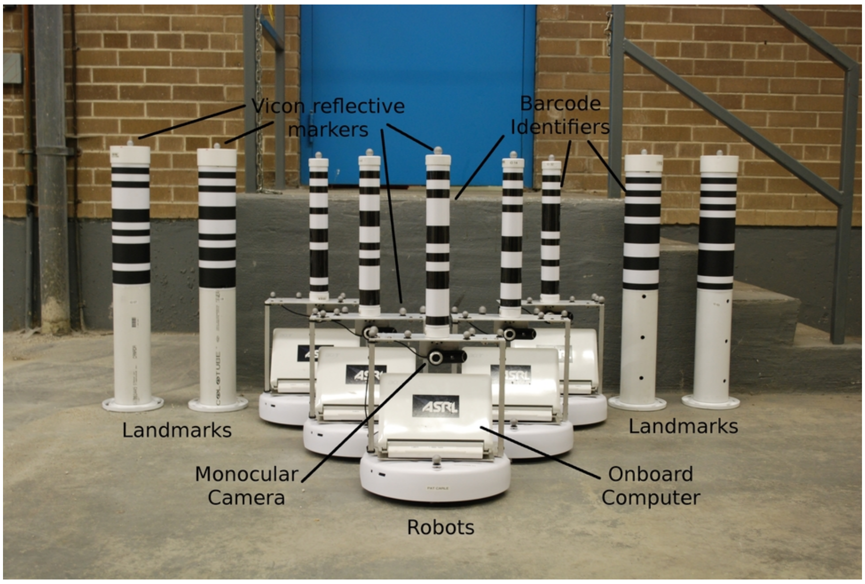


FIGURE 6.2: Image of robots and landmarks used for MRCLAM dataset [101].

parameters are also measured and reported in both datasets, providing a principled starting point for filter tuning.

Another factor that impacts the usability of the dataset is the presentation and format of the data. All three datasets have an accompanying website that describes the dataset, provides links to download the data, shows images of the experimental setup, describes the data format, and references the corresponding paper which provides more technical details. The MRCLAM website also provides some example scripts to load the data and visualise the missions, making it straightforward for a new user to get started.

6.2 Dataset Requirements

Based on the above analysis of popular existing datasets, as well as an understanding of what elements are missing from these datasets, we developed the following set of goals for the dataset that will eventually be created;

1. The overall aim is to produce a high-quality dataset for benchmarking collaborative localisation algorithms.
2. In order to demonstrate complex inter-vehicle interactions, the dataset should contain at least 4 robots.
3. To demonstrate interactions between heterogeneous robots, the dataset should use both ground-based and aerial robots.
4. Inertial measurement data should be recorded by each robot at a frequency greater than 200 Hz.
5. Exteroceptive measurement data should be recorded by each vehicle with sufficient frequency and accuracy to enable localisation.

(a) In order to test the current algorithms we have designed, we require relative position measurements.

(b) Robots should be able to measure their position relative to a set of fixed landmarks within the environment.

(c) To enable collaborative localisation, robots must also be able to make measurements of other robots in the network.

6. Accurate ground-truth data should be recorded by each robot to evaluate the performance of localisation algorithms.
7. The robots should be capable of operating in a moderately large outdoor environment, at least 100×100 m in area.
8. The resulting dataset should be well documented and explained, with all relevant metadata and calibration data provided.
9. The dataset should include several missions with different trajectories that demonstrate a variety of vehicle interactions such as formations.
10. Each mission should be approximately 3 to 5 minutes in duration.

The requirements serve as a guide for the development of the robotic platforms, informing the selection of components, development of software, and planning of the missions.

6.3 Overview of Experimental Platforms

Based on the high-level requirements described above, we developed two experimental platforms, an uncrewed aerial vehicle (UAV) and an uncrewed ground vehicle (UGV). These platforms consist of a mix of off-the-shelf and custom-made components carefully selected and integrated in order to meet the mission specifications. Where possible, components and subsystems are common to both platforms, but size, weight and power (SWAP) limitations of the UAV necessitate variations. Similarly, the increased SWAP capabilities of the UGV enable additions to be made beyond the baseline level of capability. In the following sections, we give an overview of each platform and the different subsystems.

6.3.1 Uncrewed Aerial Vehicle

The UAV is a quadcopter based on a Tarot Iron Man 650 carbon fibre airframe, providing a lightweight, strong, and adaptable base to attach components and sensors. The flight controller is a Holybro Durandal running the open-source Ardupilot flight control software. In addition to this, an NVIDIA Jetson Nano is mounted above the flight controller and acts as the onboard computer running the Robot Operating System (ROS). The onboard computer is responsible for recording data from all the connected sensors and provides the telemetry link back to the ground station via Wi-Fi.

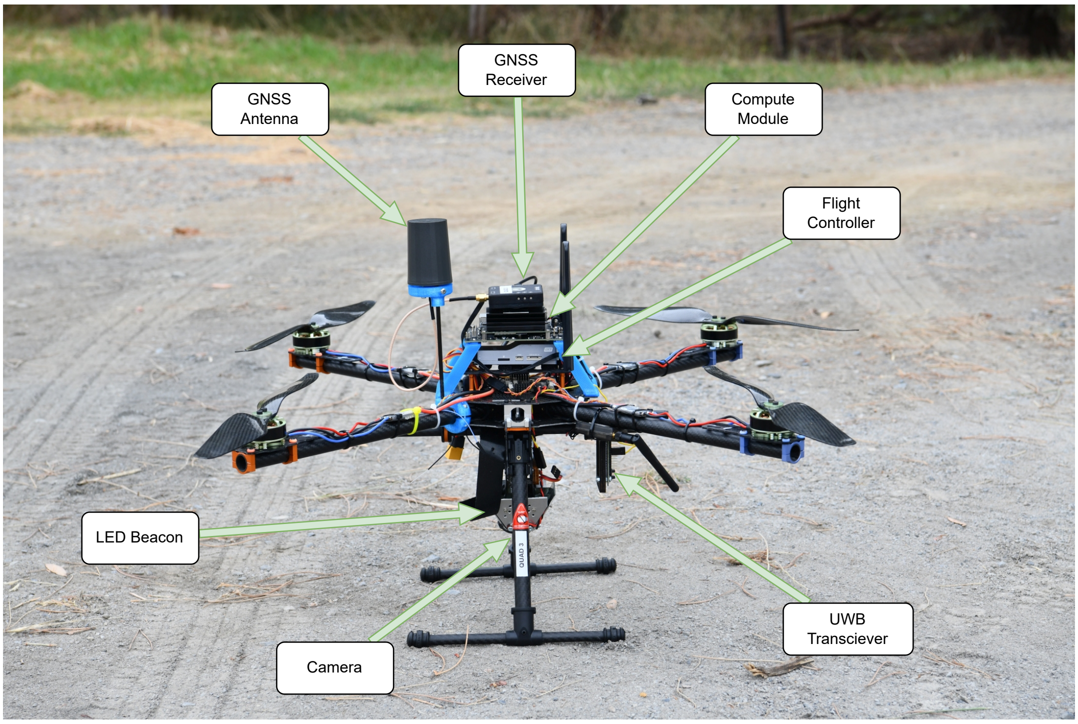


FIGURE 6.3: Annotated components of the Uncrewed Aerial Vehicle (UAV)

Other components mounted to the UAV include the RC receiver, GNSS module and antenna, UWB sensor, Camera, and LED assembly. A block diagram of how each component is connected is shown in Figure 6.4, and an image of the UAV is shown in Figure 6.3. We will describe each of the sensor subsystems in more detail in the following section.

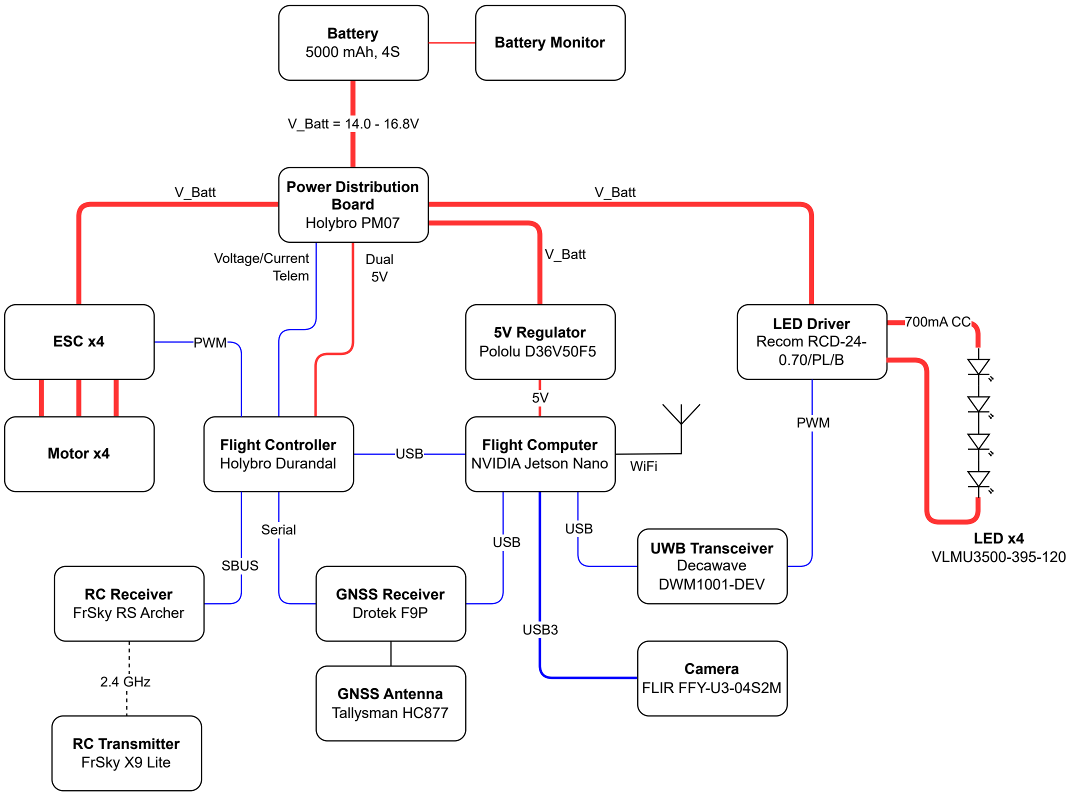


FIGURE 6.4: Hardware block diagram for the UAV showing power connections in red and data connections in blue

6.3.2 Uncrewed Ground Vehicle

The UGV is built around the Clearpath Robotics Jackal, which is a rugged, outdoor, differential-drive ground-based robot. The Jackal comes pre-equipped with an onboard computer running ROS which manages the motion control and Wi-Fi connectivity. Similarly to the UAV, we attach a Holybro Durandal flight controller running Ardupilot in order to provide a common set of sensors (specifically the IMU), although in this case the flight controller on the UGV is not performing any control functionality.

The UGV is also equipped with an RC receiver, GNSS module and antenna, UWB sensor and LED assembly. One difference to the UAV is that the UGV is equipped with two cameras, one facing forward and the other facing backward. The block diagram of the UGV, in Figure 6.6, shows how the components are interconnected, and Figure 6.5 shows the physical layout of the components.

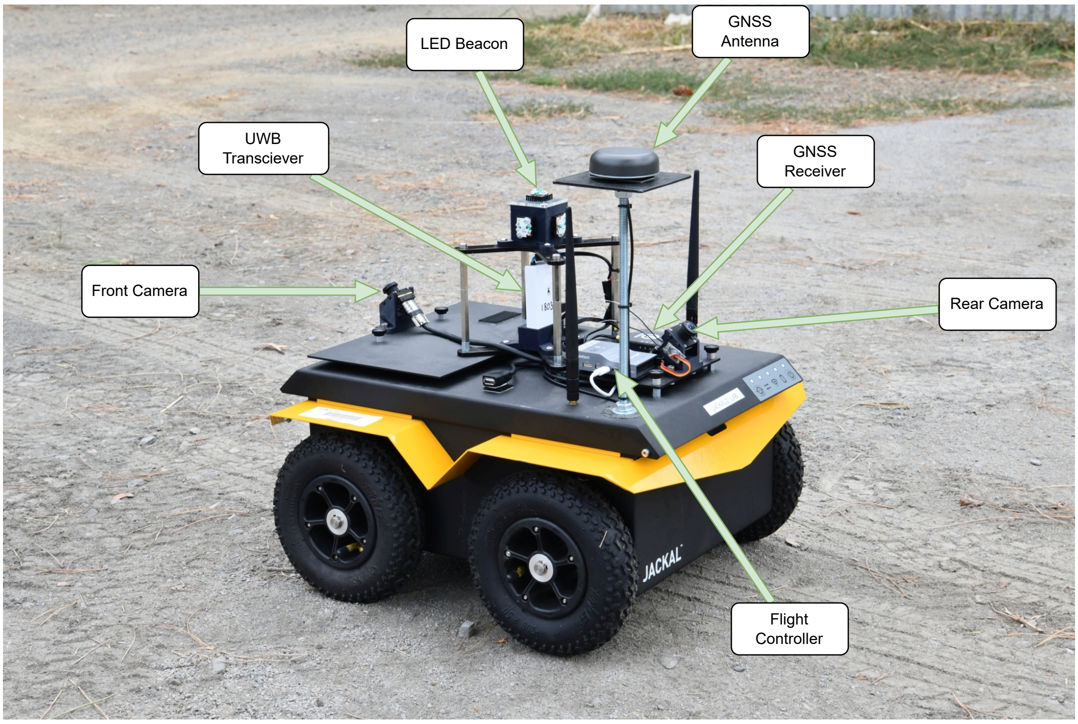


FIGURE 6.5: Annotated components of the Uncrewed Ground Vehicle (UGV)

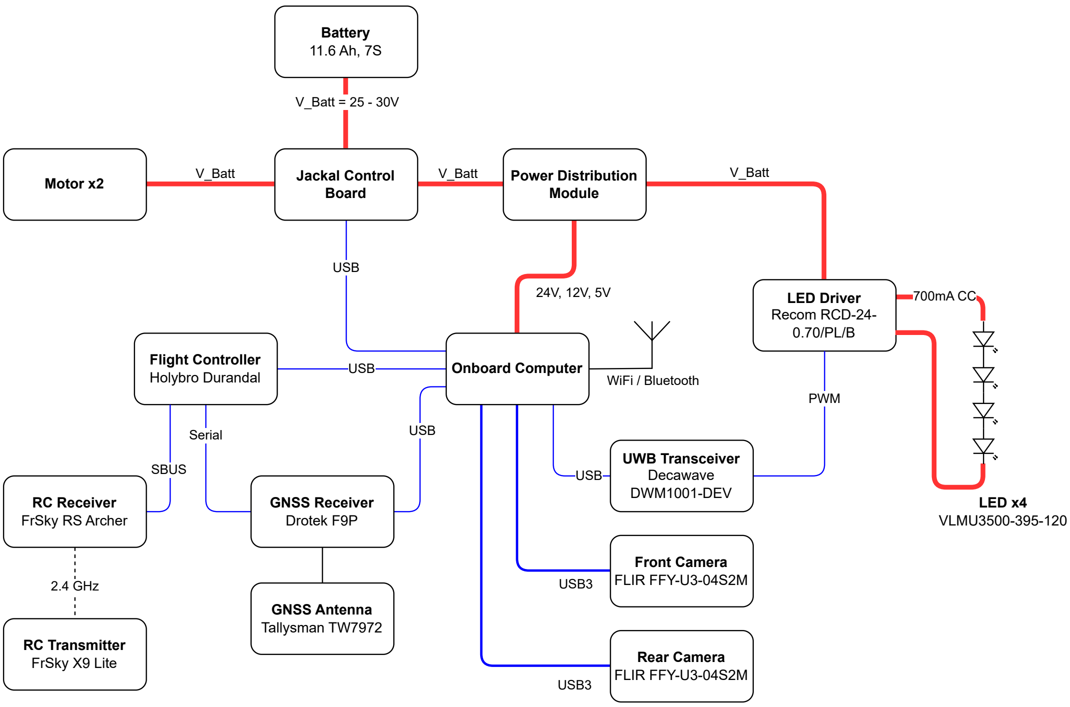


FIGURE 6.6: Hardware block diagram for the UGV showing power connections in red and data connections in blue

6.4 Sensor Systems

The three core components, mentioned above, of an inertial collaborative localisation dataset are the inertial measurements, the relative position measurements and the ground truth data. In this section, we provide details on how each of these pieces of data is measured on each platform.

6.4.1 Inertial Measurements

In addition to performing the task of guidance, navigation, and control (GNC) on the UAVs, the flight controllers conveniently come with integrated IMUs which can be recorded as part of the dataset. Each flight controller is equipped with two IMUs (ICM-20689 and BMI088), measuring linear acceleration and angular velocity, as well as a magnetometer (IST8310), measuring the local magnetic field, and a barometer (MS5611), measuring the local air pressure. While the IMU is the primary sensor of interest, we also record the magnetometer and barometer data in case this may become of use in the future. By equipping the UGVs with identical flight controllers (even though they do not perform any GNC tasks), we can provide a common, compatible set of measurements for each vehicle in the resulting dataset.

For small agile vehicles such as the UAV, a high IMU sample rate is essential to capture rapid accelerations and movements. The two onboard IMUs are capable of sampling at 8 kHz and 2 kHz respectively, however, the Ardupilot flight control software is not capable of continuously recording this high-rate data from the IMU. Instead, we rely on the built-in logging functionality within Ardupilot which can record pre-integrated IMU data at a rate of 400 Hz. This rate should be more than sufficient for localisation purposes and is double the rate of the IMU data recorded in both the EuRoC and TUM VI datasets.

6.4.2 Ground Truth Position Data

Given that we plan to conduct missions in an outdoor environment, it is not practicable to use a motion capture system to record ground-truth pose data. The laser tracking system used in some EuRoC missions is also unsuitable due to its high cost, as well as the fact that it can only track a single target. GNSS can provide a cost-effective source of highly accurate absolute positioning information and will work in a multi-vehicle environment. However, GNSS does not provide a means to determine vehicle orientation (at least with a single receiver) and also requires additional infrastructure in order to achieve high levels of accuracy.

As discussed in Section 2.1.2, standard GNSS accuracy is limited to the order of 1 metre. Accuracy can be further improved down to the centimetre level by the use of Real-Time Kinematic (RTK) correction, but this requires three components; a receiver capable of performing RTK corrections, a dual-band or tri-band GNSS antenna, and a reference ground station to transmit correction data. With this in mind, we equipped each vehicle with the uBlox F9P, a high-end consumer-grade GNSS receiver. On the UGV, we use a Tallysman TW7972 tri-band GNSS antenna, while on the UAV we use a Tallysman HC977 tri-band helical antenna due to the lower weight.

It is possible to construct a local reference station using a fixed GNSS receiver and transmit the correction data to the vehicles over the local network or radio broadcast. However, to simplify the experimental setup, we opted to use the AUSCORS NTRIP service[1], which provides the correction data over the internet from fixed continually operating reference stations (CORS).

While the reported accuracy of the F9P while using RTK corrections is 1 cm [122], several additional factors contribute to the positioning error. As the distance between the reference station and the receiver increases, the error in the RTK corrected position will increase at the rate of approximately 1 part per million (corresponding to 1 mm per km) [123]. For the location of the primary experiments in this thesis, the reference station was located approximately 30 km away which can be expected to introduce an additional 3 cm of positioning error. Additional positioning errors will be introduced by several other sources, including the phase centre variation (PCV) of the GNSS antenna, processing time delays, radio interference, and line-of-sight interruptions. It is important to consider this error as we will be using the RTK GNSS position as the 'ground-truth' data when evaluating filter performance. Thus, variations in estimation error on the order of 1 cm to 10 cm will be insignificant.

The physical mounting of the GNSS antenna requires special attention. As the signals from GNSS satellites are very weak, they are particularly sensitive to interference and obstruction. The RTK functionality requires a higher quality signal than standard positioning and thus is even more prone to interruptions. When first mounting the GNSS antennas on the UGV and UAV, we encountered significant issues with location accuracy and were unable to achieve an RTK fix. The source of interference was eventually identified as electromagnetic interference from the USB3 cable connecting the camera to the compute module, which is a common issue for some wireless devices [124]. To mitigate this issue, the USB3 cables were wrapped in conductive foil tape and the GNSS antennas were also mounted as far away from other electronic components as possible. On the UGV, pictured in Figure 6.5, the antenna is mounted to a steel plate which sits atop a vertical steel rod, while on the UAV, the antenna is attached with a carbon fibre rod and 3D-printed mounting components. The steel plate on the UGV acts as a ground plane, which helps to reduce interference, however, the helical antenna on the UAV does not benefit from a ground plane and so was not included. These modifications ensured each vehicle maintained a reliable RTK fix at all times during the experiments.

[1] https://gnss.ga.gov.au/stream

6.4.3 Relative Position Measurements

Perhaps the most challenging aspect of designing the experimental platform is the relative position measurements. Unlike the other two dataset components, there is no suitable off-the-shelf sensor that is capable of directly measuring relative positions between two vehicles. After a review of existing technologies and examples in the literature, we developed a bespoke sensor system to generate the required measurements between platforms as well as fixed ground-based landmark beacons. Similar to the MRCLAM dataset, the system measures range and bearing separately and then combines the data to produce a relative position measurement. The bearing measurements are made through a camera system inspired by Walter, Saska and Franchi [115] which tracks and identifies flashing LEDs on each target. Separately, range measurements are made using a network of Ultra-Wideband (UWB) sensors equipped on each platform and landmark beacon. Due to their complexity and bespoke nature, we will explore both of these subsystems in more detail in the proceeding sections.

6.5 Bearing Measurement Subsystem

Creating a system to measure relative bearings between two moving vehicles at distances of up to 100 m in an outdoor environment was a particularly challenging task. Many different proposals were considered, but we ultimately settled on a camera-based system inspired by the work of Walter, Saska and Franchi [115], [114]. The system consists of a camera attached to the vehicle and flashing LED markers attached to the targets to be measured (in this case, landmark beacons and other vehicles). Using computer vision, the flashing LEDs are detected in the images from the camera, and the unique flashing pattern of each target is decoded to identify the target. By calibrating the intrinsic and extrinsic parameters of the camera, the relative bearing between the vehicle and the target can be computed from the location of the LED in the image.

As is often the case in robotics, the system concept is relatively simple, however, many challenges arise in the implementation. These challenges include choosing the right set of hardware components, determining an appropriate LED flashing pattern, developing the computer vision software to detect, track and identify the LEDs, and performing the calibration of the camera parameters. On top of this, the integration with the rest of the system and mounting onto the UAV and UGV need to be considered.

6.5.1 Hardware Selection

There are four main hardware components of the system; the camera, lens, filter and LED. Each component was specially selected to maximise system performance while meeting the required specifications.

As a camera was to be mounted to each vehicle, a low SWAP camera was essential for the UAV. Also, as the identification system relies on detecting a flashing LED, a high frame-rate camera would allow for higher frequency LEDs and faster identification of targets. Based on this, we selected the FLIR Firefly FFY-U3-04S2M camera with a weight of 20 g, dimensions of 27 mm × 27 mm × 14 mm, and power consumption of 2.2 W [125]. It is also capable of capturing 120 frames per second and has a global shutter, making it ideal for the intended use case.

When selecting a camera lens, a high field of view was desirable, as bearing measurements can only be made when the target is within the field of view of the camera. Again, due to the limitations of the UAV, low size and weight were also essential. Initially, we considered the idea of using a catadioptric lens similar to [126]. Compared with a traditional lens, this would give a 360-degree cylindrical field of view with high resolution towards the horizon in all directions. However, as we were unable to source a commercial supplier for such a lens, and custom manufacturing was outside the scope of this project, we instead selected a more traditional fish-eye lens. Based on the previous success of Walter, Saska and Franchi [115], we selected the Sunex DSL215 SuperFisheye lens, which has a field of view of 185°. However, due to the high image distortion imposed by the extremely wide-angle lens, if the camera axis is vertical then the image resolution in regions near the horizon will be greatly reduced. This affects the precision of the bearing measurement when performing the inverse mapping from image coordinates.

The next component of the bearing measurement system is the lens filter. In the system designed by Walter, Saska and Franchi [115], they use a combination of Ultraviolet (UV) LEDs and an Ultraviolet band-pass filter attached to the camera. This allowed background light from the environment to be filtered out, while still making the LEDs visible in the image. The UV wavelength was selected as it is relatively uncommon in a natural environment, as most objects do not reflect UV from the sun. However, one must consider that the camera and lens combination were designed for visible wavelengths rather than UV. Thus, we must select a UV wavelength in the near-visible spectrum to retain the transmissivity properties of the lens and the sensitivity of the camera.

Figure 6.7 shows the camera sensitivity and sunlight intensity as a function of wavelength. UV is typically considered as any wavelength between 10 nm to 400 nm. We observe that the sunlight intensity decreases as the wavelength decreases in the UV spectrum, but the camera sensitivity also decreases. If the camera is not sensitive to the selected wavelength, the range of the system will be limited and exposure times may need to be increased, which could result in motion blur and reduced measurement accuracy. We are also limited by what filters and LEDs are commercially available, as well as the limits for human-eye safety. As a compromise, we selected LED beacons with a peak emission wavelength of 395 nm coupled with a UV band-pass filter centred at 395 nm. The emission spectrum of the LED and the transmissivity of the filter are also shown in Figure 6.7.

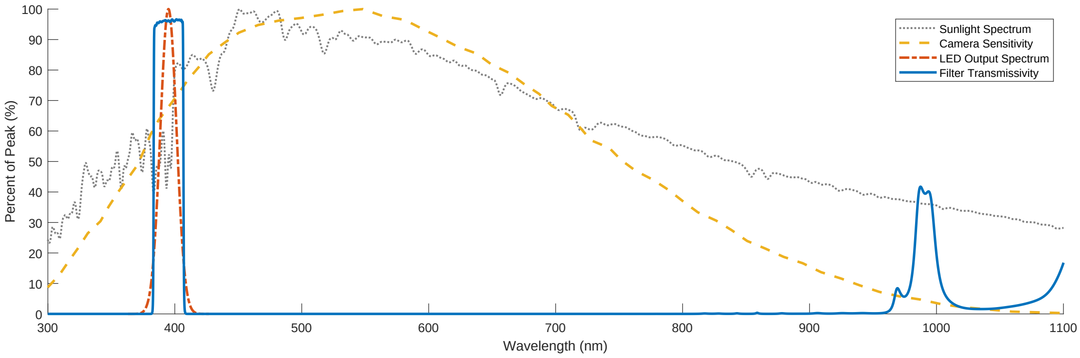


FIGURE 6.7: Relationship between LED output, filter transmissivity, camera sensitivity and natural sunlight intensity across the visible spectrum. Data from [127] and manufacturer data sheets.

Each vehicle is designed to make relative bearing measurements of other vehicles and landmarks. This means that each vehicle needs to have both a camera equipped to make the measurements and an LED beacon to act as a target for other vehicles. This poses a challenge in mounting the two components to ensure that a vehicle's own LED beacon is positioned in such a way that it is not received by its own camera. Furthermore, the LEDs need to be mounted so that they are visible to other vehicles from as many directions as possible.

On the UAV, we mounted a single downwards-facing camera to the underside of the platform. A custom assembly was fabricated to mount 4 LEDs behind the focal plane of the camera, and thus out of the field of view of the lens. An image of the LED and camera assembly is shown in Figure 6.8.

On the UGV, increased payload capabilities allow us to mount two cameras, one at the front and one at the rear. This provides greater coverage of the horizon, where other UGV and landmark beacons are likely to be detected. An assembly of 5 LEDs is mounted to the centre of the platform in such a way that the LEDs are not within the field of view of either camera. The UGV cameras and LED assembly can be seen in Figure 6.5.

Initial tests on the performance of the LED and camera system were positive. In Figure 6.9, a composite image from all 9 cameras is shown. In each image, the LED beacons of other vehicles are visible as white dots in the image. The 3 downward-facing UAV cameras, in the first column, show that almost all background light from the ground has been filtered out, leaving a mostly black image. The exception is the vehicle's own landing feet, which are illuminated by the vehicle's onboard LED beacon, however, these artefacts are static in the image and can easily be masked out. On the other hand, the cameras on the UGV, shown in the second and third columns, capture a significant portion of the sky. Ordinarily, the atmosphere does not reflect UV light and appears dark in the image, however, clouds are significantly different and scatter UV from the sun, making them appear very bright in the image. In overcast conditions, this can make it difficult to detect the LED beacons against a bright

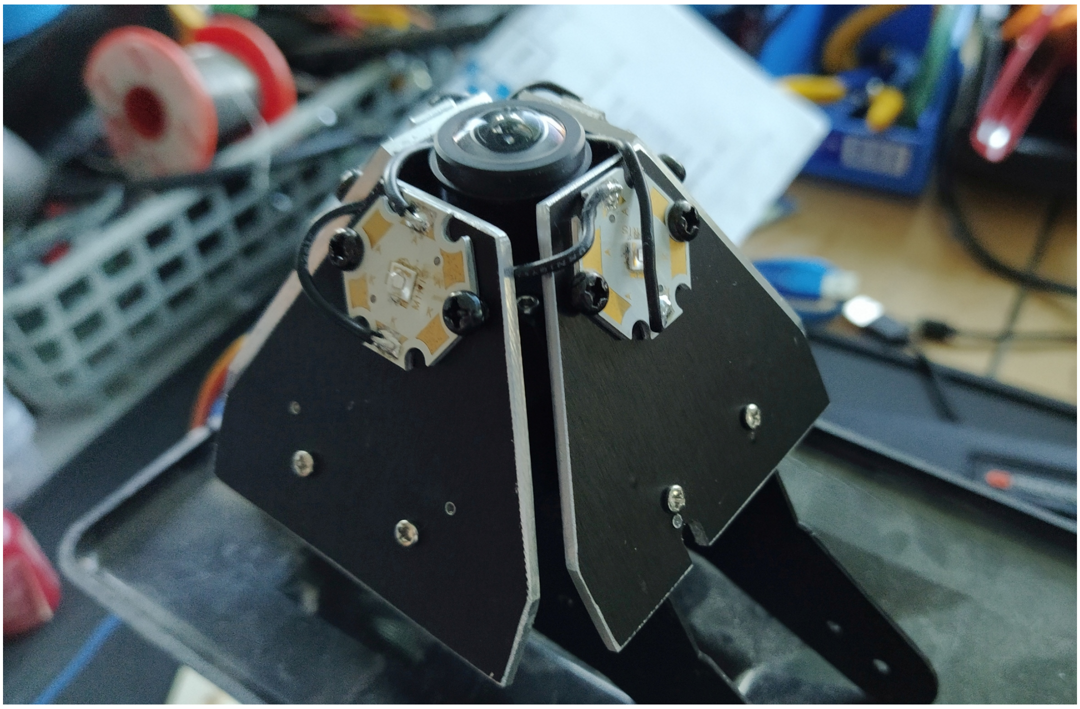


FIGURE 6.8: Close-up view of the LED and camera assembly mounted on the UAV. The camera is mounted to the underside of the UAV and points downwards.

background, but the images in Figure 6.9 show that the LEDs are still clearly visible in most conditions.

6.5.2 LED Marker Identification

In order to be useful for collaborative localisation, the system must provide a way to uniquely identify the target for each bearing measurement that is made. To achieve this, the LEDs flash on and off with a specific pattern that is unique to each vehicle and landmark. By capturing a sequence of images and tracking the LED across multiple frames, the camera system can decode the on-off pattern and identify the target.

The choice of a particular set of flashing patterns is driven by a number of constraints;

- There should be a sufficient number of unique patterns that can be distinguished. For our experimental work, we used 6 vehicles and 8 landmarks, meaning that there should be at least 14 unique patterns.
- The camera used for the experiments has a maximum frame rate of 120 fps, and so the pattern should be compatible with this.
- The tracking of the LED in the camera frame can only be performed while the LED is on. In order to maximise tracking performance, the LED should be illuminated as frequently as possible.

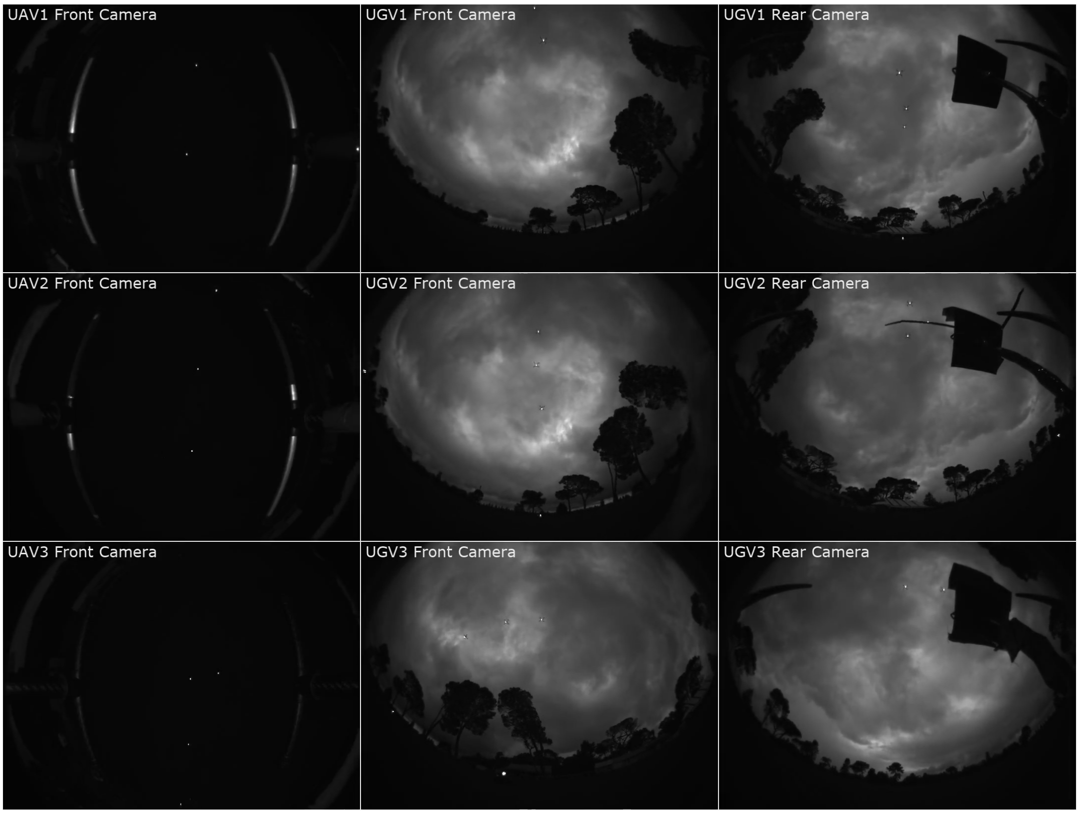


FIGURE 6.9: Composite image of all 9 cameras during a mission. The flashing LED markers of other vehicles are clearly visible as bright white dots in the images.

- In order to determine the flashing pattern, the camera must capture multiple frames and determine if the LED is on or off. If the pattern is long, it will take longer to identify the target and less measurement data will be available. Thus, patterns should be as short as possible to reduce identification time.

From these constraints, a number of different flashing patterns were investigated. The system proposed by Walter, Staub *et al.* [114] used different flashing frequencies to identify each target. This approach is simple to implement and detect but, due to the frame rate of the camera, there are a limited number of unique frequencies that can be used. Using a 72 fps camera, they used four unique flashing frequencies of 6 Hz, 10 Hz, 15 Hz and 30 Hz and were able to reliably identify each target. However increasing the number of targets would require tighter spacing between each frequency, making it more difficult to distinguish between each one.

A report by Kreylos [128] investigated a head-mounted virtual-reality device that uses an infrared LED tracking system to determine the pose of the headset. The system consists of 40 LEDs each flashing with a unique 10-bit binary code. The flashing of the LEDs is synchronised with the frame rate of the camera through a physical cable, allowing each frame to capture one bit of the LED pattern. This method provides low identification times as only 10 frames are required to determine the identifier. Additionally, any minor errors in LED detection can be corrected using digital error correction techniques, as there are only 40 valid identifiers out of 1024 possible patterns.

For our purposes, the concept of using a binary flashing pattern would provide for robust and rapid identification of targets, however, the method of wired synchronisation is not practical. Wireless synchronisation between a camera on one vehicle and LEDs on another target may be possible but can be complex and prone to failure. Also, it would require not just one camera and LED, but all cameras and LEDs on every platform and landmark to be synchronised. While theoretically possible, we avoid synchronisation to simplify the experimental setup and reduce critical failure points.

Inspired by these previous works, we designed our own asynchronous binary flashing pattern and corresponding tracking and decoding system. Due to the asynchronous nature of the LED and camera system, the flashing frequency of the LED and the camera frame rate will not be identical and may vary over time. This, combined with the fact that the exposure time of each frame may be shorter than the interval between frames, means that the camera may miss some flashes of the LED. This is illustrated in Figure 6.10. To guard against this phenomenon, we set the LED flashing frequency to be half the frame rate of the camera and use a 100 % duty cycle, which means that the LED will be detected in at least one frame regardless of the timing. Given that the camera is capable of recording at 120 fps, we set the LED to flash at 60 Hz. Compared with the synchronous method, the effective bit rate of the pattern is halved, increasing the time taken to identify an LED.

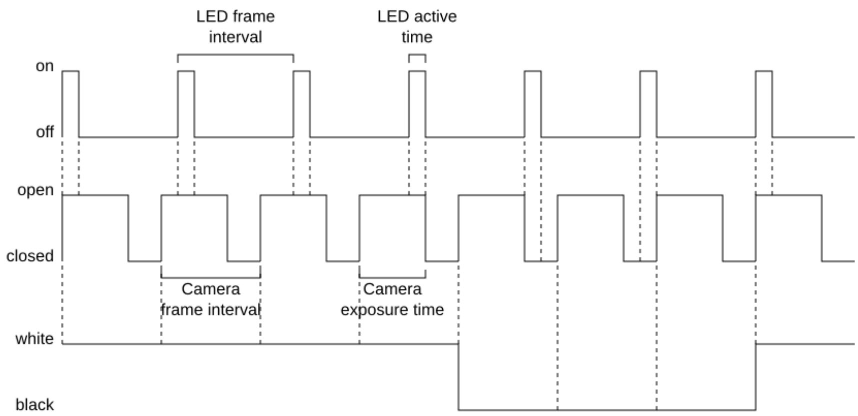


FIGURE 6.10: Illustration of missed detections due to differing camera frame rate and LED pulse frequency. [128]

For the particular values of the binary LED patterns for each vehicle and landmark, we make use of the Bose-Chaudhuri-Hocquenghem (BCH) code [129]. The BCH code was selected for three reasons;

1. It is error-correcting, and configurable to correct a different number of bit errors.
2. It is variable-length.
3. It is cyclic, which means that each rotation (or cyclic bit-shift) of a code word is also a valid code word.

The fact that the BCH code is variable length allows us to configure the level of error correction as well as the number of unique identifiers available. For missions where there are a small number of vehicles and landmarks, a shorter code can be used, or a higher level of error correction could be used. Conversely, for missions with a large number of vehicles, a longer code allows for more unique identifiers.

Also, because the BCH code is cyclic, we can associate all rotations of a given code word with a single identifier which allows the decoder to start decoding from the first detected bit rather than having to wait for the start of the code word. This decreases the time it takes to identify a target to just the length of the code word.

For the missions in our experimental dataset, we have 6 vehicles and 8 landmarks, requiring 14 unique identifiers. We use an 11-bit BCH code with an error correction capability of 1 bit, which gives a code word length of 15 bits. After grouping cyclic rotations of each code word, there are 144 unique identifiers. However, when allocating identifiers to each vehicle, some further considerations must be made. The code words `000000000000000` and `111111111111111` are both valid BCH codes, however, they do not make good LED flashing patterns, as the LED will either be always off or always on, causing it to be undetectable, or confused for another light source in the environment (e.g. the sun). Furthermore, given that we wish to maximise the time that the LED is on to improve tracking performance, code words with

TABLE 6.1: LED flashing patterns assigned to each target

Target	Identifier	Flashing Pattern
UAV 1	645	010111111011111
UAV 2	917	011110111101111
UAV 3	661	010111011101111
UGV 1	677	010110110111111
UGV 2	643	010111110110101
UGV 3	825	011011011011011
Landmark 1	675	010110111010101
Landmark 2	681	010110101101011
Landmark 3	473	001111111111011
Landmark 4	357	001010111111111
Landmark 5	405	001101111101111
Landmark 6	475	001111111011101
Landmark 7	477	001111110110111
Landmark 8	323	001011111110101

a large proportion of zeros should be avoided. Given this, we select 14 code words, detailed in Table 6.1, with the highest proportion of ones to maximise the time the LED is on and improve detection and tracking accuracy.

6.5.3 Image Processing Pipeline

On the receiving vehicle, a computer vision system is required to process the images captured by the camera and produce bearing measurements to the identified targets in the image. The purpose of designing a custom UV-LED beacon and using a corresponding UV band-pass filter on the camera is to allow for a simpler implementation of each of these components. As can be seen in Figure 6.9, the LED beacons on each vehicle show up as very bright points in the image against a mostly dark background, except for areas of sparse cloud coverage, making it relatively simple to detect these points. This allows us to use a simple tracking-by-detection approach, rather than more complex tracking algorithms.

A block diagram of this system is shown in Figure 6.11, which describes the four main components of the system; point detection, track association, LED decoding, and inverse projection. Each frame captured by the camera is independently processed by the point detection algorithm, which results in a set of candidate target locations for each frame. These detected points are passed to the track association process, which matches up spatiotemporally related points to create tracks. Each track represents a single detected target across multiple image frames. By analysing which frames the target is detected and which frames it is not, the unique binary pattern of the LED can be decoded and the target identified. Additionally, with a calibrated camera, the detected points in the image can be mapped to a relative bearing between the

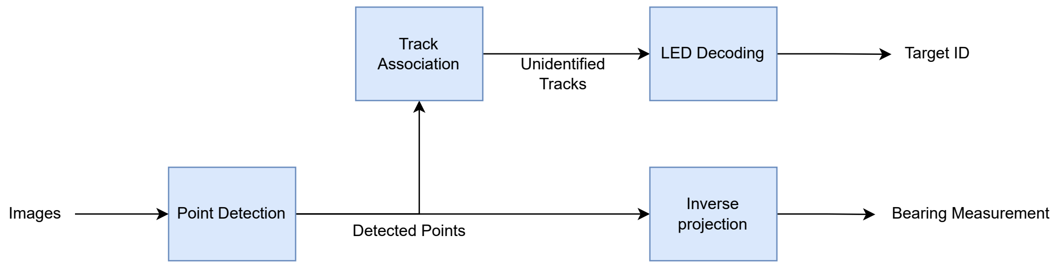


FIGURE 6.11: Block diagram of Bearing measurement subsystem.

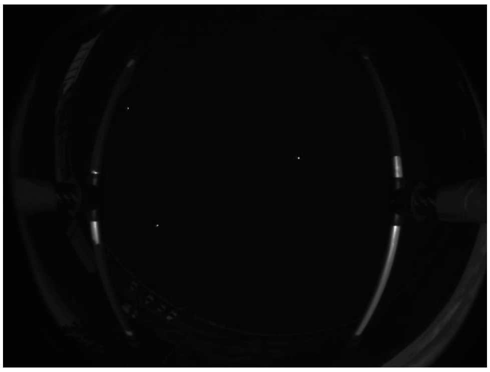
vehicle and the target.

Detection

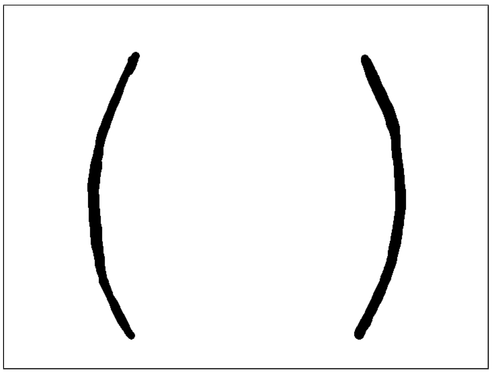
Due to the 185° field of view of the camera lens, some parts of the vehicle are visible in the image. This can be observed in Figure 6.12a, where the downward-facing camera on the UAV captures the landing feet, and in Figure 6.13a where the rearwards-facing camera on the UGV captures the Wi-Fi antennas and GNSS antenna mast. As some of these surfaces are reflective, they can reflect light from the vehicle's own LED beacon into the image, resulting in spurious detections. To address this, we apply a binary mask which is manually created for each camera to mask out any areas where reflections result in false detections. Examples of these masks are shown in Figures 6.12b and 6.13b respectively.

While a simple binary threshold on the image would work well to detect bright points in the image, it would also yield many false positives due to the bright clouds, and would also fail to detect targets at long distances, as they would appear dimmer in the image. Instead, detection of bright spots in the image is performed by determining the prominence[2] of peaks in each image. We use a persistent homology algorithm from the literature [130] to calculate the prominence of all peaks in the image, and then select the peaks with a prominence greater than a chosen threshold. For the data from our experiments, we found a threshold of 80 was sufficient. An example of the points detected for both the UAV and UGV are shown in Figures 6.12c and 6.13c respectively. One can observe in Figure 6.12c, that the three UGV targets are correctly detected with no false positives or false negatives. However, on the UGV, shown in Figure 6.13c, the clouds present a more difficult challenge. While all the targets are correctly detected, three false positives are also detected. These points are indeed prominent peaks in the image but are attributed to the clouds rather than valid LED targets. This does not present an immediate issue, as the subsequent processing steps of tracking and identification should be able to filter out these false positive detections.

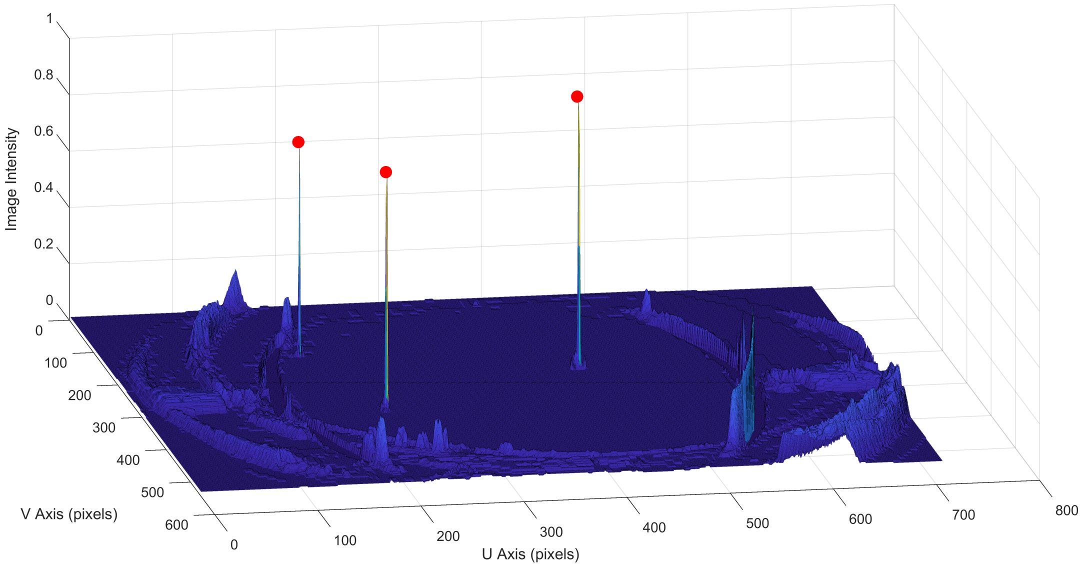
[2]See `https://au.mathworks.com/help/signal/ug/prominence.html`



(A) Example image captured by a UAV during an experiment



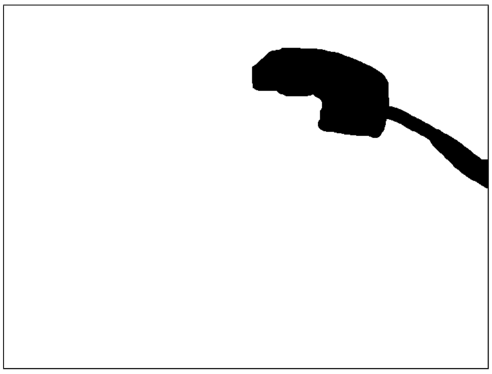
(B) Image mask used to reject reflections from own vehicle



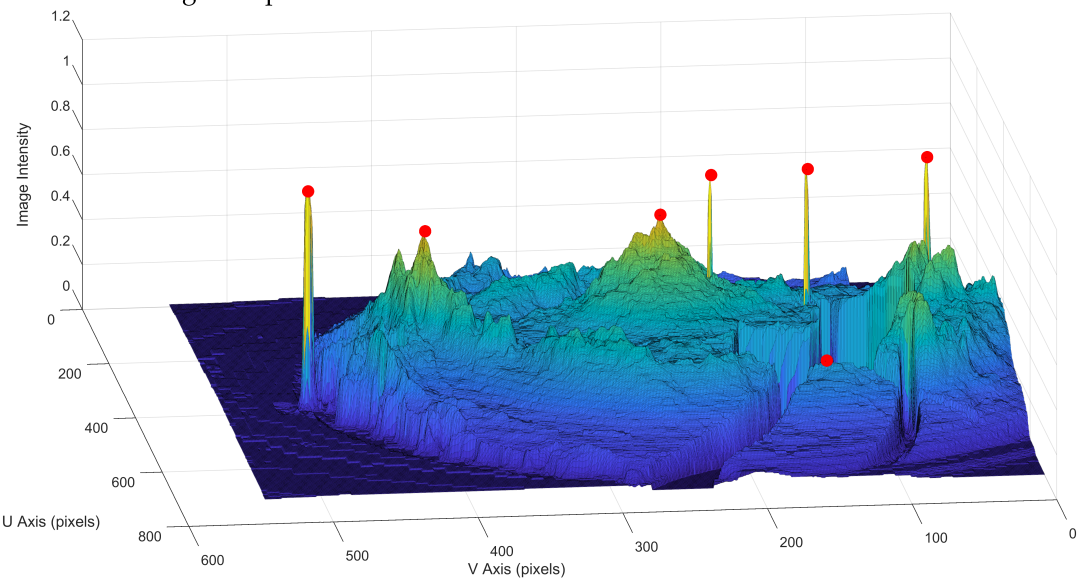
(C) 3D Surface plot of masked image intensity with detected peaks shown in red.

FIGURE 6.12: Example of peak detection on UAV image data. The three UGVs are clearly visible as highly prominent peaks in the 3D plot.

(A) Example image captured by a UGV during an experiment



(B) Image mask used to reject reflections from own vehicle



(C) 3D Surface plot of masked image intensity with detected peaks shown in red.

FIGURE 6.13: Example of peak detection on UGV image data. One UGV and three UGV are detected, but additional false positives are also detected due to light scattered by clouds. Spurious detections of the vehicle's own LED beacon reflected by the GNSS antenna mast are removed using a manually created mask.

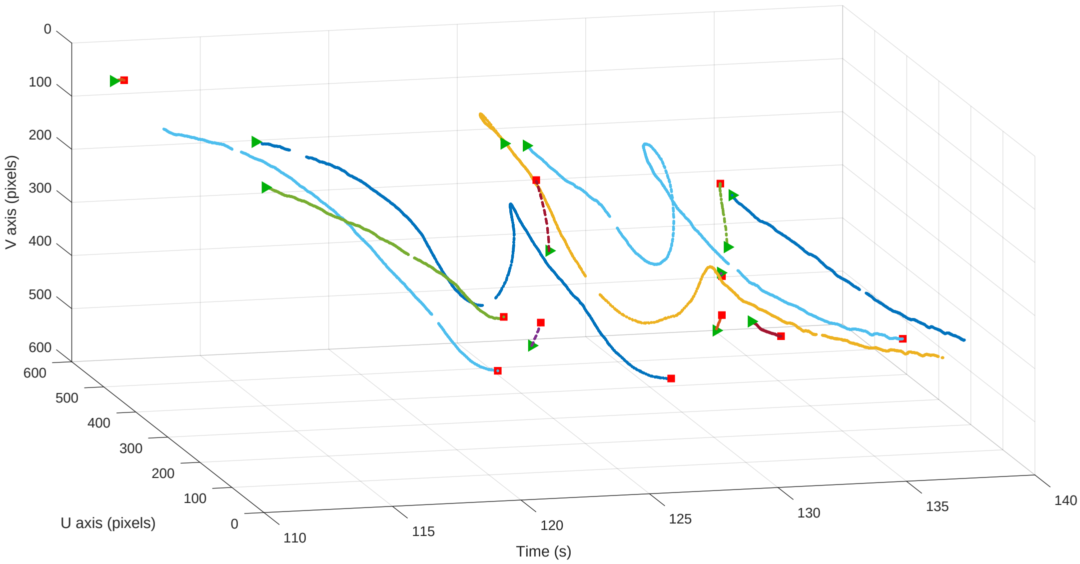


FIGURE 6.14: Example output from tracking algorithm, showing the location of each track over time. The start and end of each track are marked with a green triangle and red square respectively.

Tracking

Once a set of points is detected in each image, we use a tracking algorithm to create correspondences between points in adjacent frames. As the frame rates of each camera are relatively high (120 fps), and the velocities of the targets are comparatively low, we find that a greedy closest-point tracker is sufficient to provide good tracking results. For each frame, we calculate the Euclidean distance between the pixel coordinates of each track in the previous frame and all detected points in the current frame. The detected points are assigned to tracks in a greedy manner, starting with the smallest distance up to a maximum threshold.

Given that the LEDs detected in the image are flashing on and off, it is expected that a target may not be detected for several frames and then re-acquired. To account for this, an additional 'phantom' point is added to the set of detected points in each image at the last detected location of each track. If the point is not re-acquired after a certain number of frames, the track is considered finished and removed from consideration. Additionally, if any detected points in the current frame are not assigned to an existing track, a new track is created.

An example of the output of the tracking algorithm is shown in Figure 6.14. As targets move throughout the environment, their location in the image changes, as shown by the trajectories in the figure. Note that some tracks contain gaps where the target is not detected for some time, likely due to occlusion, but is then re-acquired. In other cases, for example, when the target moves outside the field of view of the camera, the track terminates.

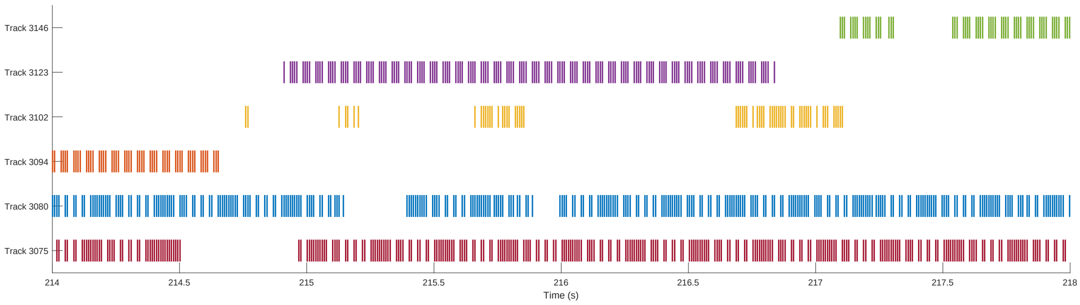


FIGURE 6.15: Example data showing point detections over time for several tracks. Note how each track has a unique pattern of detections, corresponding to the flashing pattern of different LEDs.

Identification

Once a track has been created, the LED flashing pattern needs to be decoded from the image frames. This is a relatively straightforward process as detected points represent a binary 1 and a frame with no detected point represents a binary 0. We simply convert the sequence of detections into a binary sequence and decode the BCH code from the sequence. If the decoded identifier matches a known target, either another vehicle or a landmark, then the track is valid, otherwise it is discarded. This helps to eliminate the false positives from the detection step as it is unlikely that a sequence of false detections will validly decode.

An example of some detected tracks is shown in Figure 6.15, which shows the detected points for each track over time. Recall that we selected the LED flashing frequency to be 60 Hz, while the frame rate of the camera is 120 fps, thus, we can expect a single bit of the LED pattern to be detected in two subsequent frames. Using Track 3123 in Figure 6.15 as an example, we can see the repeating pattern of 4 detections and 2 missed detections, which corresponds to a repeating pattern of `110`. Referring to Table 6.1, we can see that UGV 3 has a flashing pattern of `011011011011011`, and thus we can identify Track 3123 as UGV 3. At approximately $t = 216.8$, Track 3123 terminates as the tracker can no longer associate any detections to that track. Moments later, Track 3146 is created, which has the same flashing pattern as before, so we can identify UGV 3 with this track as well. Note that in order to determine the identifier, a full sequence of 15 bits must first be observed, and so there is some latency between the first detection and the correct identification of the target.

Inverse Projection

In order to produce the relative bearing between the vehicle and the target, we must map the detected point in the image to a point on the unit sphere in the reference frame of the vehicle. This first requires mapping the point from the image frame to the camera frame, and then from the camera frame to the vehicle frame.

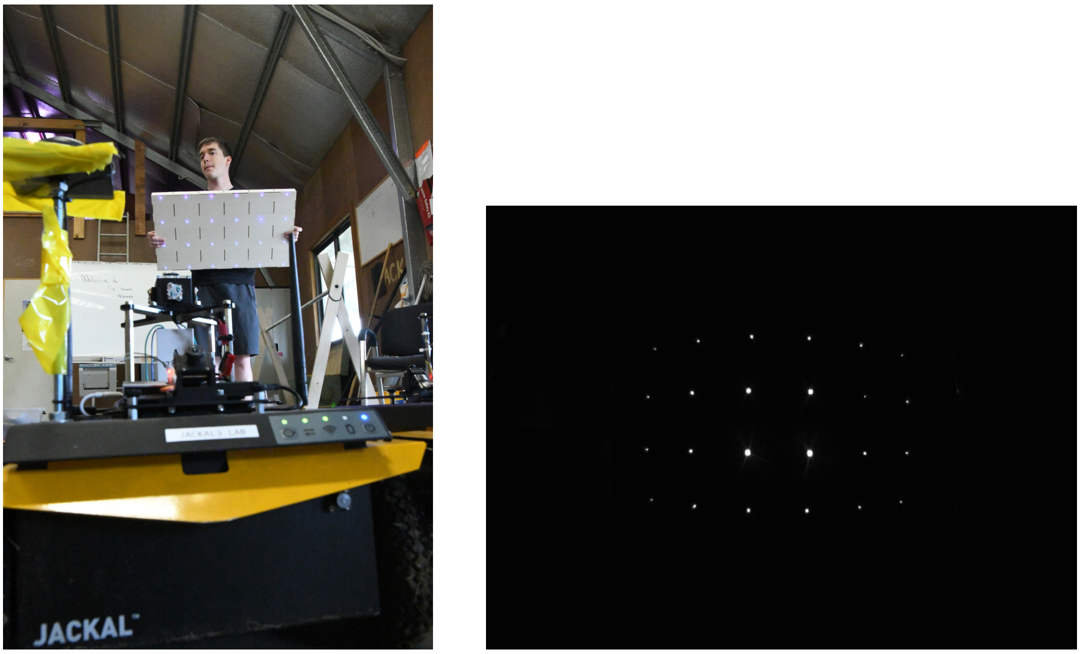
A mathematical model of the camera lens is required to describe how points in 3-dimensional space are mapped into the 2-dimensional image. Given that this is not a one-to-one mapping, the inverse mapping results in a ray through space representing all the points that map to a single location in the image — or in other words, a bearing. The most commonly used camera model is the projective model, however, this model is not suited for lenses with large fields of view, such as the 185° field-of-view lens we are using in this setup. Instead, a more complex model must be used. Our first choice of camera model was the Kannala-Brandt [131] model, as the projection, inverse projection, and calibration functions are built into OpenCV, a popular open-source computer vision library. However, upon calibrating the cameras and examining the results, we found that this model did not work well for our particular setup. Instead, we opted to use the Double Sphere camera model [132] which produced much better results.

Determining the parameters of the camera model requires calibrating each camera/lens combination individually, as each lens and camera will have slightly different characteristics. Typically, calibration is performed by capturing several images of a known planar calibration pattern, such as a black-and-white checkerboard, from a variety of different relative poses. The key points of the calibration pattern, such as checkerboard corners, are detected in each image and an optimisation process is performed to find the set of camera parameters that minimises the re-projection error of these points. This is a common and well-understood process in computer vision, and there is extensive literature on the topic, as well as a range of open-source software that performs the task.

However, due to the inclusion of a UV band-pass filter capturing images of standard calibration patterns proved difficult. Walter, Saska and Franchi [115] encountered similar problems when attempting to calibrate their own UV camera system;

> "For the chessboard-type calibration pattern to be fully visible, the pattern had to be illuminated by a UV light source, with the exposure rate and threshold manually adjusted for the different angles of view, some parts of the pattern became overbright or overly dark depending on the angle of reflection." [115]

Given that we had a total of 9 cameras to calibrate and each of these cameras requires approximately 20 to 30 calibration images, performing such manual adjustments would require an inordinate amount of time. Instead, we built a custom active calibration board with a 6×4 array of UV LEDs, spaced 200 mm apart, which can be seen in Figure 6.16a. The LEDs provide an easily detectable set of calibration points without having to change exposure settings or use external UV lighting. An example calibration image is shown in Figure 6.16b, showing that a simple intensity threshold can be used to detect each calibration point. These detected points can then be passed into existing calibration tools to provide the model parameters for each of the cameras. For the Double Sphere model, we used BabelCalib [133], an open-source tool



(A) (B)

FIGURE 6.16: Camera calibration process showing custom-built active calibration board (a), and the resulting image from the camera (b).

specifically for robust calibration of wide-angle cameras.

The final task in constructing the bearing measurement is to convert the bearing from the camera frame to the vehicle. In order to do this, the pose of the camera in the vehicle frame must be known. An approximate value can be determined from the mounting position of the camera, for example, the camera on the UAV is mounted on the central axis of the vehicle and points towards the ground. However, due to variations in assembly and mounting components, there will be a small difference from the assumed position. To correct this, we examined a subset of the experimental data and compared bearing measurements produced by the camera system against the expected bearing values derived from the ground truth data. Using the estimated pose as a starting point, we refined the orientation using an optimisation routine to determine the orientation of the camera which minimises the mean squared error of the bearing measurements.

All of these components combine to be able to produce relative bearing measurements with corresponding target identifiers. In Chapter 7, we will demonstrate this system under real-world conditions and examine the resulting measurement characteristics.

6.6 Range Sensor

To complement the bearing measurements, we also incorporated a system to provide range measurements to other vehicles and landmarks. This system

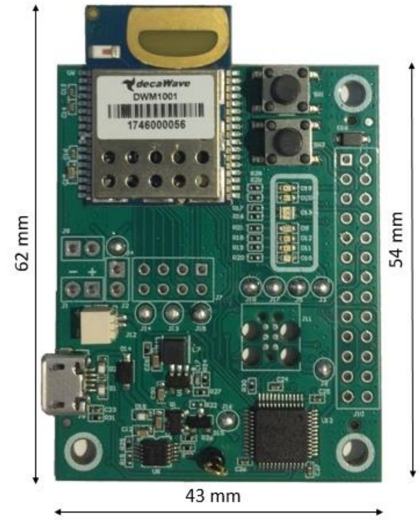


FIGURE 6.17: Decawave DWM1001-DEV Development Board. The DWM1001 module is mounted in the top left corner of the board. [134]

uses off-the-shelf hardware but with modified software to suit the particular use case of inter-vehicle and vehicle-to-landmark range measurements.

We used the Qorvo (formerly Decawave) DWM1001-DEV ultra-wideband (UWB) modules, shown in Figure 6.17. These modules utilise radio signals with very high bandwidths (500 MHz) to perform time-of-flight (TOF) measurements to other modules. As radio waves travel at the speed of light, an error of one nanosecond in the time-of-flight measurement corresponds to a range error of 300 mm and thus highly precise timing is required in order to achieve accurate results.

An illustration of the UWB two-way ranging (TWR) protocol is shown in Figure 6.18. It shows Node 1 broadcasting a ranging request at t_0, which is received by Node 2 at t_1 after some delay which is proportional to the distance between the nodes. Node 2 then prepares a response message and broadcasts a reply at t_2 which is received by Node 1 at t_3 after the same delay time. Assuming that the time for Node 2 to prepare the response message and transmit the reply $(t_2 - t_1)$ is known, the distance, d, between the two nodes can be computed by

$$d = \frac{(t_3 - t_0) - (t_2 - t_1)}{2} c \tag{6.1}$$

where c is the speed of light.

Each DWM1001-DEV board contains a microcontroller that controls the UWB radio and processes messages. Example software is provided by the manufacturer for a selection of possible use cases. For example, a common application of UWB sensors is in indoor localisation in which there are a set of stationary nodes, called anchors, and a set of mobile nodes, called tags. The tags measure their range to each of the anchors and use the principle of trilateration to

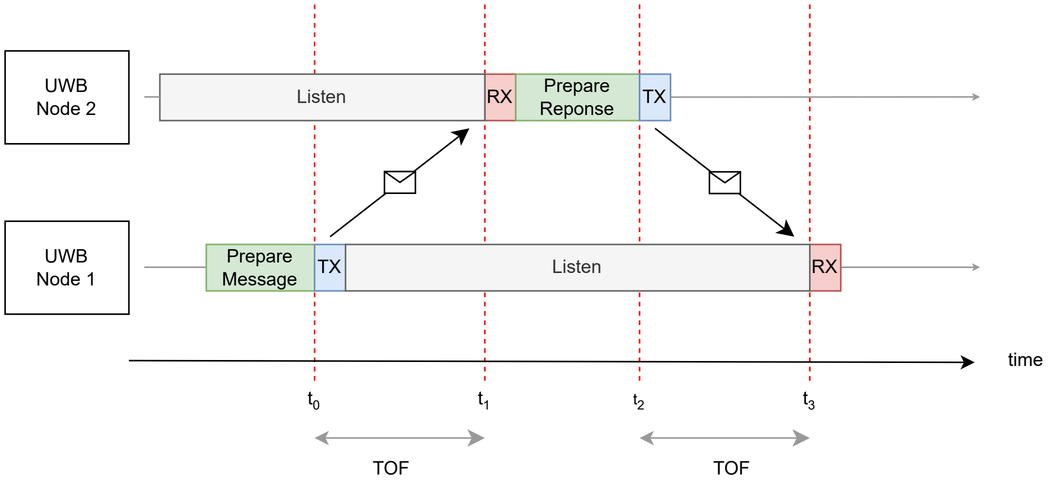


FIGURE 6.18: UWB Two-Way Ranging protocol

determine their position. This provides a useful foundation, as the concept of an anchor and a tag map analogously to a landmark and a vehicle. However, in our case, we also require range measurements between two vehicles, or two tags, which is not supported by the existing example software.

We utilised the open-source software provided by the manufacturer[3] as a foundation for the microcontroller code but made further modifications to support range measurements between tags. To avoid confusion, we will instead refer to the two node types as *active* and *passive* nodes, rather than anchors and tags. Active nodes are affixed to vehicles and make range measurements to all other nodes in the network, while passive nodes are affixed to landmarks and do not initiate any ranging requests themselves, only responding to ranging requests from active nodes.

6.6.1 TDMA Protocol

As each UWB node utilises the same frequency spectrum, there can only be one node transmitting at a time. Thus, in a network of more than two nodes, a further protocol on top of TWR is required to coordinate access to the shared spectrum and avoid collisions between messages. There are many ways this can be performed, for example, the ALOHA or carrier-sense multiple access (CSMA) protocols [135]. These protocols require no central coordination and thus can scale up to large numbers of nodes very effectively, however, they suffer from poor utilisation of the spectrum and frequent collisions if spectrum usage is high. Hillam [136] demonstrated a CSMA protocol for UWB networks and was able to achieve a rate of 167 range measurements per second across a network of four nodes. Higher measurement rates can be achieved through coordinated protocols, such as time division multiple access (TDMA), in which each node is assigned specific times to transmit messages.

[3] https://github.com/Decawave/uwb-apps/tree/master/apps/twr_nranges_tdma

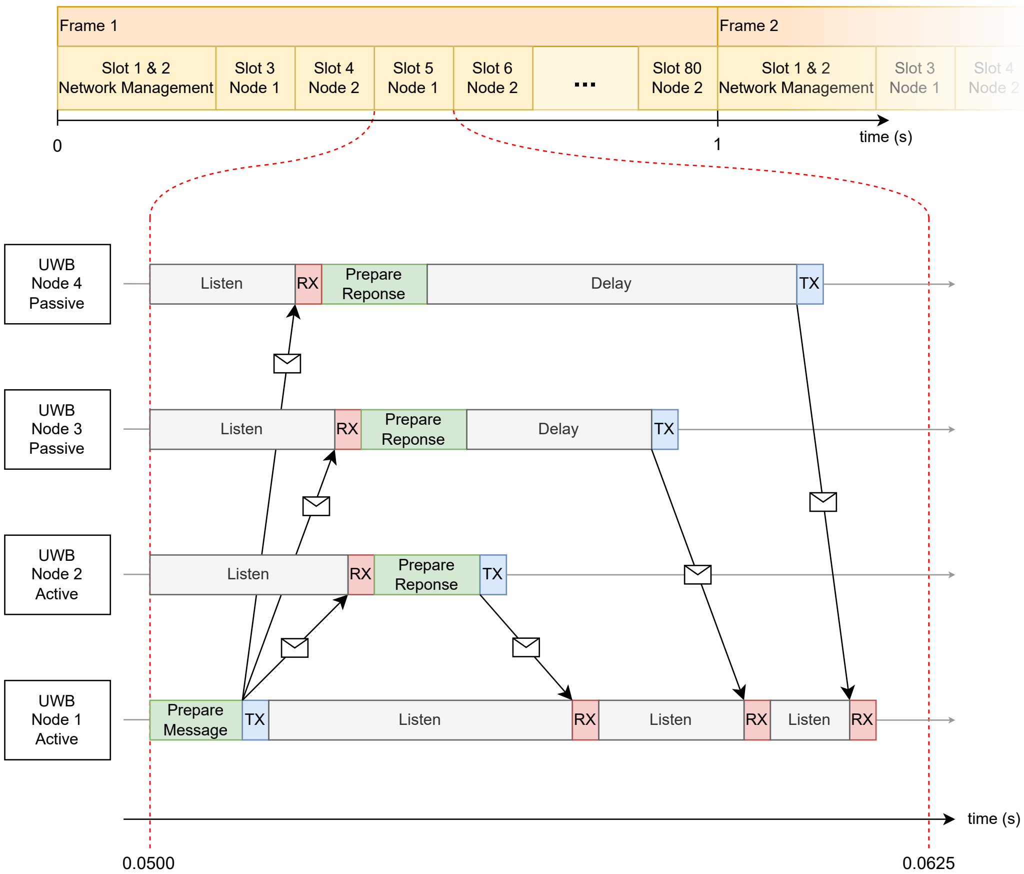


FIGURE 6.19: Illustration of TDMA protocol used for UWB nodes. For simplicity, an example network is shown with 2 active nodes and 2 passive nodes, while in the full-scale experiments, there are 6 active nodes and 8 passive nodes. Times are not to scale.

The example code provided by the manufacturer contains an implementation of the TDMA protocol which we further adapt for our use case. The TDMA protocol used divides up time into 1-second long frames, with each frame divided into 80 slots. We assign one landmark as the master node which controls the timing of the protocol and defines the start time of each frame. The first two slots in each frame are reserved for time synchronisation between the nodes, and coordination of slot allocations. The remaining slots are allocated in a round-robin fashion to each active node in the network. Within each of these slots, the specified active node broadcasts a ranging request and every other node in the network sends a reply message. The timing of the replies is staggered by a different offset for each node, which is assigned by the master node so that the replies do not collide. An illustration of the TDMA protocol is shown in Figure 6.19.

With 6 active nodes and 8 passive nodes in the network, representing 6 vehicles and 8 landmarks, each active node will receive 13 slots per frame. If all nodes are within range, then each node can potentially measure the range to 13 other nodes in every slot. This provides each vehicle with 169 range measurements per second and a network total of 1014 measurements per second. In reality, the measurement rate will be somewhat lower than this as some failures are likely, for example, if a node is out of range.

6.6.2 LED Control

The UWB sensor also serves a secondary purpose as the control circuit for the LED beacon. Given that a UWB sensor is required for each vehicle and landmark beacon, rather than adding a dedicated microcontroller to control the LEDs, we simply use one of the general-purpose input/output (GPIO) pins from the UWB board. This simplifies the landmark beacon assembly and ensures a common control method across both the vehicles and landmarks. Each UWB is configured with an identifier from Table 6.1 and repeats the binary pattern at 60 Hz on the GPIO output pin which is connected to the LED driver.

6.6.3 Limitations

Initial testing of the UWB sensors proved promising, indicating a maximum distance between nodes of approximately 100 m before measurements dropped out. However, further testing revealed several further limitations of the UWB system.

The antenna in the UWB module is vertically polarised, which means that in order to achieve the best range all nodes should be mounted vertically. The radiation pattern of the antenna provides the best signal strength between nodes in the XY plane, such as between landmarks and UGVs. However, the antenna has several areas of low sensitivity, especially in the Z direction. Thus, we found reduced performance and a lower rate of measurements for the UAVs especially when they were at high altitude directly above another node.

Another limitation of the UWB sensor is that it requires line of sight (LOS) between nodes and is susceptible to multipathing. The high frequencies used by the UWB sensor (3.5 GHz to 6.5 GHz) are heavily attenuated by solid objects, which can result in loss of measurements. In some cases, if the direct line of sight is occluded, the strongest signal received by a node may be a reflection from a nearby surface, a phenomenon known as multipathing. Shown in Figure 6.20, this results in a longer time of flight and provides erroneous range measurements.

Due to the centralised design of the TDMA protocol, each node must be within range of the master node lest it risk losing synchronisation with other nodes, causing conflicts. This could be mitigated by propagating synchronisation messages in a multi-hop setup, but this would still require network topology to remain connected. For the experiments we performed, this limitation did not prove to be a major issue as all vehicles remained within proximity of each other, but it should be a consideration for future large-scale experiments.

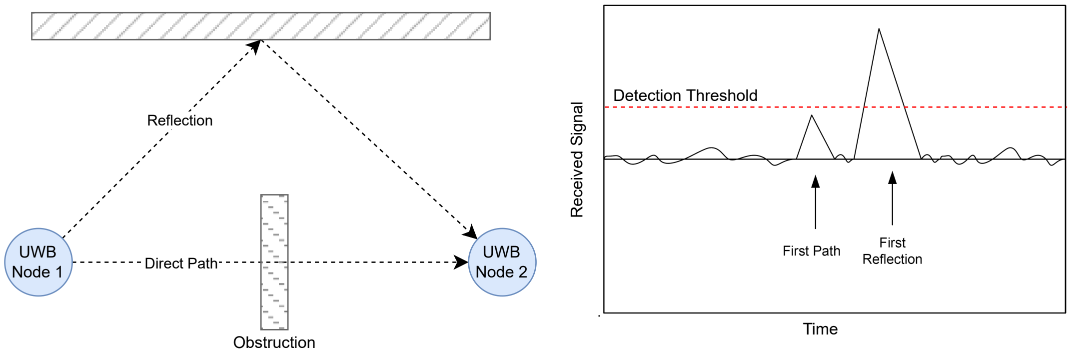


FIGURE 6.20: Diagram illustrating multi-path issues for UWB sensor. In the example shown, the peak detected by the receiver is the result of a reflection off a nearby surface. The distance measurement calculated by the receiver will be longer than the true distance between the two nodes. Adapted from [137].

The accuracy of the range measurements is explored in more detail in Chapter 7, however, multiple factors introduce errors in the measurements. These are well documented in the manufacturer's technical note [138], describing the effects of clock drift, received signal level, and antenna delays on the measurement accuracy.

6.7 System Integration

Each of the sensor subsystems discussed above, the GNSS receiver, UWB sensor, cameras, and flight controller, need to be integrated together to control the vehicles, monitor system health, and record the sensor data. This is primarily achieved through the onboard compute module which runs Robot Operating System (ROS), a modular, open-source system that uses a publish-subscribe messaging model to manage the interactions between software components (called ROS nodes) within a robotic system. The diagram in Figure 6.21 details each of the ROS nodes that run onboard the vehicles and the components that they interact with. For each sensor, there is a corresponding ROS node that reads the sensor data and publishes the data through the ROS interface. The data recording node then listens for these messages and records all the relevant sensor data in a single file onboard each vehicle (called a ROSBAG).

Each of the vehicles is connected via Wi-Fi to the ground station laptop, which can monitor the sensor data on each vehicle and identify any errors or configuration issues. An example of the tools used on the ground station is shown in Figure 6.22. A 4G cellular modem within the Wi-Fi router allows NTRIP data to be streamed from the internet to the GNSS receivers on each vehicle, which enables the high-precision RTK fix.

During a mission, the ROS sensor data is recorded onto a USB flash drive on each vehicle. While it is possible to stream telemetry data in real-time

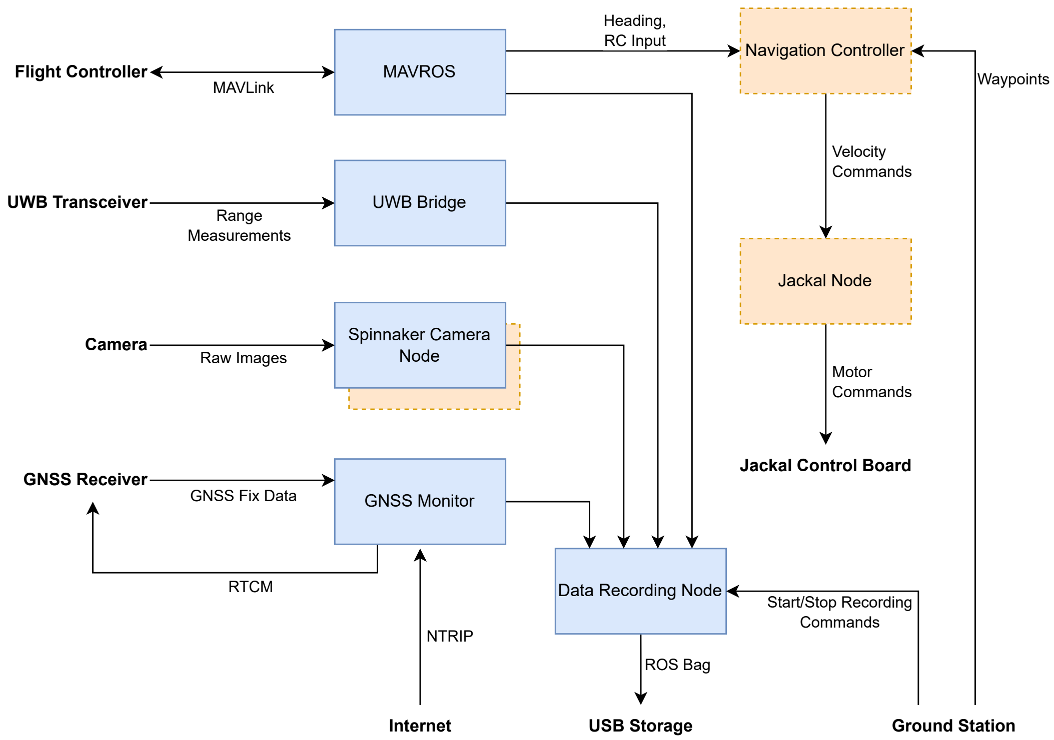
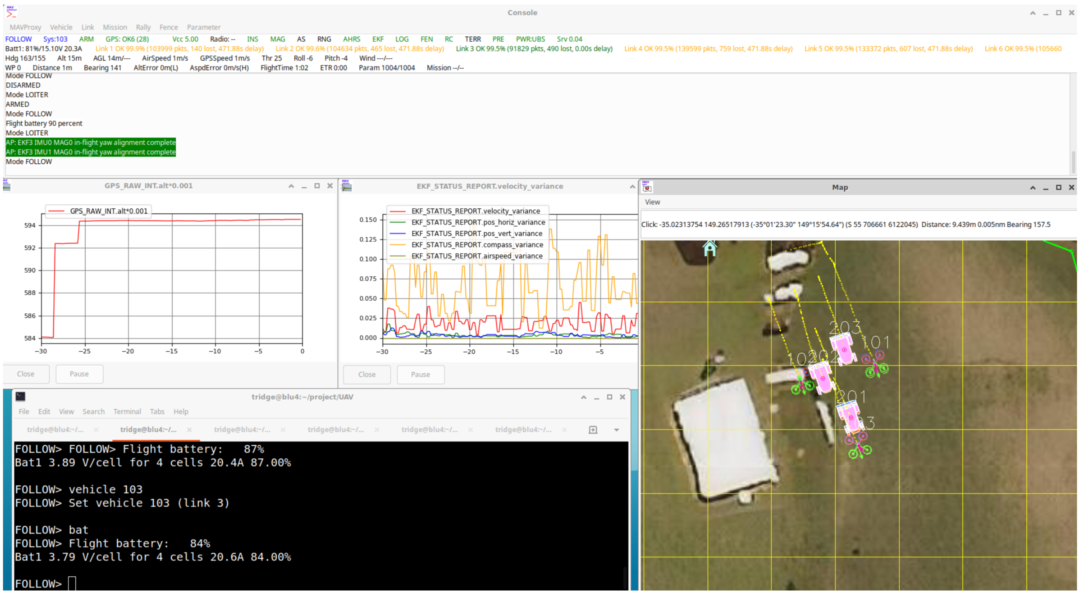
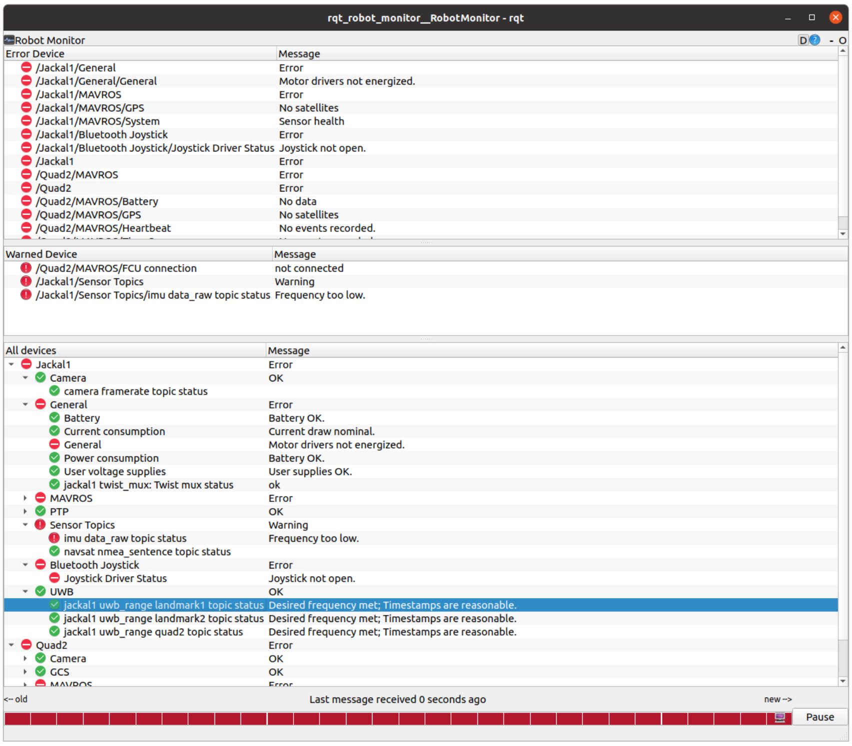


FIGURE 6.21: Software block diagram for ROS nodes running on the onboard computer of each vehicle. Blue solid boxes indicate nodes common to both UAV and UGV, while orange dashed boxes indicate nodes only run on UGV computers.



(A) MAVProxy



(B) ROS Robot Monitor

FIGURE 6.22: Screenshots of ground station monitoring tools

from the flight controller to ROS using the MAVLINK protocol, this is treated as a low-priority task for the flight controller's real-time operating system. Additionally, there is a bandwidth limitation on the serial connection between the flight controller and the onboard computer. Both of these issues led to reliability, latency and throughput issues when trying to record the full-rate IMU data and other Ardupilot telemetry through ROS.

Rather than recording the IMU data through ROS, the flight controller records its own sensor data into its internal micro-SD card. At the conclusion of a mission, the data files from each vehicle, both the ROS data and the flight controller data, are copied onto the ground station. The data files are then processed and merged to form a single data file for the mission which contains the sensor data from every vehicle.

6.7.1 Time Synchronisation

One challenge in performing this merging process is that the timestamps of the measurements are all recorded on different vehicles with potentially differing clocks. Initially, we attempted to synchronise each of the vehicle clocks to the ground station laptop using the precision time protocol (PTP) over the Wi-Fi network (a protocol similar to the network time protocol, or NTP). However, this solution was prone to errors caused by variable latency in the Wi-Fi network and the lack of a stable master clock on the laptop. GNSS time synchronisation was also considered, which would provide an accurate common time reference for all vehicles. However, the software and hardware implementation of this proved too complex, especially when trying to incorporate the pulse-per-second input signal to improve timing precision. Instead, we opted to perform the time synchronisation in the post-processing stage. Each data file contains GNSS position measurements which are timestamped with both the system time and the GNSS time. Given that the system clock may drift over time relative to the GNSS time, we perform a linear regression on the offset over time. This is then used to convert all the measurement timestamps from system time to GNSS time and ensure all vehicles are operating with a common clock. A similar process is also performed on the flight controller data to align the IMU measurement timestamps.

6.8 Summary

The fleet of autonomous vehicles described in this chapter provides a unique capability for capturing heterogeneous, three-dimensional, inertial collaborative localisation data. The sensor suite equipped on each vehicle provides accurate ground truth position data, high-rate inertial measurements, relative range and bearing measurements to landmark beacons, and most importantly, relative range and bearing measurements between vehicles in the fleet. By utilising off-the-shelf components where possible and designing and fabricating custom components where necessary, we can create a robust high-performance fleet at a reasonable cost. The capabilities of this fleet will be demonstrated in the following chapter, where we aim to use the fleet to create a new collaborative localisation dataset.

Throughout the process, countless technical challenges and issues were addressed, and while we have attempted to document the most important and relevant ones in this chapter, countless others are not mentioned. It is hoped that this document may serve as an explanatory guide for those who continue to work with this fleet in the future, or those who are designing similar systems.

Chapter 7

Creating a New Collaborative Localisation Dataset

Using the UGV and UAV platforms described in the previous chapter, we designed and executed a series of experiments to create a new dataset for testing collaborative localisation algorithms. This chapter is organised in two parts — the first part describes the experimental setup and mission profiles, including some preliminary analysis of the sensor data collected while the second part uses the experimental data to evaluate the performance of the collaborative localisation algorithms that we proposed in Chapter 4.

7.1 Data Collection

The aim of the experiment was to conduct a series of missions in which the UAVs and UGVs manoeuvre in proximity to one another in various formations and trajectories. The sensor data captured by each vehicle was collected and combined into a single data file for each mission.

The experimental data was collected over two days with the assistance of a team of volunteers[1]. Due to the safety implications of operating several large UAVs in close proximity to one another, a thorough planning process ensured all risks were appropriately managed. Gundaroo Park, shown in Figure 7.1, is a large field in the town of Gundaroo near Canberra, Australia, and was selected as a safe and remote location to conduct the experiments. At approximately 160 m in diameter, it provided ample room to manoeuvre the UAV and the flat grass is ideal for the UGV to operate on.

The 8 landmark beacons were mounted to tripods and randomly positioned throughout the field and remained in the same location for each experiment. Unfortunately, due to supply chain issues, some components for the landmark beacons were unable to be sourced in time for the experiments. This meant the UV LED beacon, used for the relative bearing measurements, was not functional on the landmark beacons. However, the UWB sensor, used for range measurements, was functional on all landmark beacons. While this

[1] Refer to Acknowledgements section on Page ii

FIGURE 7.1: Gundaroo Park, the location where the experiments were performed.

was not an ideal situation, we discuss ways to address the missing bearing measurements during the post-processing phase in Section 7.4.2.

After performing setup, calibration and initial testing, we conducted 5 missions with different vehicle trajectories. These are described below.

7.1.1 Mission 1

For the first mission, we aimed to construct a scenario that resembles the existing non-collaborative datasets where there is no interaction or coordination between vehicles. In this mission, each vehicle independently navigated along a path defined by a pre-programmed set of waypoints. To reduce the chance of a collision, each UAV flew at a constant altitude; UAV1 at 10 m, UAV2 at 20 m, and UAV3 at 30 m above ground level. Each UAV was programmed with a set of 4 unique waypoints while all the UGVs were assigned to the same set of 8 waypoints but with staggered starting times to avoid collisions. A visualisation of the trajectory of each vehicle is shown in Figure 7.2.

This mission most closely resembles the semi-synthetic dataset we constructed in Section 5.1 using separate EuRoC mission data. However, in this mission, we have real-world inter-vehicle measurement data because the vehicles were actually operating in the same environment at the same time.

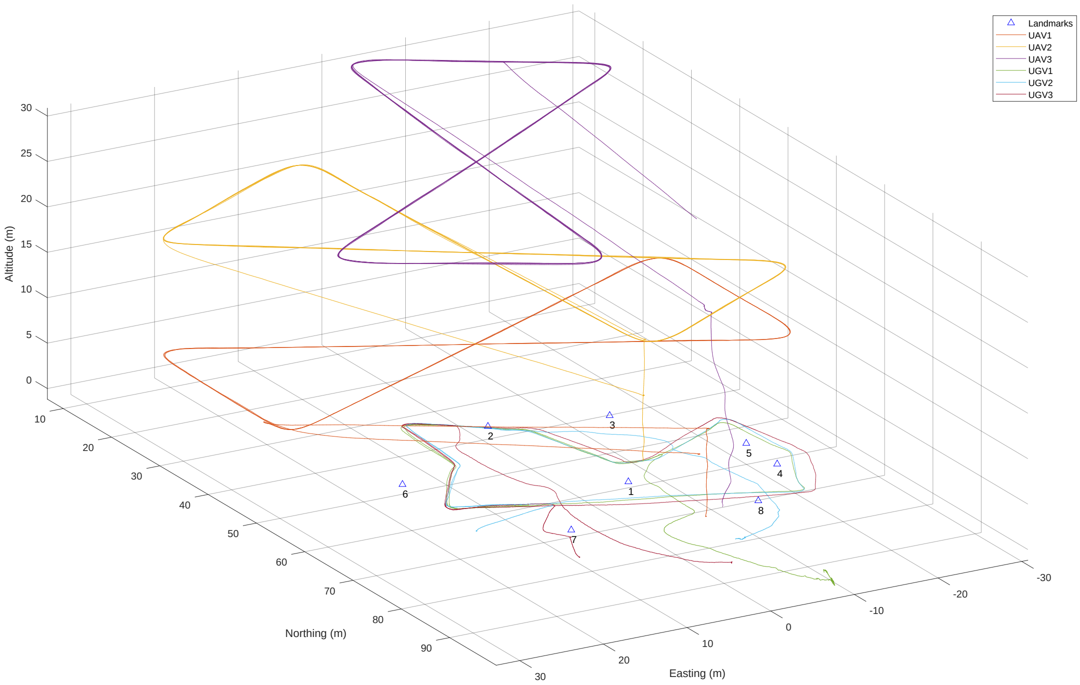


FIGURE 7.2: Visualisation of the trajectory of each vehicle for Mission 1

7.1.2 Mission 2

In Mission 2, we constructed a scenario that contains a mix of independent behaviour and coordinated movement. The vehicles were split into three groups, with each group containing one UGV and one UAV. In each group, the UGV navigated along the same path as in Mission 1 and the UAV was programmed to fly directly above the UGV at a fixed altitude. Thus, each group maintained a constant relative pose between members but was independent of the other groups. The vehicle trajectories are shown in Figure 7.3. Again, for safety, each UAV flew at a different fixed altitude, with UAV1 at 10 m, UAV2 at 15 m and UAV3 at 20 m.

7.1.3 Mission 3

In the third mission, we demonstrated a scenario where all vehicles move together in a formation. UGV1 acted as the lead vehicle and was manually driven along a meandering trajectory through the operational area. UGV2 and UGV3 are also manually controlled, aiming to follow approximately 5 m behind UGV1 and offset to the left and right, forming an 'arrowhead' formation. The UAVs created a similar formation directly above the UGV, with UAV3 programmed to fly directly overhead UGV1, and the other two UAVs automatically following 10 m behind and 5 m to the left and right respectively. In this mission, the lead UAV flew at an altitude of 15 m and the other two UAVs flew at 13 m. Figure 7.4 shows the vehicle trajectories.

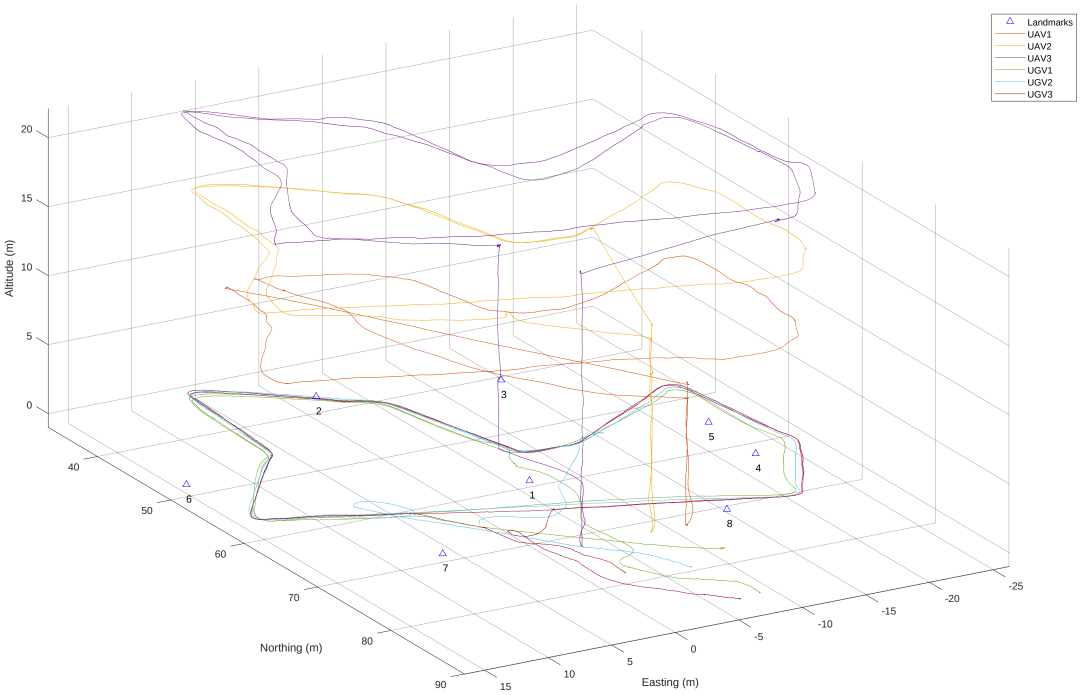


FIGURE 7.3: Visualisation of the trajectory of each vehicle for Mission 2

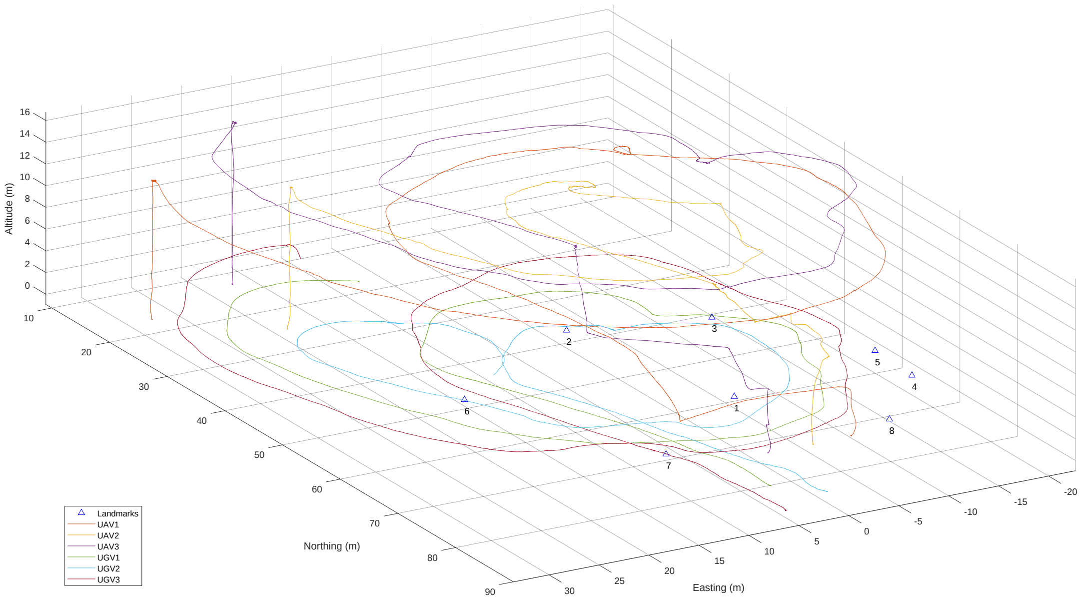


FIGURE 7.4: Visualisation of the trajectory of each vehicle for Mission 3

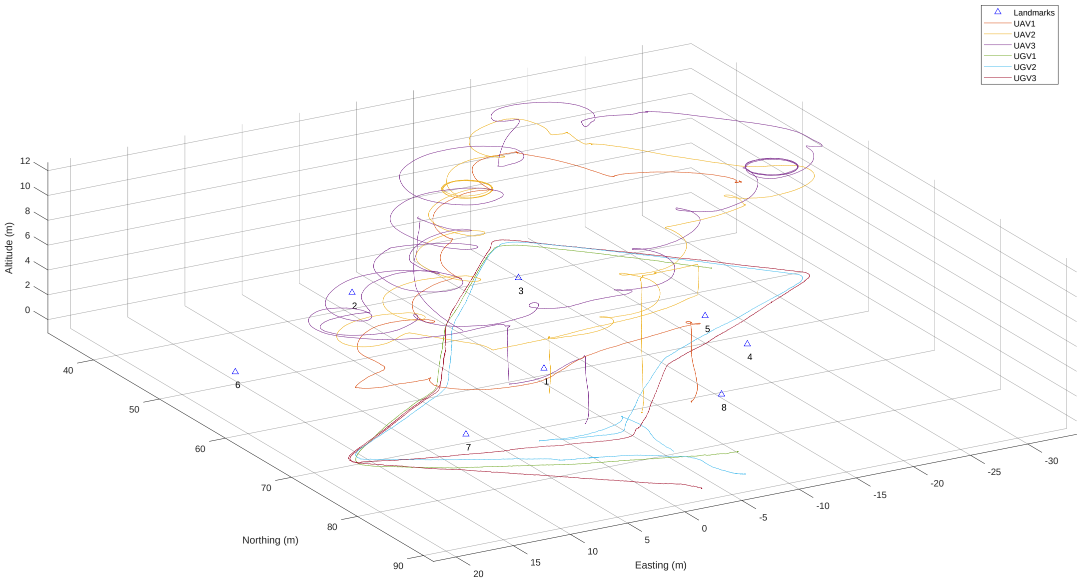


FIGURE 7.5: Visualisation of the trajectory of each vehicle for Mission 4

7.1.4 Mission 4

Mission 4 is a variation on Mission 2, designed to introduce variation in the relative poses between vehicle pairs, while still maintaining the grouping. Similarly to Mission 2, the vehicles were divided into three groups, each with one UGV and one UAV. Each UGV followed a path defined by a sequence of waypoints and the UAVs followed at a fixed altitude above. However, instead of the UAV following directly above the UAV, the UAV continually orbited in a circle with the centre point defined as the UGV's current position. Compared with Mission 2, each UAV also flew at a much lower altitude and separation, with UAV1, 2 and 3 flying at altitudes of 7 m, 9 m, 11 m respectively. The trajectories of each vehicle can be seen in Figure 7.5, which shows the unique trochoid-like trajectories that are formed when the UAVs orbit around moving points. Due to some occurrences of packet loss, there are intervals within the mission where the UAV does not receive the updated position of the UGV it is following and does not correctly orbit the right point.

7.1.5 Mission 5

The final mission within the dataset is similar to the arrowhead formation of Mission 3 but with a different formation structure. The formation used in this mission is linear, similar to how vehicles might drive in a convoy. The formation was led by UGV1, with UGV2 following 5 m directly behind and UGV3 a further 5 m behind. The UAVs follow the same pattern above, with altitudes of 12 m, 14 m, and 16 m respectively. As all the vehicle positions were close to being co-planar, this mission may represent a more challenging scenario for collaborative localisation algorithms.

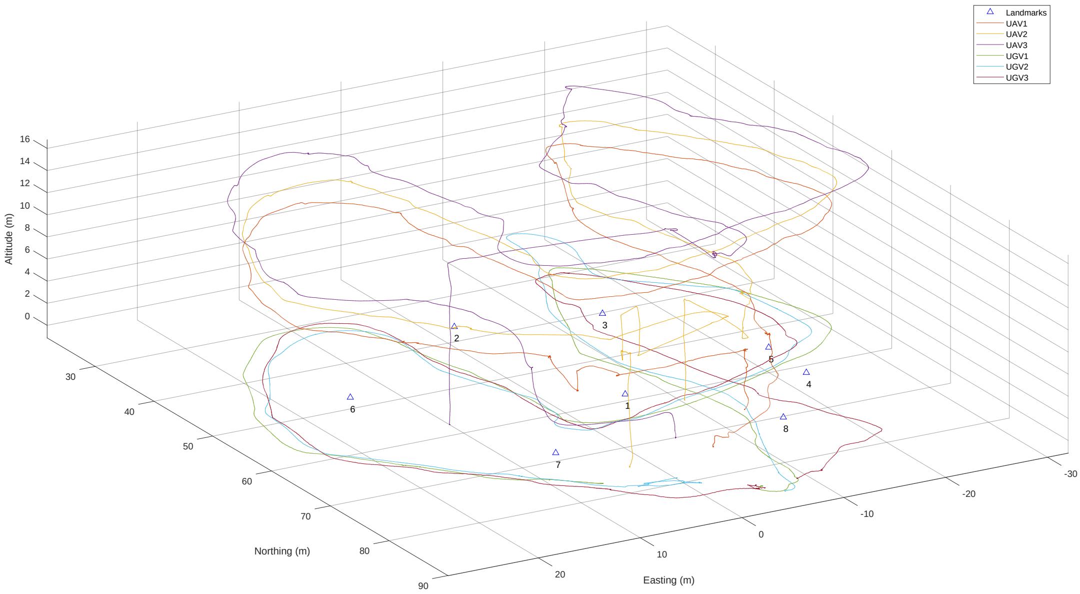


FIGURE 7.6: Visualisation of the trajectory of each vehicle for Mission 5

TABLE 7.1: Statistics from each mission

	Recording Duration (s)	Mission Duration (s)	Mean Distance Travelled (m)
Mission 1	272	141	569
Mission 2	425	113	280
Mission 3	374	176	226
Mission 4	270	86	213
Mission 5	413	205	274

In Table 7.1, we show the duration of each mission and the mean distance travelled by the vehicles. The 'recording duration' includes the time required for each of the UAVs to take off and land, as well as the time required for the vehicles to move into their formations or starting points. In many cases, a user may wish to exclude this data and only focus on the sub-interval of time when all the vehicles are executing the mission. This length of this interval is denoted as the 'mission duration'.

As an indication of the number of measurements recorded by each sensor, we show a summary in Table 7.2. From this, we can observe that the IMU and GNSS measurements were consistently sampled at the same fixed intervals on each vehicle, while the range and bearing measurements were highly variable and depended on the particular properties of the sensor. We will examine these sensors in more detail in Section 7.3.

TABLE 7.2: Number of sensor measurements recorded in Mission 1

	IMU	GNSS	Range	Bearing
UAV1	108,800	2,721	12,929	139,053
UAV2	108,800	2,719	5,753	47,503
UAV3	108,800	2,719	5,506	46,542
UGV1	108,800	2,721	31,900	101,718
UGV2	108,800	2,718	16,888	104,138
UGV3	108,800	2,720	33,234	103,442

7.2 Data Processing

After the experimental data was collected in the field, further post-processing was required to transform the raw data into a single easy-to-use data file containing sensor data from every vehicle, ground truth data, and metadata. As discussed in Section 6.7, each vehicle recorded sensor data locally, and the first step in the post-processing pipeline is to merge the 12 different data files (one ROSBAG and one Ardupilot log file per vehicle) into a single file.

After considering multiple options, we selected the Hierarchical Data Format (HDF5) as the format for the mission data. HDF5 is a widely supported scientific data format that allows for the efficient storage of collections of numerical data in a structured and well-defined manner. Almost all programming languages provide support for reading and writing HDF5 files, including MATLAB, Python and C++, making it interoperable and platform-agnostic. The format also allows for embedded user-defined metadata, which makes it easy to document additional parameters such as sensor calibration data, noise parameters, or experimental configurations. An example of one of the mission data files is shown in Figure 7.7, showing how the data has been organised within the hierarchical structure.

Once the raw experimental data had been collated into the HDF file, we performed some additional processing steps. This included processing the camera data and running the bearing measurement pipeline, filtering out some erroneous sensor data, and creating secondary sensor data by merging the range and bearing measurements together.

7.2.1 Image Processing

Due to the complexity of the experimental procedure and computational resources available, the camera data processing had to be performed in the post-processing stage, and not in real-time onboard each vehicle. This does mean that the system is not yet capable of running the collaborative localisation algorithms onboard in real-time. However, further development and improvements to the computer vision pipeline would enable real-time bearing measurements, and could potentially enable collaborative localisation algorithms to run onboard in real-time.

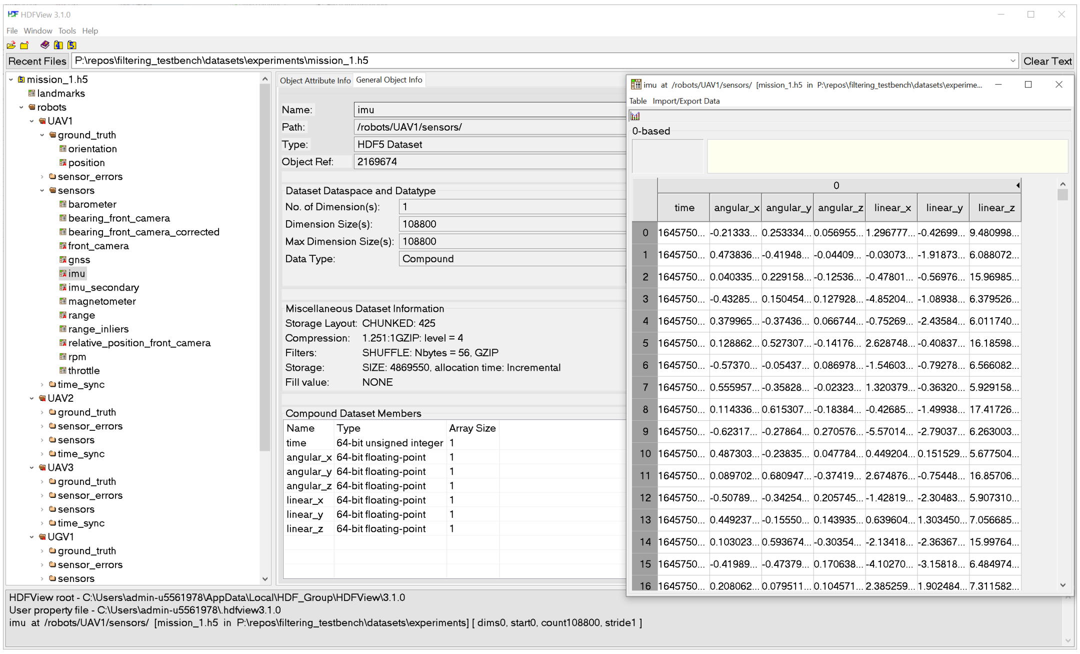


FIGURE 7.7: Example HDF5 file within the HDFView graphical user interface. Here we are viewing the data from Mission 1, and we can see the collection of individual data arrays in the tree structure on the left. The centre region shows specific information about the IMU data of UAV1, and the pop-out window on the right shows the raw data.

From the camera data that was stored in the ROSBAG on each vehicle, we extracted all the image frames and ran the image processing pipeline, described in Section 6.5.3, which generated the relative bearing measurements. This measurement data was then stored back in the mission HDF file.

7.2.2 Filtering Invalid Measurements

After an initial analysis of the experimental data, several issues were identified. Firstly, there was a misconfiguration of one of the UWB sensors, which meant that the UWB sensors on UAV2 and UGV2 were assigned the same TDMA slot. This caused two problems; it meant that some UWB measurements on these vehicles were corrupted, and it caused the LED beacons on both vehicles to flash with identical patterns.

The corrupted UWB range measurements were easily identified by comparing the measurement to the true distance to the target. Analysis of UWB measurements from other vehicles showed that range measurements typically had a standard deviation of 0.3 m from the true value, whereas measurements from the misassigned sensors showed some errors in excess of 50 m. Based on this, we removed all measurements from the dataset where the error in the measurement was greater than 1.5 m. We acknowledge that it is not good practice to remove outliers using ground truth information, but in this case, the cause of the errors is not any fault of the sensor, but a human error in

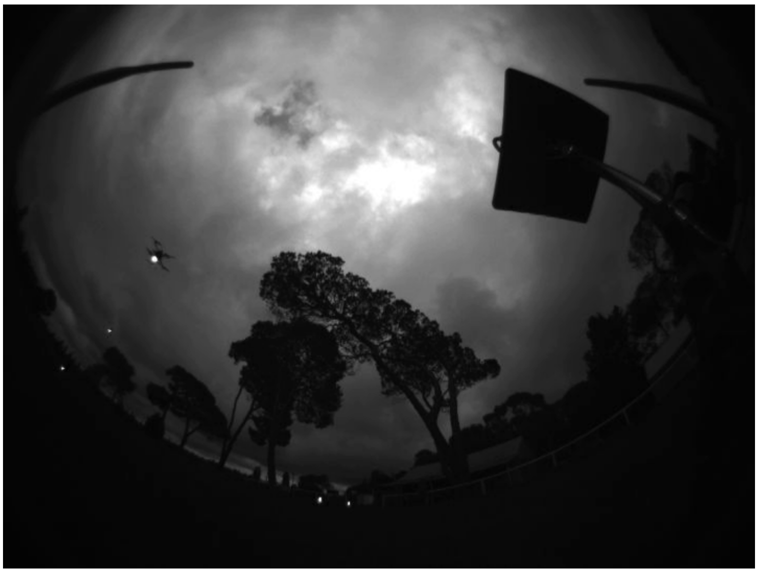


FIGURE 7.8: Example frame from the rear camera of UGV1, showing potential for false-positive detections caused by overcast sunlight filtering through trees.

misconfiguration that could have been avoided.

Recall from Section 6.6.2, that the UWB sensor was also responsible for controlling the flashing of the LED beacon. The misconfiguration of the UWB sensor also resulted in both vehicles using identical flashing patterns. This meant that any bearing measurement that was identified as UGV2 could actually be from either UGV2 or UAV2. Rather than work through the process of attempting to correctly identify the correct target, we simply discarded any bearing measurements where UGV2 is the identified target. However, on UGV2 and UAV2, we retained the measurements knowing that a vehicle cannot measure itself, thus we can be confident of the correct target. Given the abundance of other range and bearing measurements, the removal of these measurements should not significantly impact the utility of the dataset.

A separate issue identified in the data was multiple false positive detections from the bearing sensor. This was primarily an issue on the UGVs and not on the UAVs. A deeper investigation revealed that many of the false positives were caused by a conjunction of several phenomena. Due to the overcast day, the sky appeared significantly brighter in the UV spectrum than it would on a clear day. This bright light filtered through a tall tree causing a number of small point sources of light in the image, very similar to the LED beacons — an example of this is shown in Figure 7.8. While the computer vision system is designed to be robust against false positive detections, occasionally, the movement of the tree caused the false detections to form similar patterns to the LED beacons. Thus, even with the error correction in the pattern decoding, there are rare circumstances when a false positive bearing measurement is recorded.

Ordinarily, these false positives would not be a major issue and are a good test

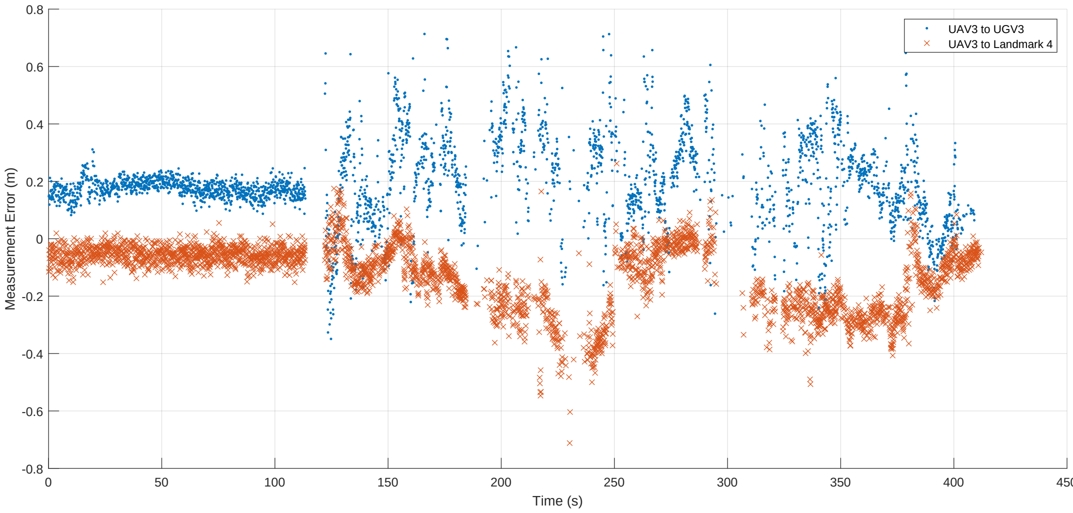


FIGURE 7.9: Indicative sample of UWB range measurement errors over time, showing only a selected subset of measurements from Mission 5 made by the UWB sensor on UAV3.

of the robustness of localisation algorithms. Thus, we opted to retain these false positives, as they are representative of the true output of the sensor. One exception to this is when the false positive was of a landmark beacon. As mentioned in Section 7.1, we were unable to conduct the experiment with LED beacons on the landmarks. Thus, any bearing measurement where the identified target is deemed a landmark beacon must be a false positive and was also eliminated.

7.3 Sensor Characterisation

Having performed the experiments and processed all the data that was recorded, we can now analyse sensor measurements and begin to characterise their properties. The two sensors that are of primary interest are the range sensor and the bearing sensor, as these represent the unique aspects of this experimental platform. We will not focus on the analysis of the off-the-shelf components including the IMU and GNSS sensors, as there is already extensive literature in this area.

7.3.1 Range Sensor

The range measurement data collected by the UWB sensors has many interesting features and the best way to understand the behaviour of the UWB is to examine the raw sensor data. In Figure 7.9, we show the error between the ground truth and the measurement data captured by UAV3 during Mission 5. Rather than attempting to present the entire set of over 40,000 measurements made by UAV3, only the range measurements of UGV3 and Landmark 4 are shown.

From the measurement error data, we can see that the range measurements do not fit the model of an i.i.d. Gaussian process. During approximately the first 110 seconds of the mission, when UAV3 is stationary, we observe a clear bias in the measurement errors, which is different for each pair of UWB nodes. As the vehicle takes off and begins to move, the sensor behaviour changes dramatically, showing a significant increase in the variance and a clear time dependence between successive measurements. The interval between successive measurements is also variable over time, and in some cases, the measurements drop out completely for several seconds. The most prominent example of this is at $t = 112$, where there is over an 8-second gap between consecutive measurements, while smaller gaps can be observed at $t = 185$ and $t = 295$.

The source of these measurement errors, biases and dropouts are due to several factors. One component of the error can be attributed to the intrinsic properties of the sensor, as discussed in the manufacturer's technical note [138]. This includes a physical variation between nodes in the antenna properties, clock drift, and received signal intensity variation. Additionally, there are extrinsic properties of the vehicle which also contribute to the error. The position of the UWB sensor and GNSS antenna relative to the vehicle origin both affect how the ground-truth distance is calculated. Thus, any error in the calibration of these parameters or the GNSS position estimate will manifest as increased measurement error.

The dropouts in the measurements are most likely due to either the occlusion of direct line-of-sight between UWB sensors or one of the sensors being in a null spot in the antenna radiation pattern. As the vehicles move through the environment, their relative pose is constantly changing, and on occasion, the vehicle itself will block the line-of-sight to some portion of the other UWB nodes in the network. Additionally, there is a significant null region in the antenna along the z-axis where the received signal is significantly attenuated. Either of these effects is sufficient to interrupt measurements. This behaviour would explain dropouts of measurements to one or two nodes at any given time, but there are circumstances, such as observed in Figure 7.9 at $t = 112$, where no measurements are received by UAV3 at all. These cases are a result of the particular TDMA protocol used to control the timing of the network. If a node loses connection to the master node (which in these experiments is Landmark 1) for an extended period, then it will lose clock synchronisation with the network and be unable to transmit in its assigned TDMA slot. The node must wait until it receives the next time synchronisation packet, which is only transmitted by the master node once per second (refer back to Section 6.6.1 for a more detailed explanation).

Despite all of these sources of error, the quality of the data is remarkably high for such a low SWAP sensor. The histograms in Figure 7.10 show that the overall performance of each of the UWB sensors is broadly similar. Based on the typical measurement distance of 50 m, the observed standard deviation of 0.25 m represents a relative error of only 0.5 %. With further calibration efforts to address the issues discussed above, measurement errors could be

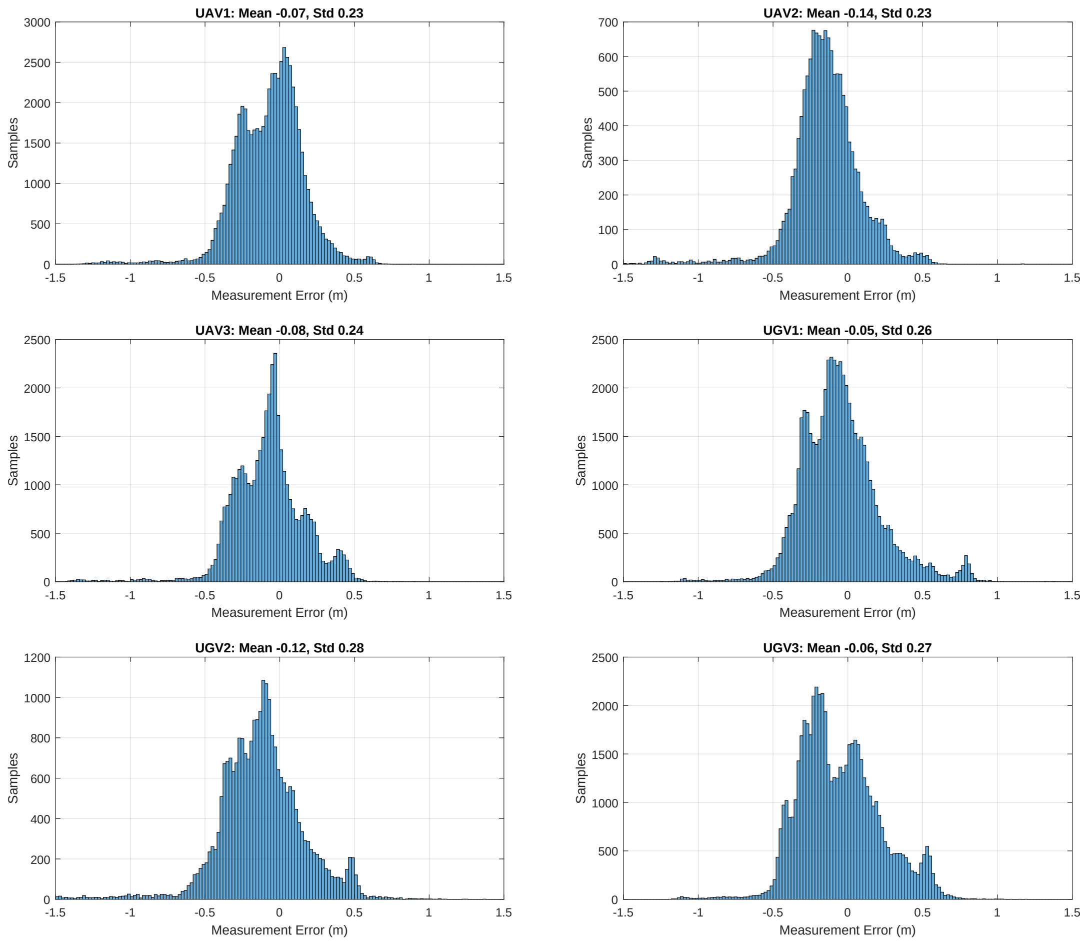


FIGURE 7.10: Histograms of UWB range measurement errors from each vehicle's respective sensor. The data represents all measurements recorded by each vehicle during Mission 5.

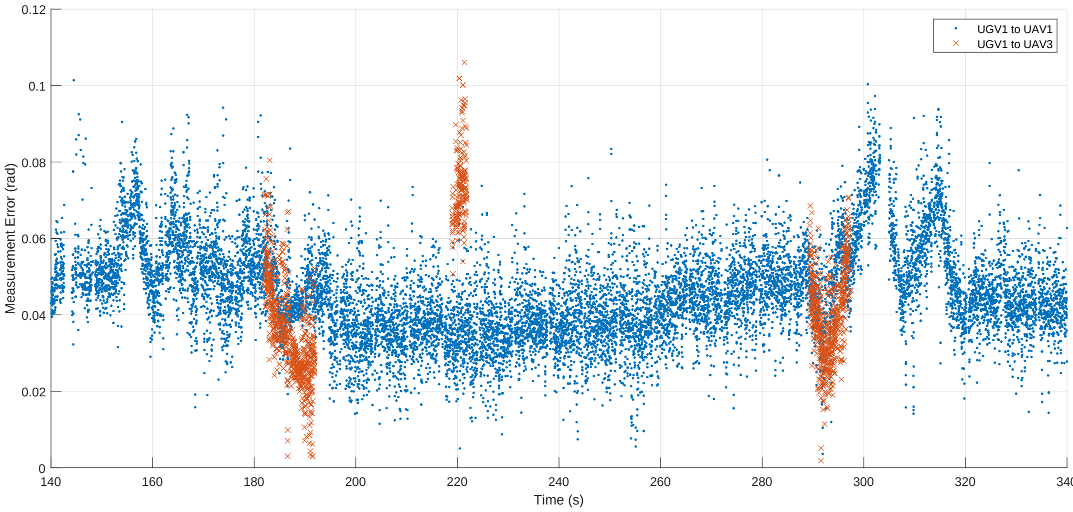


FIGURE 7.11: Indicative sample of bearing measurement errors over time, showing only a selected subset of measurement from Mission 5 made by the front camera on UGV1.

reduced significantly. Due to the position-dependent biases in the individual measurements, the distribution of errors will be affected by how long a vehicle is in a particular position, which leads to multiple peaks in the histograms.

Although we did not specifically test the maximum range of the sensors, we observed reliable performance up to about 50 m of range. However, the experiments were mostly performed with inter-vehicle distances less than 50 m, and thus it is difficult to quantify performance at longer ranges using only the data we collected. From the 5 missions, the longest measurement recorded by any sensor was 95 m.

7.3.2 Bearing Sensor

The vision-based bearing sensor was the most unproven technology element of the entire experimental setup, with few examples of similar technology in the literature. While the system concept is inspired by the work of Walter, Saska and Franchi [115], the system we implemented has very few common elements, as we used different hardware and flashing patterns, as well as different algorithms for camera calibration, LED detection, tracking, decoding, error correction, and inverse projection. While there is room for improvement, the results we obtained from the bearing sensor are very positive and show that this kind of sensor system is a viable option for inter-vehicle bearing measurements.

In Figure 7.11, we show a sample of the bearing measurement errors over time taken from the front camera on UGV2. The measurement error represents the angle between the measured bearing and the ground-truth bearing, which is calculated from the GNSS position of each vehicle and their flight controller estimates of orientation. As with the UWB sensor, this represents a small, but indicative, fraction of the hundreds of thousands of bearing measurements recorded by all the vehicles in the network. The two different measurement targets, UAV1 and UAV3, were selected to highlight some unique features of the bearing measurement data.

The biggest limitation of the bearing sensor system is that targets must be within the field of view of the lens in order to be measured. As Mission 5 was a colinear formation mission, the relative pose between the vehicles remains more or less constant, and thus UAV1 is within the field of view of the front camera on UGV1 for the entire mission. However, we can see that UAV3 is only briefly visible by the same camera at three points during the mission, leading to a lack of measurements for most of the mission duration. This unavoidable gap in measurements is partially offset by the rear-facing camera on the UGV, which provides additional coverage in the opposite direction, however, the UAVs only have one downward-facing camera, and thus cannot make any bearing measurements to vehicles above them.

The sample data also highlights the time-varying nature of the measurement error, which will be dependent on many factors, including camera calibration, detection accuracy, and ground-truth accuracy. Another factor that impacts the bearing measurements more than the range measurements is the vibration and movement of the platforms. As a UGV travels along the ground, the bumps and dips in the grass field combined with the lack of suspension introduce high levels of vibrations in the vehicle. Naturally, these vibrations will be recorded by the IMU, but if the timestamps of the IMU measurements are slightly offset from the camera frame timestamps, then there can be a significant error of several degrees in the bearing measurements.

If we examine the entire set of bearing measurement data from Mission 5, shown in Figure 7.12, we can see that each camera system has slightly different behaviour. The histograms of measurement errors highlight several important aspects, most notably the high variation in the number of measurements recorded by each sensor, and the difference in the distribution of the errors. As would be expected, vehicles at the front and rear of the formation recorded the lowest number of measurements, such as the front camera of UGV1 and the rear camera of UGV3, while the vehicles in the middle of the formation, such as UAV2 and UGV2, recorded the highest number of measurements due to their high visibility of other vehicles.

The variation in measurement errors between vehicles is most likely attributed to errors in the calibration of each camera's extrinsic and intrinsic parameters. As discussed in Section 6.5.3, calibrating these parameters is particularly challenging due to the extremely wide angle of the lens and the added complexity of the UV filter, which makes traditional methods unviable. It may be possible to improve the calibration results by designing a more precise calibration method for both the intrinsic and extrinsic parameters, which would in turn result in improved measurement accuracy.

As discussed in Section 7.2.2, outlier bearing measurements can be introduced through false detections of background objects or misidentification of valid targets. In Figure 7.12, measurements are classified as outliers if their mea-

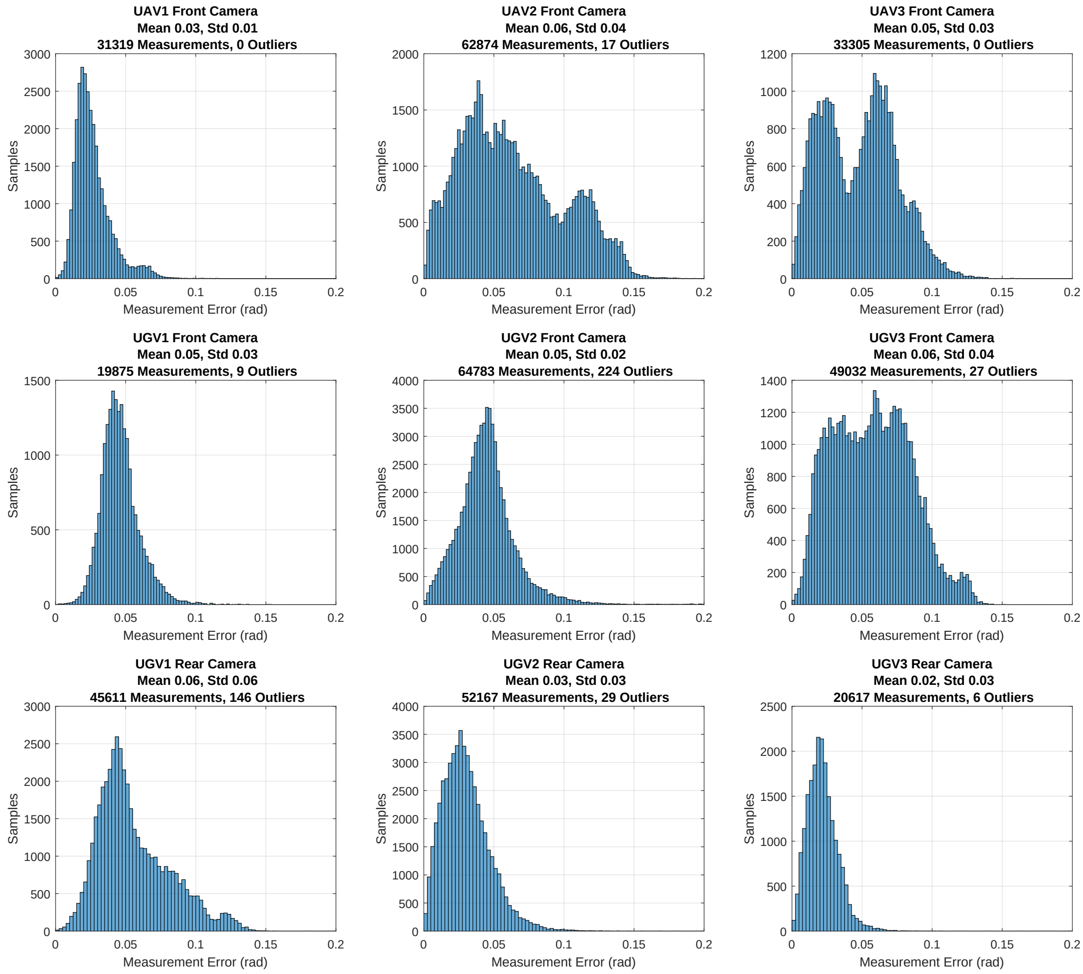


FIGURE 7.12: Histograms of bearing measurement errors from each vehicle. Data is from Mission 5.

surement error is above 0.2 rad (11°). We can see that the number of outliers is very small compared with the number of recorded measurements and that some sensors do not record any outliers at all. In the worst case, the 224 outliers recorded by the front camera of UGV2 represent only 0.3 % of the measurements made by that sensor. While it may be possible to improve the robustness of the detection and identification algorithms to reduce the number of outliers, the current rate does not present a concern.

Overall, the bearing measurement system shows a good level of performance and should be ideal for effectively evaluating various collaborative localisation algorithms. While many aspects of the system could be improved, including the camera calibration and robustness to outliers, the levels of accuracy observed are more than sufficient.

7.4 Evaluation of Collaborative Localisation Algorithms

From the analysis we have done on the dataset, we are confident that it will provide a useful tool to evaluate collaborative localisation algorithms under realistic conditions. However, there are still two small barriers that remain which must be addressed before we can run algorithm evaluations. First, due to the supply chain issues discussed earlier, we did not collect relative bearing measurements to the landmark beacons, only range measurements. Second, the filters we have designed in this thesis utilise relative position measurements and are not yet capable of processing bearing and range measurements independently.

The first issue would be ideally addressed by running additional experiments with the full set of equipment, and the second issue could be addressed by developing extensions to the localisation framework that incorporates separate range and bearing measurements as input. However, given the time constraints and scope of this thesis, we will instead address these issues by making modifications to the dataset. We must emphasise that these modifications represent one particular choice of methodology to make the dataset compatible with the algorithm we wish to test. Moreover, the dataset will still include separate range and bearing measurements available for algorithms that can process these independently.

7.4.1 Merging Bearing and Range Measurements

As we have seen in Section 7.3, the range and bearing measurements have significantly different characteristics, in terms of frequency, availability, and error distribution. Thus, merging these two measurement sources to create a single relative position measurement requires careful consideration.

The primary issue is that the range sensor and bearing sensor captured measurements independently and measurement times were not synchronised.

Add to this the fact that measurement data may be missing for extended periods for either of the sensors, *e.g.* if the target is outside the field of view of the camera. Given that the vehicles are in constant motion, measurements from different times correspond to different relative positions. Thus, if we choose to merge a bearing measurement with a range measurement from a different time point, we will introduce additional errors due to the difference in the vehicle's pose when both measurements were made.

If we examine the experimental data collected, we can quantify the amount of error that is introduced by merging measurements that were recorded at different times. We determine that the maximum ground-truth velocity of the UGV observed in any of the missions is $2\,\mathrm{m\,s^{-1}}$ and the maximum velocity of the UAV is $5\,\mathrm{m\,s^{-1}}$. Thus, in the worst case, two UAVs could have a relative velocity of $10\,\mathrm{m\,s^{-1}}$. As an example, consider a bearing measurement recorded at time t made by UAV1 to UAV2. If a corresponding range measurement was recorded at time $t + \delta t$, then we could merge these two measurements to create a relative position measurement at time t. The amount of error introduced by using a range measurement from a different time point is at most $10\delta t$ m. Thus, if we choose to merge measurements only when the difference between the measurement times is less than 0.1 s, then the maximum error that is introduced into the measurement will be 1 m. In the vast majority of cases, the actual error will be significantly less than this, as the relative velocities are often much smaller — especially if the vehicles are in a formation, where the relative velocity should be near zero.

Using this technique, we create a new set of relative position measurements for each vehicle. For each bearing measurement made to a given target, we select the range measurement to that target with the smallest difference in measurement time. If the difference between the recorded times of the two measurements is less than 0.1 s, the unit vector of the bearing is multiplied by the range to produce a 3-dimensional relative position measurement to the target. If there is no corresponding range measurement within the time interval, then no relative position measurement is created.

Due to the nature of the two sensors, a significant number of measurements are likely to be lost, because a correspondence between range and bearing measurements can not be made. This is particularly evident in Mission 1 and can be seen in the statistics in Table 7.3. For example, UAV1 makes over 139,000 bearing measurements to other vehicles during the entire mission, however, it only makes 12,900 range measurements during the same period. Thus, many of the bearing measurements do not have a corresponding range measurement within the 0.1 s threshold, and only 8,658 merged relative position measurements are created. In later missions, such as Mission 5, where the vehicles are in formation and the inter-vehicle distances are lower, the number of relative position measurements is significantly higher, as both the range and bearing measurements are more consistent.

In Figure 7.13, we show histograms of the relative position errors for each vehicle. We consider any measurement with a squared error above $5\,\mathrm{m}^2$ as an

TABLE 7.3: Number of measurements made by each vehicle's sensor during Mission 1, including post-processed relative position measurements.

		Front Camera		Rear Camera	
Vehicle	Range	Bearing	Relative Pos.	Bearing	Relative Pos.
UAV1	12,923	139,053	8,658	-	-
UAV2	5,722	47,503	4,437	-	-
UAV3	5,506	46,542	6,190	-	-
UGV1	31,867	33,868	67,850	20,538	31,781
UGV2	16,886	51,466	52,672	24,235	23,318
UGV3	33,234	65,324	38,118	32,493	19,035

outlier and, even with such tight bounds, we observe very few outliers in the dataset.

7.4.2 Generating Synthetic Landmark Measurements

To address the lack of bearing measurements to landmarks, we create additional synthetic measurements using ground truth data. Rather than create these measurements at a fixed time interval, as we did in the Simulation chapter, we create a corresponding bearing measurement for each UWB range measurement that is received. This captures the more realistic aperiodic and intermittent nature of the measurements. The measurement error is sampled from an i.i.d. normal distribution. This distribution doesn't capture the more complex features that we observed in the real bearing measurements such as the time correlations and projection distortions, and thus we intentionally selected a higher variance than we normally observe to help compensate for this.

In Figure 7.14 we show histograms of the norm-squared errors for these synthetically generated measurements. Note the significant increase in error compared with the real inter-vehicle measurement data in Figure 7.13. This ensures that the synthetic measurements provide as little information as possible to the localisation algorithm and that it will be necessary to utilise the real-world inter-vehicle measurement in order to improve localisation accuracy.

With the addition of these synthetic measurements to the dataset, we finally have the entire set of required components in order to test and evaluate different collaborative localisation algorithms. Given the volume of data and the various missions that were recorded, it is infeasible and unproductive to test every algorithm on different variations of every mission. Instead, we will focus our analysis on Mission 5 as it provides an interesting example of vehicles operating in formation and is rarely seen in the literature. The colinear geometry of the formation is likely to create a challenge for the filtering algorithms and accentuate the inter-vehicle state dependencies.

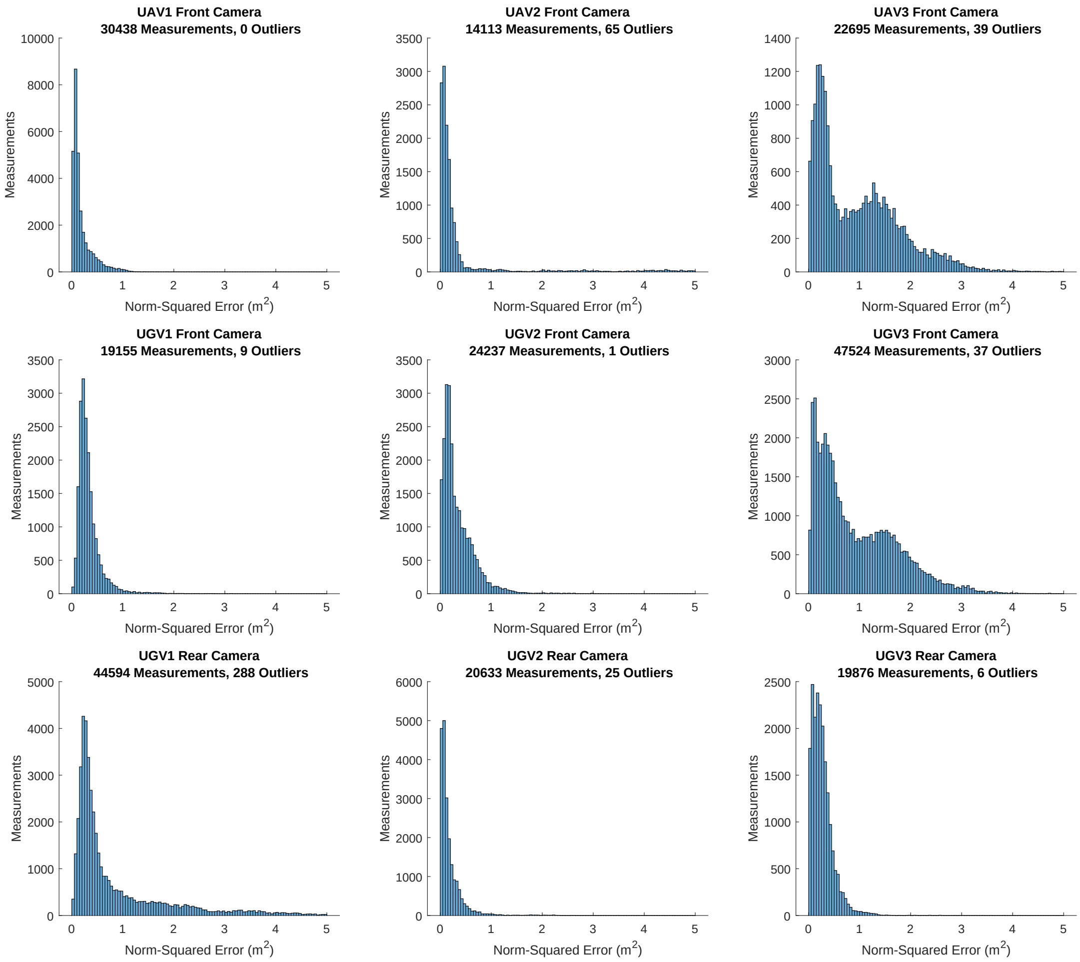


FIGURE 7.13: Histograms of norm-squared error for each relative position sensor. Data is from Mission 5.

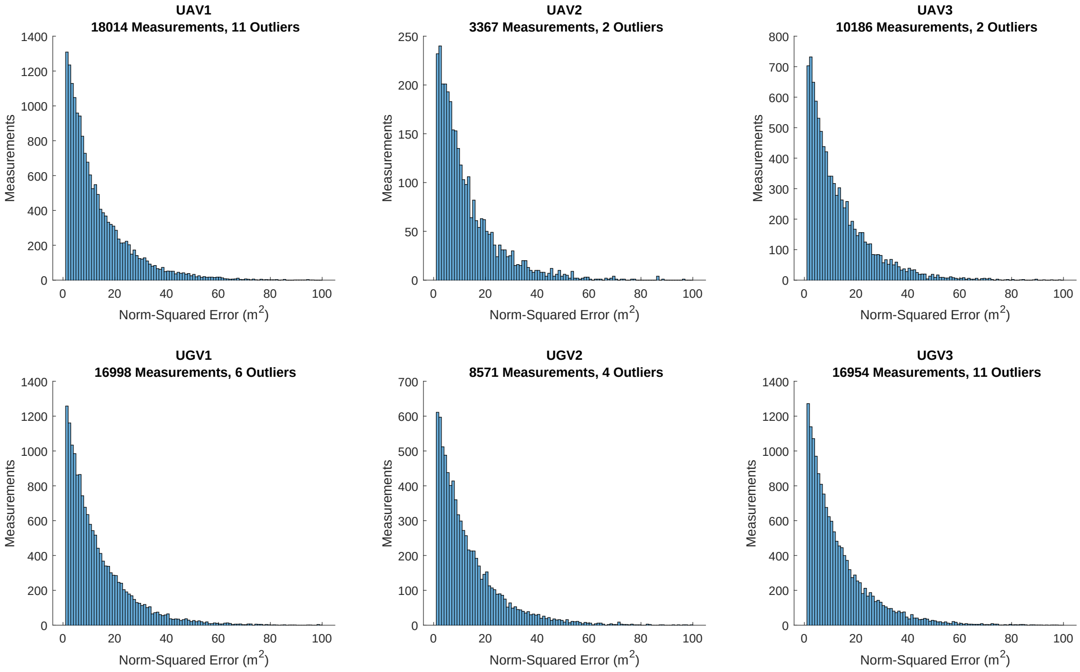


FIGURE 7.14: Histograms of norm-squared error for synthetically generated landmark measurements. Data is from Mission 5.

7.4.3 Mission 5A

Similarly to Chapter 5, we will create multiple variants of the same mission by modifying the sensor data available to each vehicle. As a baseline, we create Mission 5A, in which all vehicles have access to all the real-world inter-vehicle measurements, as well as the synthetically generated landmark measurements. This will serve as a demonstration of whether the collaborative localisation algorithms can successfully localise using the available data, and provide a baseline reference for further mission variations. Using the same notation as in Chapter 5, the visibility matrices are represented by

$$\Theta_V(t) \equiv \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 \end{bmatrix}, \quad \Theta_L(t) \equiv \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

noting that the ordering of the vehicles in the matrix is [UAV1, UAV2, UAV3, UGV1, UGV2, UGV3], and the landmarks are ordered 1 through 8.

As before, we will simulate the following five minimum energy filters and compare their relative performance;

1. **Distributed Minimum Energy Filter (D-MEF)**
   Implementation of Algorithm 10

2. **Distributed Minimum Energy Filter without Curvature (D-MEF*)**
   Implementation of Algorithm 10 with the curvature term discarded, as in Section 5.2.3
3. **Non-Collaborative Minimum Energy Filter (NC-MEF)**
   Implementation of Algorithm 1. Each vehicle operates independently using only IMU and landmark sensors.
4. **Distributed Schmidt Minimum Energy Filter (DS-MEF)**
   Implementation of Algorithm 11
5. **Distributed Approximate Schmidt Minimum Energy Filter (DAS-MEF)**
   Implementation of Algorithm 12

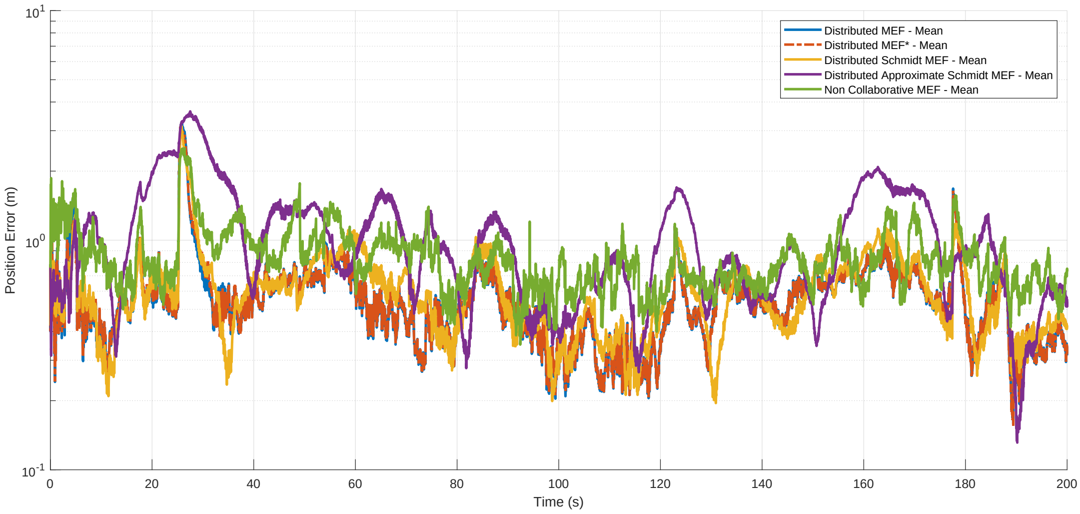
The tuning parameters of each filter remained the same as in Table 5.1 except for $\check{D}$, which is given as

$$\check{D}^i_l = \check{D}^i_j = \sqrt{1.5}\boldsymbol{I}.$$

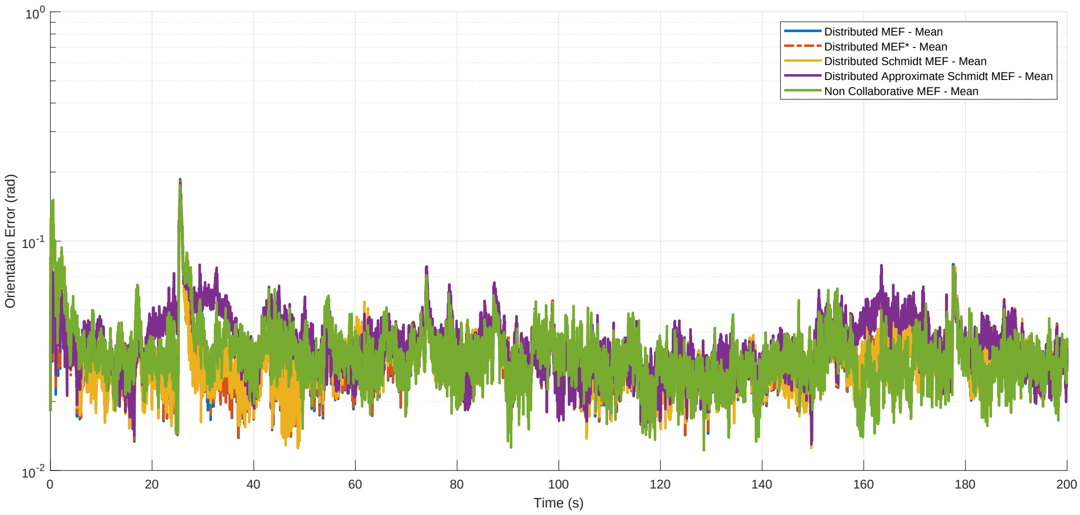
The mean position and orientation estimate errors for each of the filters are shown in Figure 7.15. Promisingly, all the filters remain stable and provide accurate position and orientation estimates for the duration of the mission. The significance of this result alone cannot be overstated. It validates the entire experimental process from the hardware selection, experimental procedure, data collection, post-processing, and sensor calibration. It also confirms that the reference frames for each sensor are correctly configured and the ground-truth data is compatible with the sensor measurements. There are countless ways in which the misconfiguration of a parameter, or a bug in the data processing pipeline could have entirely invalidated the entire dataset, which would lead to filter divergence instead of the stable results observed here (many of which the author discovered).

The filter results also indicate that the frequency and accuracy of the sensor measurements are sufficient to accurately localise, even with the variability of the sensor measurements identified in Section 7.3. In Table 7.4, the overall mean estimation errors are given. The errors observed here are larger than those observed in the simulation data from Section 5.2.4, which were on the order of 0.12 m and 0.02 rad (1.15°) for position and orientation error respectively, but this is to be expected due to the additional sources of error in the experimental data.

The relative performance between the different collaborative filters, most clearly seen in Figure 7.16, is generally consistent with the results observed in the simulation. The Distributed MEF shows the best results, and again there is a negligible difference when the curvature term is discarded, as indicated by the overlap between the Distributed MEF and Distributed MEF*. This once again highlights the improvements in localisation performance that can be achieved through collaborative localisation.

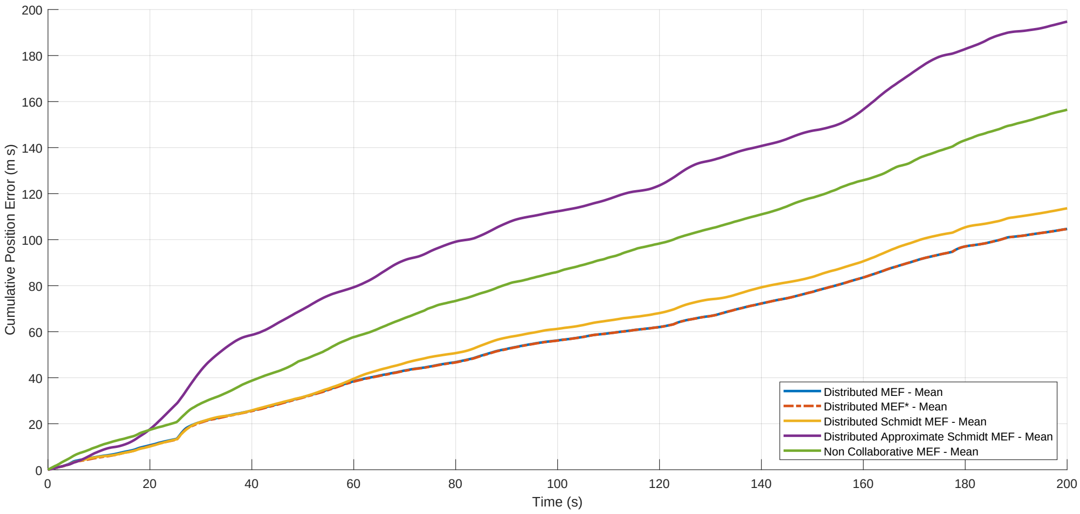


(A) Position Error

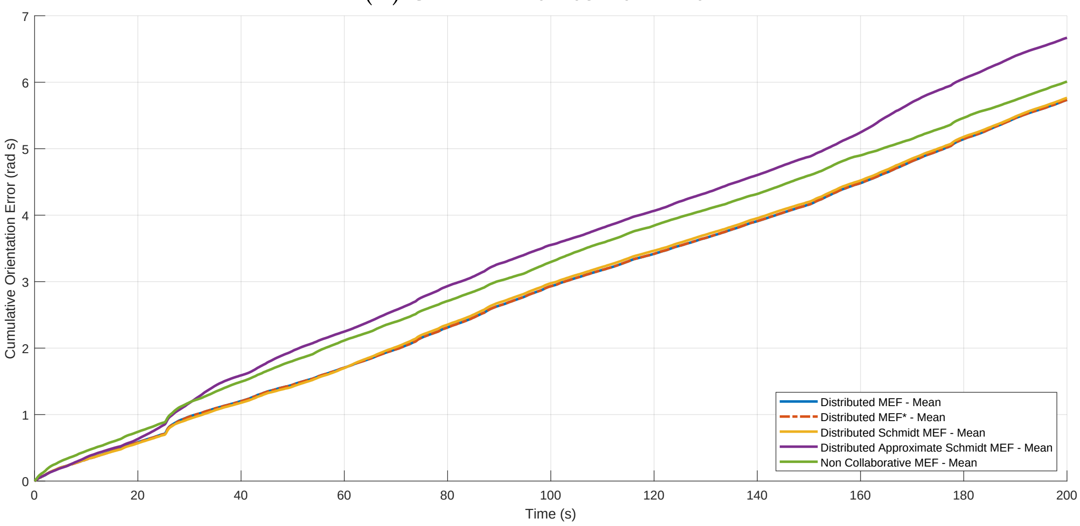


(B) Orientation Error

FIGURE 7.15: Estimate errors for different filters on Mission 5A.



(A) Cumulative Position Error



(B) Cumulative Orientation Error

FIGURE 7.16: Cumulative estimate errors for different filters on Mission 5A.

TABLE 7.4: Mean estimation error of Mission 5A

	Position m	Orientation rad
Distributed MEF	**0.566**	**0.031**
Distributed MEF*	**0.566**	**0.031**
Distributed Schmidt MEF	0.615	**0.031**
Distributed Approximate Schmidt MEF	1.054	0.036
Non-Collaborative MEF	0.847	0.033

The Distributed Schmidt MEF shows almost no increase in orientation error compared with the Distributed MEF, but a slight increase in position error. We suggest that this is due to the higher relative noise levels of the accelerometer compared with the gyroscope, meaning that the position estimate will be more reliant on exteroceptive measurements than the orientation estimate. In this case, the DS-MEF shows how a significant reduction in communication overhead can be achieved without significantly reducing localisation accuracy.

Interestingly, the results for the Distributed Approximate Schmidt MEF (DAS-MEF) are significantly worse than even the Non-Collaborative MEF. This indicates that the approximations made in the DAS-MEF construction are doing more harm than good. We observe that the DAS-MEF provides better performance than the Non-Collaborative MEF for approximately the first 20 seconds and thereafter performance is worse. Similar behaviour can be observed in Simulation 3 (Section 5.4), but it is not present in other simulations. This suggests that there are some network topologies or measurement sequences in which the approximations made by the DAS-MEF are more accurate than others.

The final observation is that the landmark data alone is sufficient to localise each vehicle, as evidenced by the stability of the Non-Collaborative MEF. In the next two sections, we will modify the mission data and further reduce the sensor measurements to demonstrate how collaborative localisation is also effective in degraded sensor environments.

7.4.4 Mission 5B

Having demonstrated in Mission 5A that the experimental data is of sufficient quality to support accurate localisation, we now move to a more challenging scenario that demonstrates the capabilities of collaborative localisation in degraded sensor conditions. Using Mission 5 as the baseline, we will remove a significant amount of the landmark measurements from each vehicle, similar to the conditions in Simulation 2 (Section 5.3). For the first 60 seconds, we include all the available landmark measurements in the dataset, which gives a landmark visibility matrix of

$$\Theta_{\boldsymbol{L}}(t)_{t\in[0,60)} \equiv \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}.$$

Then to introduce a period of significant sensor degradation, we reduce the visibility matrix to

$$\Theta_{\boldsymbol{L}}(t)_{t\in[60,160)} \equiv \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

indicating that each vehicle can only make measurements of one of the landmarks. Such a situation might occur in the real world due to environmental conditions that obstruct the line of sight to the landmarks, such as steep terrain, dense tree coverage, or heavy fog. After 100 seconds of degraded sensor data, we return to full landmark visibility, as indicated by the change in visibility matrix to

$$\Theta_{\boldsymbol{L}}(t)_{t\in[160,200)} \equiv \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix},$$

until the end of the mission at $t = 200$.

For the duration of the mission, we do not impose any constraints on the inter-vehicle measurements, and thus the inter-vehicle visibility matrix is represented as

$$\Theta_{\boldsymbol{V}}(t) \equiv \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 \end{bmatrix}.$$

The position and orientation errors for each of the tested algorithms are shown in Figure 7.17, with the cumulative errors shown in Figure 7.18 and summary data shown in Table 7.5. Consistent with Simulation 2, these results again

TABLE 7.5: Mean estimation error of Mission 5B

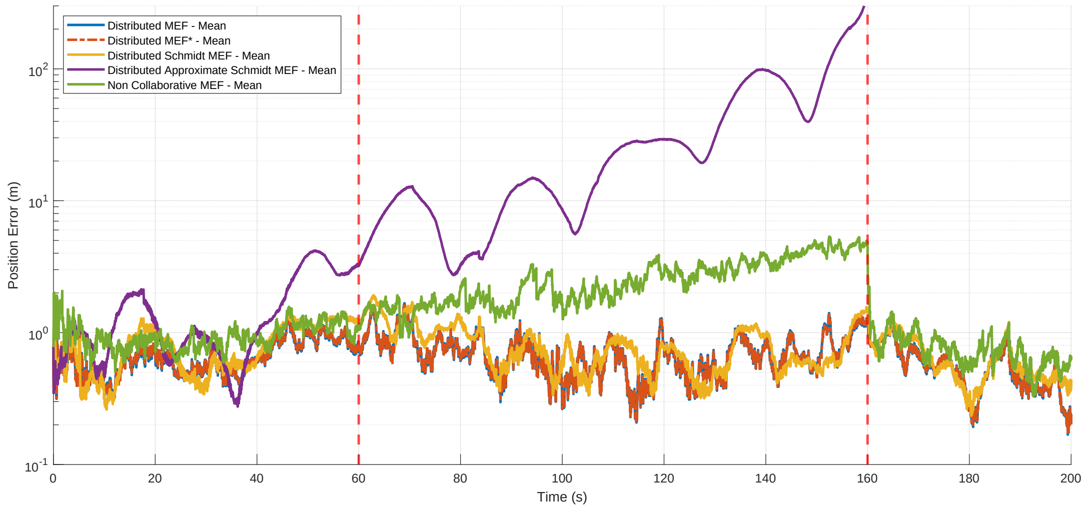
	Position m	Orientation rad
Distributed MEF	**0.667**	**0.039**
Distributed MEF*	**0.667**	**0.039**
Distributed Schmidt MEF	0.773	0.040
Distributed Approximate Schmidt MEF	N/A	0.739
Non-Collaborative MEF	1.734	0.060

demonstrate the effectiveness of collaborative localisation in maintaining accurate state estimates even when individual robots do not have sufficient information.

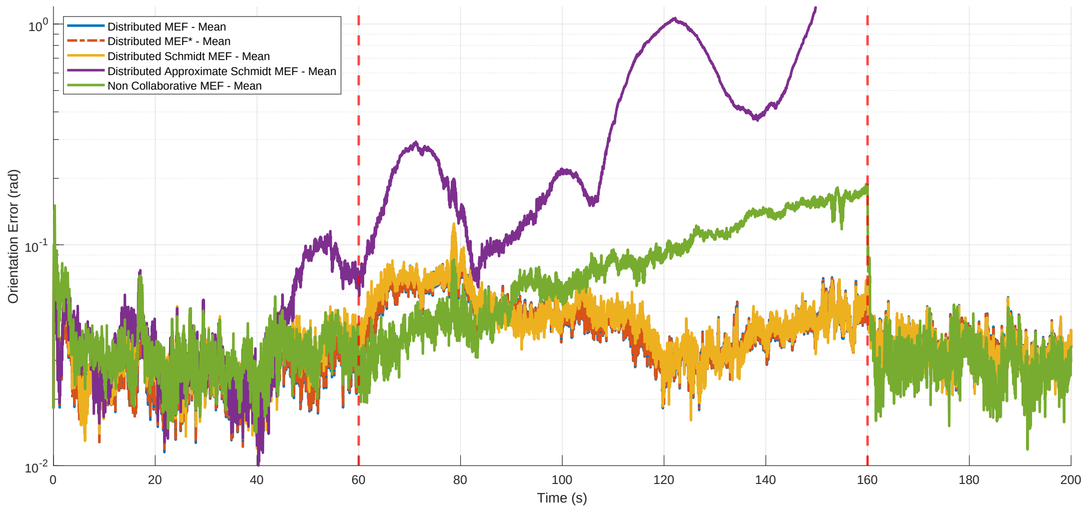
Looking at the performance of the Non-Collaborative MEF, it is clear that the information obtained from one landmark is not sufficient to maintain an accurate estimate. The position and orientation estimates both begin to increase, with no indication of abating until the full set of landmark measurements are restored at $t = 160$.

On the other hand, there is almost no discernible difference between position estimate errors of the Distributed MEF during the period of degraded landmark information compared with the periods of full landmark measurements. Similarly, the D-MEF* and DS-MEF appear unaffected by the reduction in measurements. The orientation estimates do appear to increase slightly, but not to the same level as the non-collaborative filter.

The DAS-MEF again struggles to maintain an accurate estimate, even during the initial period of full landmark measurements. As the landmark measurements are reduced, the error in the DAS-MEF estimate increases even further, to the point where it is significantly worse than the non-collaborative estimate. Even when landmark measurements are restored, and the non-collaborative filter re-converges to the baseline level of performance, the DAS-MEF continues to diverge and completely destabilises.

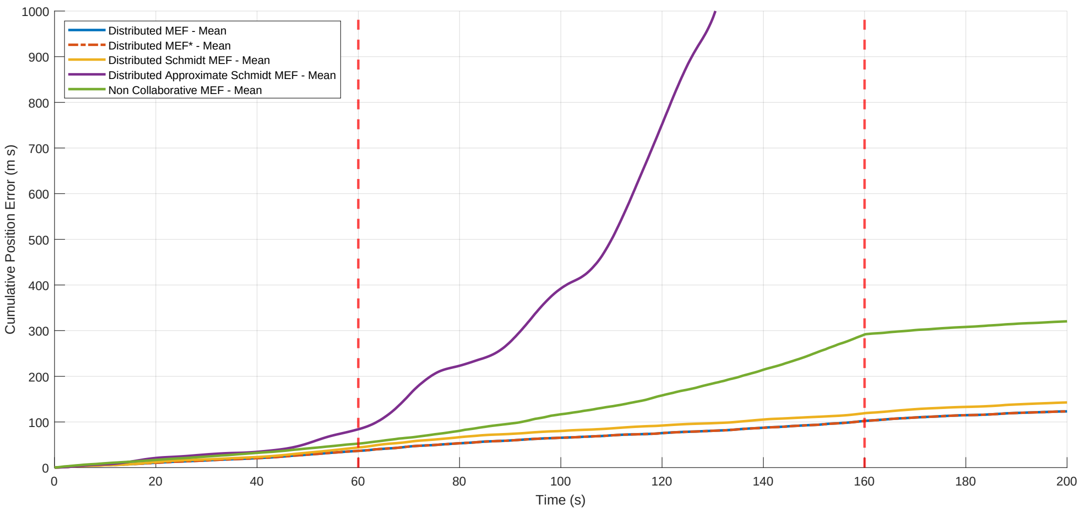


(A) Position Error

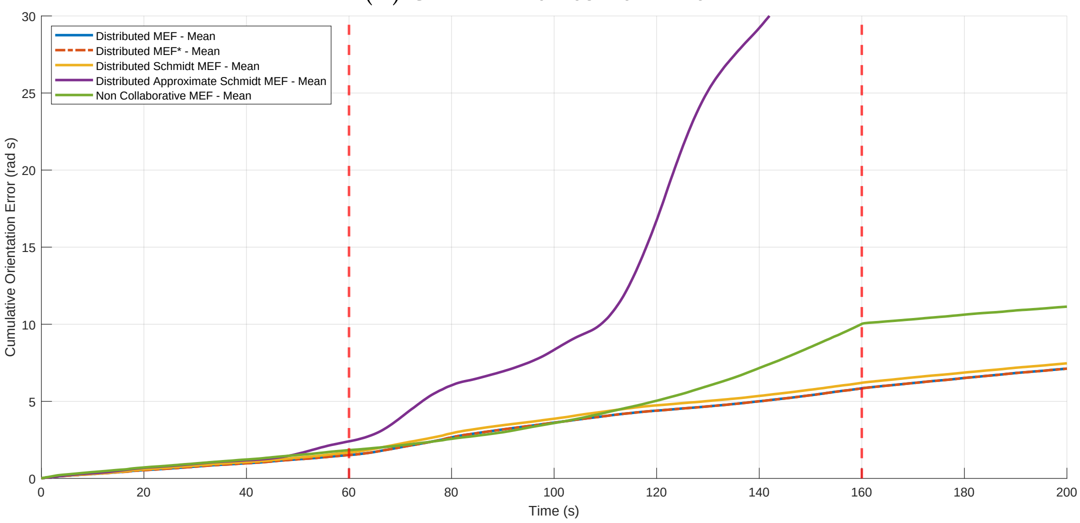


(B) Orientation Error

FIGURE 7.17: Estimate errors for different filters on Mission 5B.



(A) Cumulative Position Error



(B) Cumulative Orientation Error

FIGURE 7.18: Cumulative estimate errors for different filters on Mission 5B.

7.4.5 Mission 5C

In the final mission, we demonstrate how collaborative localisation can be used to support vehicles that do not have access to any absolution position information. Similarly to Mission 5B, for the first 60 seconds of the mission, all vehicles receive all the available landmark measurements, with the landmark visibility matrix represented by

$$\Theta_{\boldsymbol{L}}(t)_{t\in[0,60)} \equiv \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}.$$

After 60 seconds of elapsed time, we restrict access to landmark measurements to only one robot, namely UAV1. This is represented by the visibility matrix

$$\Theta_{\boldsymbol{L}}(t)_{t\in[60,160)} \equiv \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

Here we are demonstrating a situation similar to that in Simulation 3, where the remaining five robots in the network must rely on relative measurements and the information shared by UAV1. After 100 seconds of restricted measurements, the landmark measurements are re-enabled for all vehicles, represented by

$$\Theta_{\boldsymbol{L}}(t)_{t\in[160,200)} \equiv \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}.$$

As in the previous mission, we do not restrict the inter-vehicle measurements, and thus the inter-vehicle visibility matrix remains as

$$\Theta_{\boldsymbol{V}}(t) \equiv \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 \end{bmatrix}$$

for the duration of the mission.

TABLE 7.6: Mean estimation error of Mission 5C

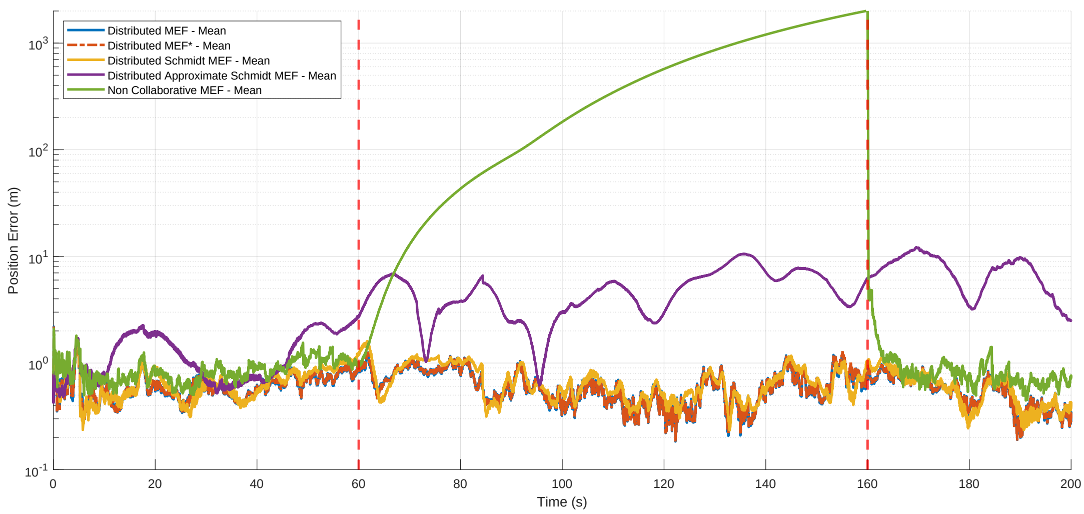
	Position m	Orientation rad
Distributed MEF	**0.607**	0.044
Distributed MEF*	0.609	0.044
Distributed Schmidt MEF	0.663	**0.043**
Distributed Approximate Schmidt MEF	4.331	0.086
Non-Collaborative MEF	294.870	0.068

The position and orientation errors of the 5 localisation algorithms tested for this mission are shown in Figure 7.19, with the cumulative errors shown in Figure 7.20 and summary data shown in Table 7.6. The results paint a very similar picture to those shown in the previous section.

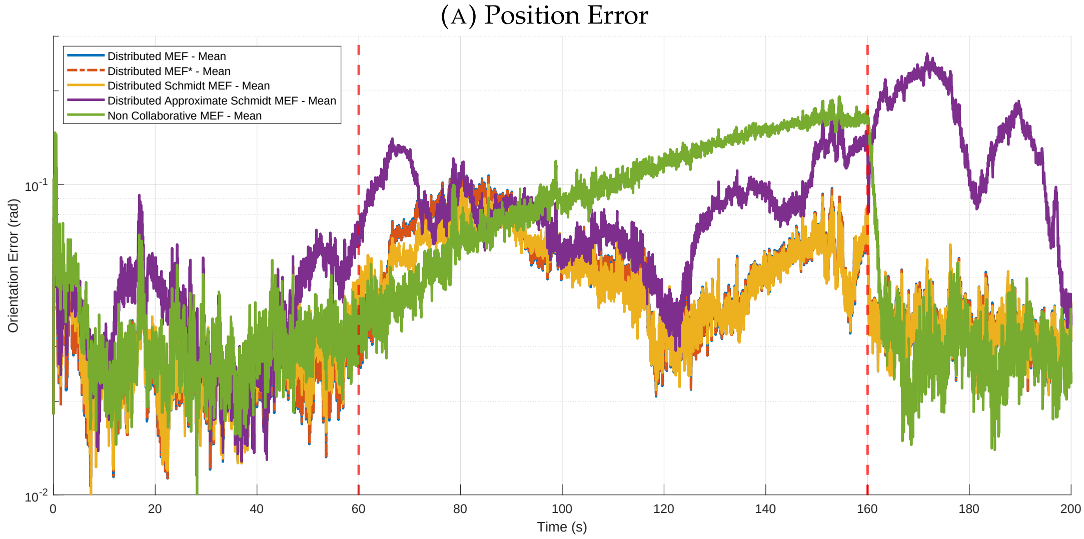
Again, it is very clear that there is not a sufficient amount of landmark measurements for the non-collaborative MEF to accurately localise during the period from $t = 60$ to $t = 160$. Given that only one of the robots receives any landmark measurements during this period, the errors in the estimates of the remaining five robots will continue to increase unbounded, driven by the errors in the IMU measurements. The difference between the position and orientation estimates is remarkable — the mean position error rapidly increases to over 2000 m, whereas the mean orientation error increases at a much slower rate, reaching a peak of only 0.19 rad (10.9°). This is a result of both the higher noise levels of the accelerometer compared to the gyroscope, and the compounding error of the double integration between acceleration and position. Despite such a large position error, the Non-Collaborative MEF rapidly converges to an error of less than 1 m once landmark measurements are restored at $t = 160$. This highlights the strength of using a geometric filter, such as the minimum energy filter.

The performance of the D-MEF, D-MEF* and the DS-MEF are all very similar, with the DS-MEF slightly outperforming the orientation estimate of the D-MEF. Just like the previous mission, there is very little difference in position error between the period of full landmark measurements and reduced measurements for these three filters. It is important to remember that, even though the bearing component of the landmark measurements was synthetically generated, only UAV1 receives landmark measurements during the period $t = 60$ to $t = 160$. During this time, the remaining five robots are all relying on real-world bearing and range measurements, with no synthetically generated components. Thus, the fact that the collaborative filters maintain a similar level of performance throughout the whole mission demonstrates the high quality of measurement data that was captured during the experiments.

The DAS-MEF again shows a higher level of error than all other collaborative filters, and, in some periods, has a higher error than the non-collaborative filter. However, compared with the last mission, the filter does not appear to diverge, instead fluctuating between a position error of 1 m to 10 m. Given

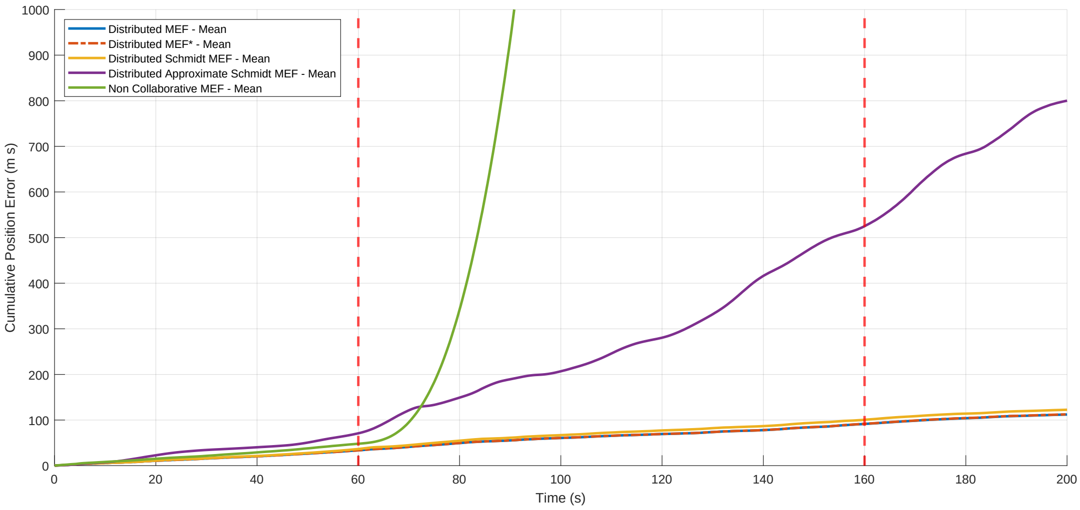


(A) Position Error

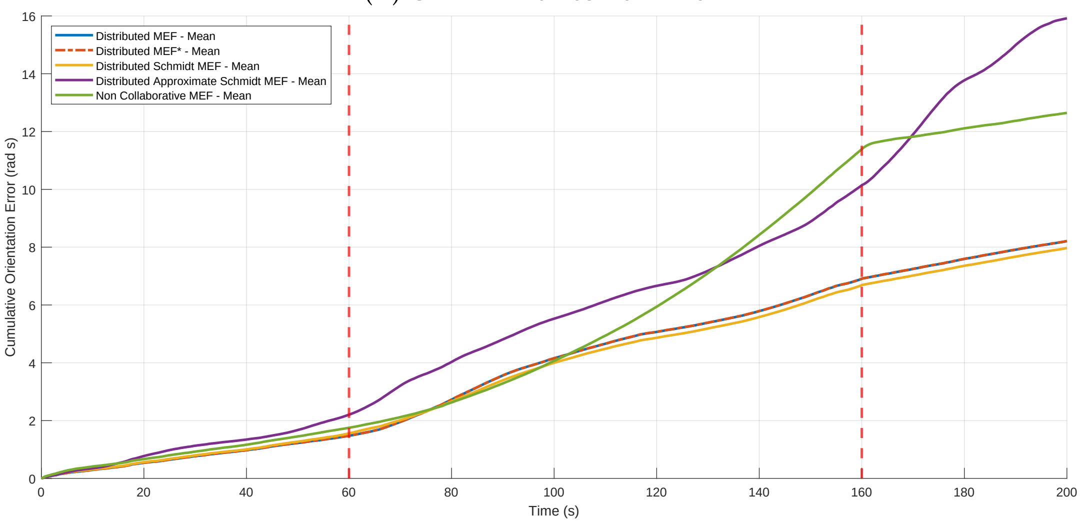


(B) Orientation Error

FIGURE 7.19: Estimate errors for different filters on Mission 5C.



(A) Cumulative Position Error



(B) Cumulative Orientation Error

FIGURE 7.20: Cumulative estimate errors for different filters on Mission 5C.

the variability in the performance of this filter in both the simulation and real-world testing, a more thorough investigation into this filter should be performed to investigate if there are particular topologies of networks or other factors that affect filter performance. Without a more detailed understanding, this filter appears to be too unpredictable in its performance to provide any utility over the traditional non-collaborative filter.

7.5 Summary

The results presented in this chapter demonstrate the validity of the experimental data that was collected for this thesis. We have successfully created a new state-of-the-art multi-robot dataset that can be used to evaluate collaborative localisation algorithms. The dataset contains 5 different missions, each with different trajectories and varying levels of vehicle interaction, which captures a cross-section of typical multi-robot behaviours. The analysis of the new dataset reveals a number of interesting and unique characteristics of the bespoke bearing and range measurement system. This is exactly the kind of data that cannot be replicated easily in simulation and is the primary reason that creating such a dataset is necessary.

By running the different localisation algorithms we have developed on this new dataset, we have demonstrated that the dataset contains all the necessary components and is configured correctly and that the quality of the measurement data is sufficient. But, perhaps more importantly, we have demonstrated that our proposed collaborative localisation minimum energy filters are capable of performing accurate localisation based on real-world experimental data.

Given the limitations of existing collaborative localisation datasets, this new dataset should become a valuable resource for the research community. While we have performed some post-processing steps, by merging the range and bearing measurements together, and adding in synthetic landmark bearing measurements, many other use cases for this dataset do not require these modifications. Future researchers in the field may use this dataset to test other variations of collaborative localisation algorithms, such as range-only localisation, relative localisation, or GNSS-assisted localisation. In Missions 5B and 5C, we have demonstrated two ways in which the dataset can be modified to test different challenging localisation scenarios. It is now up to the community to put the data to use in their own creative ways.

Chapter 8

Conclusion

This thesis has drawn on nearly 5 years of research and experimentation on collaborative localisation, culminating in the demonstration of several proposed minimum energy collaborative localisation algorithms on real-world experimental data.

The novel minimum-energy filter developed in Chapter 3 demonstrated how the minimum-energy approach can be successfully applied to velocity-aided inertial localisation. This filter, as described in Algorithm 1, is a novel application of the minimum energy approach and is readily implementable on autonomous vehicles for inertial navigation. This filter design also provided the foundation necessary for the extension to a minimum-energy collaborative localisation algorithm.

In Chapter 4, the in-depth analysis of existing approaches to EKF-based collaborative localisation elicited the fundamental mathematical structure that allows for the distribution of centralised algorithms, shown succinctly in Algorithm 6. In turn, this guided the process of transforming the centralised multi-vehicle collaborative localisation filter, Algorithm 9, into a distributed filter that minimises network communication between vehicles, Algorithm 10 — the first known case of applying such techniques to a filter that is not a derivative of the EKF. We further showed how other EKF-based distributed collaborative filters have parallels in the minimum energy filter, including the distributed Schmidt filter, shown in Algorithm 11 and the distributed approximate Schmidt filter, shown in Algorithm 12. These filters have the advantage of reducing the amount of communication between vehicles but come at the cost of reduced performance.

This trade-off in performance was further explored in Chapter 5 which demonstrated the different filter algorithms in simulation. From the simulation results, we observed the ability of the minimum energy filters to accurately localise even in challenging conditions. The improvement that collaborative localisation provided over the non-collaborative approach is clearly evident in multiple scenarios, such as those in Sections 5.3 and 5.4 where landmark sensor measurements are restricted.

Recognising the limitations of existing datasets and simulation tools, in Chapter 6 we designed and constructed a bespoke fleet of both aerial and ground-based vehicles, described in Section 6.3, that were specifically designed to collect sensor data required for collaborative localisation. Given that such a dataset has not previously been constructed, we addressed several technical engineering challenges with novel solutions. A key component of the system and a novel contribution of this thesis was the bearing measurement system, described in Section 6.5, that captured relative bearing information between vehicles as well as from the vehicle to fixed landmarks. In addition to the bearing sensor, the Ultra-Wideband (UWB) sensor system produced relative distance measurements, detailed in Section 6.6, which required a custom firmware implementation to enable the collection of inter-vehicle distance measurements.

The final experimental results in Chapter 7 draw together all the prior contributions of this thesis. We show how the minimum energy filter, in both the collaborative and non-collaborative form, can accurately localise vehicles using real-world sensor data. Further to this, the scenarios with degraded sensor information highlight the effectiveness of the collaborative algorithms to maintain accurate pose estimates while the non-collaborative algorithms diverge. The experimental data itself is another major contribution of this thesis, allowing other researchers to benchmark and compare different collaborative localisation algorithms on a modern, heterogeneous, multi-vehicle dataset.

8.1 Future Work

While this research has led to a number of insights into collaborative localisation, it also raises new questions and opens new avenues for future research.

An immediate continuation of this work would explore the feasibility of deriving separate update equations for independent bearing and range measurements. This would enable bearing-only or range-only collaborative localisation, as well as obviate the need for bearing and range measurements to be synthesised into relative position measurements (as in Section 7.4.1). An improvement to the discretisation process described in Section 3.4 may yield improved numerical stability of the filter and potentially increased accuracy.

To improve the quality of the experimental data, several improvements could be made to both the hardware design and the experimental procedure. The integration of event cameras [4] onto each platform could improve the detection and tracking component of the bearing measurement system and would help to reduce the number of outliers in the data. The UWB system would benefit significantly from a redesigned TDMA protocol that does not rely on synchronisation with the master clock. Subsequent experiments should be conducted incorporating all the lessons learned from the first experiment, as well as utilising the new hardware to record vehicle to landmark bearing measurements and implementing a formation control system for the vehicles.

A demonstration of the proposed algorithms running on-board vehicles in real-time would be a significant milestone towards a feasible real-world filter implementation. While this was outside the scope of this research, the tight synchronisation requirements imposed by the current algorithm are likely to be affected by latency, packet loss, jitter, and bandwidth constraints that are present in all physical wireless networks. An initial implementation in a controlled lab environment would reveal any fundamental limitations and allow for the filter to be evaluated under varying levels of network quality.

Looking further towards the future applications of collaborative localisation, further effort should be dedicated to continuing the work of Section 4.5, in which we explored the concept of dynamically restricting communication to a subset of vehicles in the network. While the preliminary results of this work were not promising, a more in-depth study may yield crucial insights. The foundation of this problem is that it is impractical to require all-to-all communication between vehicles every time a measurement is made (as in the case of the distributed EKF [79]), but one-to-one communication (such as the Approximate Schmidt-Kalman filter [98]) does not make the best use of available resources. A middle ground, where information can be shared with a select few vehicles, based somehow on a measure of informativeness, would provide a significant improvement in localisation performance for a minimal increase in communication. The open problems are both in determining which vehicles to share information with, and constructing a method to fuse this information without requiring additional communication or making invalid approximations.

Appendix A

Notation, Definitions and Conventions

This appendix provides further explanation for some of the symbols, operators and conventions used in this thesis.

A.1 Matrix Operators and Identities

A.1.1 Symmetric Projection

The symmetric projection operator, $\mathbb{P}_s$, transforms an arbitrary square matrix into a symmetric matrix, defined as

$$\begin{aligned} \mathbb{P}_s &: \mathbb{R}^{n\times n} \to \mathbb{R}^{n\times n}, \\ \mathbb{P}_s(A) &:= \frac{1}{2}(A + A^\top). \end{aligned}$$

For two vectors $x, y \in \mathbb{R}^n$ and a matrix $A \in \mathbb{R}^{n\times n}$, the following identity holds

$$\frac{1}{2}(x^\top Ay + y^\top Ax) = x^\top \mathbb{P}_s(A)y \tag{A.1}$$

A.1.2 Block Diagonal Constructor

The block-diagonal matrix constructor, blkdiag, is shorthand notation for

$$\mathrm{blkdiag}(A, B, C, \dots) := \begin{bmatrix} A & 0 & 0 & \\ 0 & B & 0 & \dots \\ 0 & 0 & C & \\ & \vdots & & \ddots \end{bmatrix}$$

A.1.3 Inverse Identities

The inverse of a block matrix can be represented as

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix}^{-1} = \begin{bmatrix} A^{-1} + A^{-1}B(D - CA^{-1}B)^{-1}CA^{-1} & -A^{-1}B(D - CA^{-1}B)^{-1} \\ -(D - CA^{-1}B)^{-1}CA^{-1} & (D - CA^{-1}B)^{-1} \end{bmatrix} \tag{A.2}$$

where A and D are square, B and C have appropriate dimensions, and both A and $(D - CA^{-1}B)$ are invertible.

We also have the Woodbury matrix identity which states that

$$(A + UCV)^{-1} = A^{-1} - A^{-1}U(C^{-1} + VA^{-1}U)^{-1}VA^{-1} \tag{A.3}$$

where A and C are square, U and V have appropriate dimensions, and both A and $(C^{-1} + VA^{-1}U)$ are invertible.

A.1.4 Weighted Norm

The weighted 2-norm of a vector $x \in \mathbb{R}^n$ with weight matrix $A \in \mathbb{R}^{n \times n}$ is defined as

$$\|x\|_A := \sqrt{x^\top A x}. \tag{A.4}$$

Thus, the squared weighted norm becomes

$$\|x\|_A^2 = x^\top A x$$

A.2 Differential Geometry

A.2.1 Exponential Functor

Given a linear map between two vector spaces, $f : U \to V$, and a third vector space W, the exponential functor $(.)^W$ lifts the map f to the linear map $f^W : \mathfrak{L}(W, U) \to \mathfrak{L}(W, V)$ defined by $f^W(g) = f \circ g$.

As an example, consider f applied a linear map, $g : W \to U$, applied to an arbitrary element $w \in W$. The exponential functor gives the relation

$$f \circ g \circ w = \left(f^W \circ g\right) \circ w \tag{A.5}$$

A.2.2 Connection Function and Torsion Tensor

The left-invariant affine connection on a Lie group G is characterised by the connection function $\Lambda : \mathfrak{g} \times \mathfrak{g} \to \mathfrak{g}$. Three connection functions are of particular interest, namely the $(-)$-connection, the $(+)$-connection, and the (0)-connection [139]. The three connection functions are defined as

$$\Lambda_-(\Gamma, \Psi) := 0,$$

$$\Lambda_+(\Gamma, \Psi) := [\Gamma, \Psi] = \mathrm{ad}_\Gamma(\Psi),$$
$$\Lambda_0(\Gamma, \Psi) := \frac{1}{2}[\Gamma, \Psi] = \frac{1}{2}\mathrm{ad}_\Gamma(\Psi).$$

Each connection function induces a corresponding torsion tensor, $T : \mathfrak{g} \times \mathfrak{g} \to \mathfrak{g}$, which are given by

$$T_-(\Gamma, \Psi) := -[\Gamma, \Psi],$$
$$T_+(\Gamma, \Psi) := [\Gamma, \Psi],$$
$$T_0(\Gamma, \Psi) := 0.$$

A.3 Matrix Lie Groups used in Robot Localisation

There are a number of matrix Lie groups which are used in the field of robotics. Among these, are the special orthogonal group and the special euclidean group which are used to represent orientations and poses of rigid bodies.

The Special Orthogonal group, $\mathrm{SO}(n)$, is a family of groups defined by

$$\mathrm{SO}(n) := \left\{ R \in \mathbb{R}^{n \times n} \mid R^\top R = \boldsymbol{I}_n, \ \det\{R\} = 1 \right\},$$

with the corresponding Lie algebra given by

$$\mathfrak{so}(n) := \left\{ \Gamma \in \mathbb{R}^{n \times n} \mid \Gamma^\top = -\Gamma \right\}.$$

In particular, we use $\mathrm{SO}(2)$ to represent orientations of rigid bodies and coordinate frames in 2-dimensional space, and $\mathrm{SO}(3)$ is used in 3-dimensional space.

The Special Euclidean group, $\mathrm{SE}(n)$, is defined by

$$\mathrm{SE}(n) := \left\{ \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix} \in \mathbb{R}^{n+1 \times n+1} \mid R \in \mathrm{SO}(n), \ p \in \mathbb{R}^n \right\},$$

with the associated Lie algebra given by

$$\mathfrak{se}(n) := \left\{ \begin{bmatrix} \Gamma & v \\ 0 & 0 \end{bmatrix} \in \mathbb{R}^{n+1 \times n+1} \mid \Gamma \in \mathfrak{so}(n), \ v \in \mathbb{R}^n \right\}.$$

In robotics, the Special Euclidean group is used to represent rigid body poses, reference frames, and coordinate transformations, with $\mathrm{SE}(2)$ used in 2-dimensional space and $\mathrm{SE}(3)$ used in 3-dimensional space.

An extension to the Special Euclidean group, denoted as $\mathrm{SE}_2(3)$, can be used to represent both the pose and linear velocity, v, of a rigid body in 3-dimensional space (refer to [140, IV.B] and [70, A.2] for further details). The extended

Special Euclidean group is defined as

$$\mathrm{SE}_2(3) := \left\{ \begin{bmatrix} R & x & v \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \in \mathbb{R}^{5\times5} \mid R \in \mathrm{SO}(3),\ x, v \in \mathbb{R}^3 \right\}$$

with the Lie algebra given by

$$\mathfrak{se}_2(3) = \left\{ \begin{bmatrix} \Gamma_R & \Gamma_x & \Gamma_v \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \in \mathbb{R}^{5\times5} \mid \Gamma_R \in \mathfrak{so}(3),\ \Gamma_x, \Gamma_v \in \mathbb{R}^3 \right\}.$$

A.3.1 Operators on Matrix Groups

The wedge and vee operators transform between a group's Lie algebra and its vector representation. These operators are defined as

$$(.)^\wedge : \mathbb{R}^n \to \mathfrak{g},$$
$$(.)^\vee : \mathfrak{g} \to \mathbb{R}^n,$$

such that

$$(\Gamma^\vee)^\wedge := \Gamma.$$

On $\mathfrak{so}(3)$ the wedge operator has a special name and symbol, commonly referred in the literature as the skew operator, or the cross operator, which is given by

$$(.)_\times : \mathbb{R}^3 \to \mathfrak{so}(3),$$
$$\omega_\times = \omega^\wedge := \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix},$$

We can also apply the wedge operator to other matrix lie groups, for example the special euclidean group

$$(.)^\wedge : \mathbb{R}^6 \to \mathfrak{se}(3)$$
$$\begin{bmatrix} \omega \\ v \end{bmatrix}^\wedge := \begin{bmatrix} \omega_\times & v \\ 0 & 0 \end{bmatrix}$$

and the extended special euclidean group

$$(.)^\wedge : \mathbb{R}^9 \to \mathfrak{se}_2(3)$$
$$\begin{bmatrix} \omega \\ v \\ a \end{bmatrix}^\wedge := \begin{bmatrix} \omega_\times & v & a \\ 0 & 0 & 0 \end{bmatrix}.$$

The wedge and vee notation allows us to represent operators on the Lie algebra as matrix operators on the Lie algebra's vector representation. For example, consider the adjoint representation of $\mathfrak{g}$, which is equivalent to the Lie bracket;

$$\mathrm{ad}(\Gamma, \Psi) = \mathrm{ad}_\Gamma \circ \Psi = [\Gamma, \Psi].$$

We can define a new matrix operator $\check{\mathrm{ad}} \in \mathbb{R}^{n \times n}$, which is related to the original adjoint operator by

$$\check{\mathrm{ad}}_\Gamma \Psi^\vee = (\mathrm{ad}_\Gamma \circ \Psi)^\vee.$$

As an example, the matrix representation of the adjoint operator for $\mathfrak{se}_2(3)$ is given by

$$\check{\mathrm{ad}}_\Gamma := \begin{bmatrix} \Gamma_{R\times} & 0 & 0 \\ \Gamma_{x\times} & \Gamma_{R\times} & 0 \\ \Gamma_{v\times} & 0 & \Gamma_{R\times} \end{bmatrix}.$$

A.3.2 Product group for inertial state estimation

We extend $\mathrm{SE}_2(3)$ with two additional vectors, $\theta, \phi \in \mathbb{R}^3$, to create the direct product group G, which is used to represent the full 15-degrees-of-freedom (DoF) state of a single vehicle, namely the pose, velocity, and IMU sensor biases;

$$\begin{aligned} &\mathrm{G} := \mathrm{SE}_2(3) \times \mathbb{R}^3 \times \mathbb{R}^3, \\ &X = (P, \theta, \phi) \in \mathrm{G}. \end{aligned}$$

For $X, Y \in \mathrm{G}$, the group product operation is

$$X \cdot Y := (P_X P_Y, \theta_X + \theta_Y, \phi_X + \phi_Y)$$

and the lie algebra, $\mathfrak{g}$, has the structure

$$\Gamma = (\Gamma_P, \Gamma_\theta, \Gamma_\phi) \in \mathfrak{g}$$

where

$$\begin{gathered} \Gamma_P = \begin{bmatrix} (\Gamma_R)_\times & \Gamma_x & \Gamma_v \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \in \mathfrak{se}_2(3), \\ \Gamma_R, \Gamma_x, \Gamma_v, \Gamma_\theta, \Gamma_\phi \in \mathbb{R}^3. \end{gathered}$$

This yields the following properties for $X \in \mathrm{G}$ and $\Gamma, \Psi \in \mathfrak{g}$;

$$\begin{aligned} X^{-1} &= (P^{-1}, -\theta, -\phi), \\ [\Gamma, \Psi] &= ([P_\Gamma, P_\Psi], 0, 0), \\ T_e L_X(\Gamma) &= X\Gamma = (P\Gamma_P, \Gamma_\theta, \Gamma_\phi). \end{aligned} \tag{A.6}$$

A.3.3 Exponential Maps

The exponential map, $\exp : \mathfrak{g} \to \mathrm{G}$, for matrix Lie groups coincides with the matrix exponential operator, which is given by

$$\exp(\Gamma) := \sum_{k=0}^{\infty} \frac{1}{k!}\Gamma^k = I + \Gamma + \frac{1}{2}\Gamma^2 + \frac{1}{6}\Gamma^3 + \cdots$$

To disambiguate notation, we also define $\mathrm{Exp} : \mathbb{R}^n \to \mathrm{G}$ as

$$\mathrm{Exp}(\gamma) := \exp\left(\gamma^{\wedge}\right)$$

For the matrix group $\mathrm{SO}(3)$, the exponential map has a closed form, given by

$$\begin{aligned} \exp : \mathfrak{so}(3) &\to \mathrm{SO}(3) \\ \exp(\omega_\times) &= I + \frac{\sin\theta}{\theta}\omega_\times + \frac{1-\cos\theta}{\theta^2}\omega_\times^2 \end{aligned}$$

where $\theta = \|\omega\|$.

For the matrix group $\mathrm{SE}(3)$, the exponential map also has a closed form, given by

$$\begin{aligned} \exp : \mathfrak{se}(3) &\to \mathrm{SE}(3) \\ \exp\left(\begin{bmatrix} \omega_\times & v \\ 0 & 0 \end{bmatrix}\right) &= \begin{bmatrix} \exp(\omega_\times) & Vv \\ 0 & 1 \end{bmatrix} \\ V = I + \frac{1-\cos\theta}{\theta^2}\omega_\times &+ \frac{\theta - \sin\theta}{\theta^3}\omega_\times^2. \end{aligned}$$

On $\mathrm{SE}_2(3)$, the exponential map is similar to that of $\mathrm{SE}(3)$, which is

$$\begin{aligned} \exp : \mathfrak{se}_2(3) &\to \mathrm{SE}_2(3) \\ \exp\left(\begin{bmatrix} \omega_\times & v & a \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}\right) &= \begin{bmatrix} \exp(\omega_\times) & Vv & Va \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \end{aligned}$$

with V defined as above.

For the direct product group $\mathrm{G} = \mathrm{SE}_2(3) \times \mathbb{R}^3 \times \mathbb{R}^3$ from Section A.3.2, the exponential map is expressed as

$$\begin{aligned} \exp : \mathfrak{g} &\to \mathrm{G} \\ \exp(\Gamma) = \exp\left((\Gamma_P, \Gamma_\theta, \Gamma_\phi)\right) &= \left(\exp(\Gamma_P), \exp(\Gamma_\theta), \exp(\Gamma_\phi)\right). \end{aligned}$$

The exponential map for Abelian groups is the identity, which gives

$$\exp(\Gamma) = \left(\exp(\Gamma_P), \Gamma_\theta, \Gamma_\phi\right)$$

where $\exp(\Gamma_P)$ is the exponential map on $\mathrm{SE}_2(3)$.

A.4 Identities on the Cross Product

For $x, y \in \mathbb{R}^3$,

$$x \times y = x_\times y \tag{A.7}$$

$$(x_\times)^\top = -x_\times \tag{A.8}$$

$$x \times y = -y \times x \tag{A.9}$$

$$x_\times y = -y_\times x \tag{A.10}$$

$$(x \times y)_\times = (x_\times y)_\times = [x_\times, y_\times] = x_\times y_\times - y_\times x_\times \tag{A.11}$$

Appendix B

Discussion on Filter Consistency and Conservativeness

Amongst the literature on estimation, filtering, data fusion, and distributed localisation a wide range of terminology and definitions are used to describe certain properties of algorithms. In some cases, the same terminology is used by different fields to represent different concepts, and this leads to contradictory or incompatible statements depending on the interpretation used. In this appendix, we introduce a set of concepts and definitions in an attempt to provide a unified set of terminology with which we can describe filter properties. Mostly, we will adopt accepted terms from the literature and provide precise definitions, but in some cases, we explicitly choose new terms to disambiguate conflicting definitions.

B.1 Dynamical Systems

Consider the dynamical system shown in Figure B.1. The system is composed of the following components;

- the control input, $u \in \mathbb{R}^m$,
- the system state, $x \in \mathbb{R}^n$,
- the measurement, $z \in \mathbb{R}^k$,
- the system model, $g : \mathbb{R}^m \times \mathbb{R}^n \to \mathbb{R}^n$,

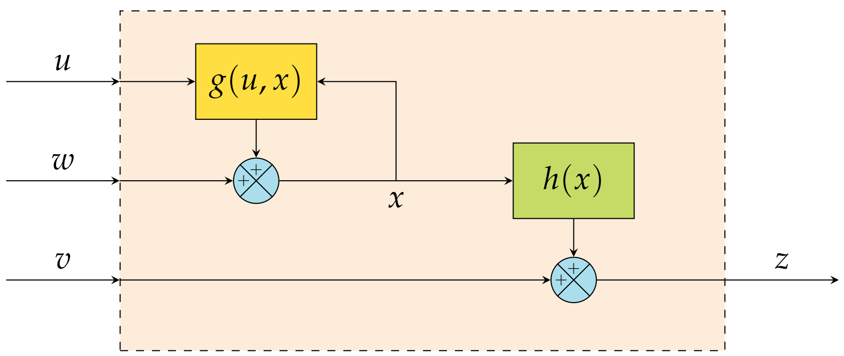


FIGURE B.1: System Block Diagram

- the measurement model, $h : \mathbb{R}^n \to \mathbb{R}^k$,
- the model error, $w \in \mathbb{R}^n$,
- the measurement error, $v \in \mathbb{R}^k$.

As this is a dynamical system, we are interested in how the system evolves over time. For the purposes of our analysis, we will only consider the system evolution in discrete time. We use $t \in \mathbb{N}$ as a time index.

Using the behavioural approach, the system, S, can be described by the tuple

$$S := (\mathbb{T}, \mathbb{M}, \mathbb{L}, \mathfrak{B}_f)$$

where $\mathbb{T}$ is the time axis, $\mathbb{M}$ is the manifest signal space, $\mathbb{L}$ is the latent variable space, and $\mathfrak{B}_f \subseteq (\mathbb{M} \times \mathbb{L})^{\mathbb{T}}$ is called the full behaviour. As we are only considering discrete-time systems, we use the set of natural numbers, $\mathbb{N}$, for the time axis, $\mathbb{T}$.

The manifest signal space is the set of all directly observable system variables. For the system we are describing, we assume that the control input and the measurement are the only directly observed variables, and thus the manifest signal space is given by

$$m := (u, z) \in \mathbb{M} := \mathbb{R}^m \times \mathbb{R}^k$$

We will use the term 'manifest signal', denoted by M, to describe a sequence of manifest variables, such that

$$M := (m_t)_{t \in \mathbb{N}} = (u_t, z_t)_{t \in \mathbb{N}} \in \mathbb{M}^{\mathbb{N}}.$$

The latent variable space represents the remainder of the system variables. Thus, we have

$$(x, w, v) \in \mathbb{L} := \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R}^k$$

A 'latent variable signal' is a sequence of latent variables, denoted by L, where

$$L := (x_t, w_t, v_t)_{t \in \mathbb{N}} \in \mathbb{L}^{\mathbb{N}}.$$

The full behaviour of the system, $\mathfrak{B}_f$, describes which signals are valid. From the system model, we have the constraint

$$x_{t+1} = g(t, x_t, u_t) + w_t, \tag{B.1}$$

and from the measurement model, we have an additional constraint

$$z_t = h(x_t) + v_t. \tag{B.2}$$

Thus, we can define the full behaviour as the set of all signals that satisfy the above constraints,

$$\mathfrak{B}_f := \{(M, L) \in \mathbb{M}^{\mathbb{N}} \times \mathbb{L}^{\mathbb{N}} \mid (\text{B.1}) \text{ and } (\text{B.2}) \text{ are satisfied}\} \tag{B.3}$$

A signal that is an element of the full behaviour is called a trajectory. Each trajectory is composed of the manifest trajectory and the latent variable trajectory.

Subsequently, we define the manifest behaviour (or external behaviour), $\mathfrak{B} \subseteq \mathbb{M}^{\mathbb{T}}$, as

$$\mathfrak{B} := \{M \in \mathbb{M}^{\mathbb{N}} \mid \exists L \in \mathbb{L}^{\mathbb{N}} \text{ such that } (\text{B.1}) \text{ and } (\text{B.2}) \text{ are satisfied}\}$$

In other words, the manifest behaviour is defined as all manifest signals such that, for each of these signals, there exists a latent variable signal that 'explains' the observed data. The key point is that this latent variable trajectory is not necessarily unique. In fact, for a given manifest trajectory, there may be an infinite number of different latent variable trajectories that satisfy the constraints, as there are more free variables than constraints.

A useful concept is a 'maximally free set' of variables, which is a set of unconstrained variables for which corresponds to a unique trajectory in the full behaviour. For this system, there are multiple unique maximally free sets. These include

$$\{U, Z, X \in (\mathbb{R}^m \times \mathbb{R}^k \times \mathbb{R}^n)^{\mathbb{N}}\},$$
$$\{U, X, V \in (\mathbb{R}^m \times \mathbb{R}^n \times \mathbb{R}^k)^{\mathbb{N}}\},$$
$$\{U, W, V, x_0 \in (\mathbb{R}^m \times \mathbb{R}^n \times \mathbb{R}^k)^{\mathbb{N}} \times \mathbb{R}^n\}.$$

Depending on the properties of h and g (e.g. if they are invertible), then there may be other maximally free sets.

A prefix of a signal of length i is a subsequence of the signal from the first element up to and including the i-th element, and is denoted using the subscript i. For example, the prefix of the manifest signal of length i is denoted by

$$M_i := (m_0, m_1, m_2, \ldots, m_i) \in \mathbb{M}^i$$

B.1.1 Modelling the Error Signals

As the behavioural approach emphasises, there is nothing unique about the particular choice of latent variables. The system could just as easily be described with a different set of latent variables and a different model. However, in this particular case, we have constructed the model such that the system state, x, represents some property of the system that we are interested in knowing but which is not directly observable. The system model, g, and the measurement model, h, to describe how the system state interacts with the manifest variables. The additional latent variables, w and v, account for any errors between the modelled system and the true behaviour. For physical systems, this may be due to unmodelled effects such as air resistance, or because a system responds to an input differently than expected. Physical sensors may also introduce errors such as thermal noise into measurements.

As discussed above, there are a potentially infinite number of latent variable trajectories that correspond to a single manifest trajectory. When aiming to provide an estimate of the system state, x, we must select a single trajectory from this set that is 'best' in some way. This is the task of estimation and filtering, and the choice of definition of which trajectory is 'best' results in different filter designs.

When attempting to compare and evaluate particular filter designs, we need to consider their performance over the whole range of full system behaviours. However, it is important to recognise that not all trajectories are equally 'likely' to occur. To make this notion precise, we will add some additional structure to these errors.

Recall the system model error, w, and define the system model error sequence, $W \in \mathbb{W}$, as

$$W := (w_t)_{t\in\mathbb{N}}, \qquad \mathbb{W} := (\mathbb{R}^n)^{\mathbb{N}}.$$

Additionally, we define the measurement error sequence, $V \in \mathbb{V}$ as

$$V := (v_t)_{t\in\mathbb{N}}, \qquad \mathbb{V} := (\mathbb{R}^k)^{\mathbb{N}}.$$

We consider $W : \Omega \to \mathbb{W}$ as a measurable function from a finite measure space $(\Omega, \mathcal{F}, \mu)$ to the measurable space $(\mathbb{W}, \sigma(\mathbb{W}))$, where $\sigma(\mathbb{W})$ denotes the cylinder σ-algebra of $\mathbb{W}$. Similarly, we consider $V : \Omega \to \mathbb{V}$ as a measurable function from the same finite measure space to the measurable space $(\mathbb{V}, \sigma(\mathbb{V}))$. The finite measure, μ, induces the measures μ_W and μ_V on the spaces $(\mathbb{W}, \sigma(\mathbb{W}))$ and $(\mathbb{V}, \sigma(\mathbb{V}))$ respectively.

It will also be useful to consider the initial value of the state, x_0, as a function from $(\Omega, \mathcal{F}, \mu)$ to the measurable space $(\mathbb{R}^n, \mathcal{B}(\mathbb{R}^n))$, where $\mathcal{B}$ denotes the Borel σ-algebra.

We can use these measures to quantify latent trajectories. The measure μ can be interpreted as a measure of the 'probability', 'likelihood', or frequency of occurrence. An equally valid interpretation is to consider μ as an arbitrary weighting on particular trajectories. Regardless of the interpretation, the mathematical description is equivalent.

Definition B.1. *The expectation operator, $\mathbb{E}$, for a measurable function, $f : \Omega \to E$, from the finite measure space $(\Omega, \mathcal{F}, \mu)$ to a measurable space $(E, \mathcal{E})$ is given by*

$$\mathop{\mathbb{E}}_{\omega\in\Omega}[f] = \int_\Omega f(\omega)\, d\mu(\omega),$$

whenever it exists.

While the expectation may not always be defined, hereafter, we will assume that the expectation exists whenever it is used. Unless explicitly noted, the expectation will always be taken over $\omega \in \Omega$, and we will drop the subscript from the $\mathbb{E}$ operator.

B.2 Filters for Dynamical Systems

Although we cannot directly determine the system state, x, from the manifest trajectory, it may be possible to infer the value of the system state. This is the task of state estimation. Commonly, we also wish to perform this task in real-time, which the system is operating. This requires generating the estimate of x_t at time t, using only the information available at time t.

Definition B.2. *A non-anticipating estimator for the system state, x, of the system described by (B.3) at time t is a function, $f_t : \mathbb{M}^t \to \mathbb{R}^n$, given by*

$$\hat{x}_t = f_t(M_t)$$

which produces an estimate, $\hat{x}_t$, of the system state at time t, based on the manifest trajectory up to time t.

A rule for constructing a sequence, $(f_1, f_2, \ldots, f_t)$, of non-anticipating estimators is called a filter.

Definition B.3. *A recursive filter is a rule for constructing a sequence of non-anticipating estimators, where the rule is defined by a recursive function, $f : \mathbb{M} \times \mathbb{R}^n \to \mathbb{R}^n$, given by*

$$\hat{x}_t = f_t(M_t) = f(m_t, \hat{x}_{t-1})$$

and an initial estimate, $\hat{x}_0$.

Note that this implies that two filters with the same recursive rule, but different initial estimates are actually two different filters.

B.2.1 Metrics for Filter Performance

Naturally, an estimator, and by extension a filter, is only useful if the resulting estimate is somewhat close to the true system state. However, quantifying this is non-trivial. One useful metric is the squared-error,

$$e := (x_t - \hat{x}_t)^\top (x_t - \hat{x}_t).$$

Ideally, this value should be close to zero, indicating that the estimate of the filter is close to the true value. However, given that x is a latent variable and may take one of an infinite number of trajectories, we use the expectation operator to quantify the aggregate properties of the filter.

Definition B.4. *A filter is 'convergent' to the true system state in the mean-square sense if*

$$\forall U \in \mathbb{U} : \lim_{t\to\infty} \mathbb{E}\left[(\hat{x}_t - x_t)^\top (\hat{x}_t - x_t)\right] = 0. \tag{B.4}$$

Note how U is constrained in the quantifier, and W, V, and x_0 are constrained by the expectation operator, which together forms a maximally free set, and hence x_t and $\hat{x}_t$ are fully constrained.

This definition is compatible with the traditional definition of estimator 'consistency' on stationary systems. A stationary system is just a special case of the system shown in Figure B.1 where $g(.) = x_{t-1}$ and $w_t = 0$, thus $x_t = x_0$. The above definition of convergence then simplifies to

$$\lim_{t\to\infty} \mathbb{E}\left[(\hat{x}_t - x_0)^\top (\hat{x}_t - x_0)\right] = 0$$

which is the standard mean-square definition of consistency for estimators [141, Equation 2.7.1-2].

Lemma B.5. *If a filter is convergent as per Definition B.4 then*

$$\forall U \in \mathbb{U} : \lim_{t\to\infty} \mathbb{E}\left[\hat{x}_t - x_t\right] = 0. \tag{B.5}$$

Note that the converse of Lemma B.5 is not necessarily true. It also does not guarantee that, for every manifest trajectory, the filter estimate will converge to zero error.

In practice, Definition B.4 is rarely used for two main reasons. Firstly, the squared error will never converge to zero for most filters as the added error introduced by v counteracts the information gained by z. Secondly, we are often more interested in bounds on the finite-time behaviour of a filter rather than the limit at infinity.

B.2.2 Second-Order Filters

A large class of filters, which we will call second-order filters, define a sequence of estimators that produce both an estimate of the state and an estimate, $\hat{\Sigma}$, of the covariance matrix, $\Sigma \in \mathbb{S}^n_{++}$ (the set of positive definite $n \times n$ matrices), defined by

$$\Sigma_t = \mathbb{E}\left[(\hat{x}_t - x_t)(\hat{x}_t - x_t)^\top\right]. \tag{B.6}$$

A recursive second-order filter has the structure

$$f : \mathbb{M} \times \mathbb{R}^n \times \mathbb{S}^n_{++} \to \mathbb{R}^n \times \mathbb{S}^n_{++} \tag{B.7}$$

$$(\hat{x}_t, \hat{\Sigma}_t) = f(m_t, \hat{x}_{t-1}, \hat{\Sigma}_{t-1}) \tag{B.8}$$

with initial estimates $\hat{x}_0$, $\hat{\Sigma}_0$.

Given that Definition B.4 is of little use, we introduce a different property for the finite-time behaviour of second-order filters.

Definition B.6. *A second-order filter is 'weakly-consistent', in the mean-square sense, if for all manifest trajectories, the sequence of estimated covariances produced by the filter, $(\hat{\Sigma})_t$ is identical to the sequence of true covariances, $(\Sigma)_t$, as defined in* (B.6). *More precisely,*

$$\forall (M, t) \in \mathfrak{B} \times \mathbb{N} : \Sigma_t = \hat{\Sigma}_t. \tag{B.9}$$

In the literature, this property is often called 'consistency', however we use 'weak' as qualifier to distinguish this property from estimator consistency, which is more closely related to 'convergence', as in Definition B.4. Note that weak-consistency does not imply convergence and convergence does not imply weak-consistency.

We show in Section B.3 that, under a broad set of assumptions, the Kalman filter is weakly-consistent. However, for almost all other filters, it is not possible to provide an algebraic proof of weak-consistency. If the measurement and model error covariances are not precisely known, or the system is non-linear (e.g. Extended Kalman Filter), then it may not be possible to prove that a given filter is weakly-consistent.

The alternative is to verify filter consistency empirically, either through real-world experiments or in simulation. If the true trajectory, $(x)_t$, of the system is known, it is possible to measure the error in the filter's state estimate and compare it against the filter's estimated covariance. While it may be impossible to evaluate every possible trajectory, With a large enough number of samples it is possible to say with some level of confidence that the filter is consistent. This concept is captured by the Normalised Estimation Error Squared (NEES), which is defined as

$$\epsilon = (\hat{x} - x)^\top \hat{\Sigma}^{-1} (\hat{x} - x).$$

If the induced measures, μ_W, μ_V and μ_{x_0}, are in the form of a Gaussian function, then

$$\mathbb{E}\left[(\hat{x} - x)^\top \Sigma^{-1} (\hat{x} - x) \right] = n,$$

and thus, if $\hat{\Sigma}$ is equal to Σ then ϵ should, on average, be equal to n. The Average Normalised Estimation Error Squared (ANEES), $\bar{\epsilon} = \frac{1}{t} \sum_{i=1}^{t} \epsilon_i$, can be used in a hypothesis test to verify if a filter is consistent (to within a certain confidence level).

If $\bar{\epsilon} > n$, it indicates that the filter is under-estimating the covariance, and the filter is over-confident in its estimate. Correspondingly, if $\bar{\epsilon} < n$ then the filter is over-estimating the covariance matrix and is being conservative in its estimate of the error.

B.2.3 Conservativeness

As discussed above, it is often intractable to prove that a filter is weakly-consistent. However, it may otherwise be possible to prove that the estimated covariance is always an overestimate of the true covariance.

Definition B.7. *A second-order filter is 'conservative', if for all manifest trajectories, every estimated covariance produced by the filter, $\hat{\Sigma}_t$, is greater or equal to the true covariance, Σ_t, in the positive definite sense. i.e.*

$$\forall (M, t) \in \mathfrak{B} \times \mathbb{N} : \hat{\Sigma}_t \succeq \Sigma_t. \tag{B.10}$$

Recall that, for square matrices, A and B, $A \succeq B := (A - B) \succeq 0$.

Lemma B.8. *If a second-order filter is conservative and*

$$\forall M \in \mathfrak{B} : \lim_{t \to \infty} \hat{\Sigma}_t = 0$$

then the filter is convergent.

B.2.4 Non-Destructive Updates

Another property we wish to express is that the fusion of the measurement information, z, should not adversely affect the accuracy of the estimate. In order to describe this, we divide the estimator at each time step into two parts, the prediction step and the update step. Recall from (B.8) the structure of a second-order filter;

$$(\hat{x}_t, \hat{\Sigma}_t) = f(z_t, u_t, \hat{x}_{t-1}, \hat{\Sigma}_{t-1})$$

We split f into the prediction step, f_p, and the update step, f_u, which gives

$$\begin{aligned} (\hat{x}_t^-, \hat{\Sigma}_t^-) &= f_p(u_t, \hat{x}_{t-1}, \hat{\Sigma}_{t-1}) \\ (\hat{x}_t, \hat{\Sigma}_t) &= f_u(z_t, \hat{x}_t^-, \hat{\Sigma}_t^-) \end{aligned}$$

where $\hat{x}_t^-, \hat{\Sigma}_t^-$ denote the intermediate estimates of the system state and covariance before the measurement information is incorporated.

Definition B.9. *A filter is 'non-destructive' if, for some scalar measure m,*

$$\forall (M, t) \in \mathfrak{B} \times \mathbb{N} : m(\hat{\Sigma}_t) \leq \min(m(\hat{\Sigma}_t^-), m(R_t))$$

where $R_i = \mathbb{E}[v_i v_i^\top]$.

One might, for example, use the trace or determinant as the scalar measure, m. Note that this provides an upper bound on the covariance estimate, namely that $m(\hat{\Sigma}_t) \leq m(R_t)$.

B.3 Properties of existing filter designs

In this section, we will review some existing filter designs and see which of them satisfy the definitions provided above. None of the filters discussed are convergent.

B.3.1 Kalman Filter

The Kalman filter is weakly-consistent under the following conditions;

- The system model, g, is linear in x,
- The measurement function, h, is linear in x,
- The covariances of w and v are known ($\mathbb{E}[w_i w_i^\top] = Q_i$, $\mathbb{E}[v_i v_i^\top] = R_i$),
- The errors, w and v are unbiased ($\mathbb{E}[w_i] = 0$, $\mathbb{E}[v_i] = 0$),
- The errors sequences $(w)_t$ and $(v)_t$ are mutually independent ($\forall i, j \in \mathbb{N} : \mathbb{E}[v_i w_j^\top] = 0$),
- The error sequences are white noise ($\forall i, j \in \mathbb{N}, i \neq j : \mathbb{E}[v_i v_j^\top] = 0, \mathbb{E}[w_i w_j^\top] = 0$), and
- The initial estimate of the filter, $(\hat{x}_0, \hat{\Sigma}_0)$, is chosen such that $\mathbb{E}[\hat{x}_0 - x_0] = 0$ and $\hat{\Sigma}_0 = \Sigma_0$.

If all the above conditions hold, then it is possible to directly prove that the Kalman filter satisfies Definition B.6. This is because a linear transformation of the state also results in a linear transformation of the covariance matrix. Note however, that the Kalman filter is not convergent as per Definition B.4 but it does satisfy Lemma B.5.

The Kalman filter is conservative under the same conditions above, except that the covariance estimates may be conservative rather than exact;

$$\hat{\Sigma}_0 \succeq \Sigma_0, \qquad Q_i \succeq \mathbb{E}[w_i w_i^\top], \qquad R_i \succeq \mathbb{E}[v_i v_i^\top].$$

The Kalman filter is also non-destructive. The simplest way to demonstrate this is by considering the inverse form of $\hat{\Sigma}$, commonly called the information form.

B.3.2 Extended Kalman Filter

It is not possible to provide a general proof of weak-consistency for the EKF due to the unquantifiable linearisation error. Instead, the ANEES is often used as a measure of the consistency for a particular implementation of an EKF. It is common practice to inflate the value of Q to account for linearisation errors and other unmodelled effects.

B.3.3 Covariance Intersection

The Kalman filter can only guarantee weak-consistency or conservativeness if the error signals are independent (or if the cross-correlation is exactly known). Alternatively, the Covariance intersection method [86] can be used in the update step of the Kalman filter to guarantee conservativeness regardless of the level of correlation. This modified filter is no longer weakly-consistent, but it remains non-destructive.

B.3.4 Schmidt-Kalman Filter (Consider Kalman Filter)

The Schmidt-Kalman filter (SKF) [51], [142], also referred to as the Consider Kalman filter, is a variation on the Kalman filter where a subset of the state variables are not estimated by the filter, but the uncertainty in the values and the cross-correlations between other state variables are still 'considered'. Brink [99] provides a comprehensive analysis on the consistency of various forms of the Schmidt-Kalman filter. He shows that the SKF is consistent under the same assumptions that the Kalman filter is consistent. Furthermore, it is shown that one can switch back and forth between the SKF update and the KF update for any of the state variables, and consistency is still maintained.

B.3.5 Approximate Schmidt Kalman Filter

Discussed in Section 4.2.3, the Approximate Schmidt Kalman filter (ASKF), introduced by Luft, Schubert *et al.* [98] incorporates a further approximation to the SKF in order to facilitate a distributed implementation. The original formulation of the ASKF is done with the EKF as a base, and so weak-consistency or conservativeness is not guaranteed to begin with. However, the approach could equally be applied to the Kalman filter, which, as we have shown above, does maintain weak-consistency under a set of assumptions. Even if the KF is used as the base filter, it is possible to show by means of a counterexample that neither weak-consistency nor conservativeness are maintained.

Appendix C

Tutorial on Minimum Energy Filtering for Linear Systems

This appendix aims to provide the reader with an introduction to the theory of minimum energy filtering by means of a worked example on a linear system. The concept of minimum-energy filtering has been studied in the literature in many forms over the last 60 years [39], [48], [49], [77], [143]. However, it can be difficult to find a modern and succinct explanation of minimum energy filtering and an insight into some underlying intuitions behind the theory.

The motivation for writing this paper comes from my own journey of learning about minimum energy filtering during the first year of my PhD, and wishing there was a better explanation of the topic. Hopefully this paper makes some progress towards that goal. It is intended to provide the reader with an understanding of the foundational concepts behind minimum-energy filtering and to provide a starting point for further study in the area.

In order to get the most out of this paper, the reader should be broadly familiar with stochastic filtering theory, including the Kalman Filter, as well as optimal control theory, vector calculus, and ordinary differential equations. The derivation presented here closely mirrors Zamani, Trumpf and Mahony [74], but removes a lot of the complexity as we are only considering a linear system.

C.1 Problem Definition

Consider the following continuous-time linear system;

$$\dot{x} = F(t)x(t) + B(t)u(t) + w(t) \tag{C.1}$$

$$x(0) = x_0 \tag{C.2}$$

$$y(t) = H(t)x(t) + v(t) \tag{C.3}$$

where

$x \in \mathbb{R}^n$	System state
$F \in \mathbb{R}^{n \times n}$	System model

$u \in \mathbb{R}^p$	Control input
$B \in \mathbb{R}^{n\times p}$	Input Matrix
$w \in \mathbb{R}^n$	Model error
$y \in \mathbb{R}^m$	Sensor measurement
$H \in \mathbb{R}^{m\times n}$	Measurement model
$v \in \mathbb{R}^m$	Measurement error

We will drop the implicit dependence on t for the remainder of the example. F, B, and H are assumed to be known. It's important to note that, unlike in the derivation of the Kalman filter, we do not assume that w and v are stochastic noise processes (typically Gaussian, zero mean, white processes). Rather, we just consider w and v as deterministic, but unknown error signals. The only assumption we make is that the error signal are square integrable[1].

The filtering problem is to produce an estimate of the system state, $\hat{x}(t)$, which closely follows the true system state $x(t)$. An initial estimate of the system state, $\hat{x}_0$, is known, as well as the sensor measurements and control inputs, $y_{[0,t]}$ and $u_{[0,t]}$ respectively.

We could try to find an explicit solution for $\hat{x}(t)$, however this would be a function of $y_{[0,t]}$ and $u_{[0,t]}$. As t increases, this would require storing and processing an ever-increasing amount of data. Instead, we will aim to find a recursive implementation of the filter, which is to say we wish to find a differential equation of the form $\dot{\hat{x}}(t) = f(x(t), y(t), u(t))$. This means that, if we have an estimate of the state $\hat{x}(t)$, we can integrate (typically numerically) the differential equation forward in time to determine $\hat{x}(t^+)$. This allows us to process measurements in real time and we don't need to store or process the entire measurement history to produce the state estimate $\hat{x}(t)$.

C.1.1 Setting up the Minimum Energy Problem

Recall from Appendix B, that there is a potentially unlimited number of different trajectories of the system state that are compatible with the system and measurement models. The aim of the minimum energy filter is to select a trajectory for the system that such that the error signals are as small as possible. We do this by imposing the following cost functional on the error signals;

$$J_t(x(0), w_{[0,t]}) := \frac{1}{2}\|x(0) - \hat{x}_0\|^2_{K_0} + \int_0^t \frac{1}{2}\|w(\tau)\|^2_Q + \frac{1}{2}\|v(\tau)\|^2_R d\tau \qquad \text{(C.4)}$$

weighted by symmetric matrices K_0, Q and R. The weighted norm is defined by $\|x\|^2_W := x^\top W x$. The first term in (C.4) imposes a cost on how much the starting point of the trajectory, $x(0)$ deviates from the known initial estimate. The remaining terms impose a cost on the energy[2] of the two error signals.

[1] https://en.wikipedia.org/wiki/Square-integrable_function

[2] https://en.wikipedia.org/wiki/Energy_(signal_processing)

This is the origin of the term 'minimum energy' filter.

We need to find a trajectory for x which is consistent with (C.1) and that also minimises the cost functional. Such a trajectory will find a balance between being consistent with both the system model (a small modelling error, w) being consistent with the measurement model (a small measurement error v). We can adjust the weights, K_0, Q and R, depending on how confident we are in our initial estimate, our system model and our sensor measurements respectively.

We will approach this problem in two steps. Firstly, similar to what was discussed above, consider a specific state (x, t). Out of all the possible trajectories for $w_{[0,t]}$, there will be one which has a minimum value for the cost functional. We will introduce the value function, V, which is the value for the cost functional for the trajectory w which minimises J_t;

$$V(x,t) := \min_{w_{[0,t]}} J_t(x, w_{[0,t]}) \tag{C.5}$$

We can now determine the best estimate of $x(t)$ by selecting a state which minimises the value function.

$$\hat{x}(t) := \arg\min_{x} V(x,t) \tag{C.6}$$

C.2 Finding a Solution

Ultimately, we wish to derive a differential equation to describe how the estimate of the state, $\hat{x}$, changes over time as a function of the system input, u, and the measurement information, y. Our approach will utilise the techniques and theory from optimal control, as we can analogously consider this as an optimal control problem where v is the tracking error and w is the control input. One of the key ideas that our derivation relies on is the Hamilton-Jacobi-Bellman (HJB) Equation

One necessary condition that we will utilise is that the derivative of the value function at the minimum will be zero.

$$\left.\frac{\mathrm{d}}{\mathrm{d}x} V(x,t)\right|_{x=\hat{x}(t)} = 0 \tag{C.7}$$

C.2.1 The Optimal Hamiltonian

The Hamiltonian for this system is defined as

$$\mathcal{H}(x, \mu, w, t) := \frac{1}{2}\|w\|_Q^2 + \frac{1}{2}\|v\|_R^2 - \mu^\top (Fx + Bu + w) \tag{C.8}$$

where $\mu \in \mathbb{R}^n$ is the Lagrange multiplier. The terms in the Hamiltonian come from the system model (C.1) and the integrand of the cost functional (C.4). If the reader is unfamiliar with Optimal Control Theory, Kirk [144] provides a comprehensive introduction to the topic and discusses the properties of the Hamiltonian.

Based on the Pontryagin Maximum Principle, the derivative of the Hamiltonian with respect to w is zero;

$$0 = \frac{\mathrm{d}}{\mathrm{d}w}\mathcal{H}(x, \mu, w, t) \tag{C.9}$$

$$0 = \frac{\mathrm{d}}{\mathrm{d}w}\left(\frac{1}{2}\|w\|_Q^2 + \frac{1}{2}\|v\|_R^2 - \mu^\top(Fx + Bu + w)\right) \tag{C.10}$$

We use (C.3) to replace v

$$0 = \frac{\mathrm{d}}{\mathrm{d}w}\left(\frac{1}{2}\|w\|_Q^2 + \frac{1}{2}\|y - Hx\|_R^2 - \mu^\top(Fx + Bu + w)\right) \tag{C.11}$$

Evaluating the derivative[3] and solving for w gives

$$0 = w^\top Q - \mu^\top \tag{C.12}$$

$$w = Q^{-1}\mu \tag{C.13}$$

We can now substitute the value for w into the Hamiltonian to create the optimal Hamiltonian, $\mathcal{H}^*$

$$\mathcal{H}^*(x, \mu, t) = \frac{1}{2}\left\|-Q^{-1}\mu\right\|_Q^2 + \frac{1}{2}\|y - Hx\|_R^2 - \mu^\top(Fx + Bu + Q^{-1}\mu) \tag{C.14}$$

Simplifying

$$\mathcal{H}^*(x, \mu, t) = \frac{1}{2}\|\mu\|_{Q^{-1}}^2 + \frac{1}{2}\|y - Hx\|_R^2 - \mu^\top(Fx + Bu + Q^{-1}\mu) \tag{C.15}$$

$$\boxed{\mathcal{H}^*(x, \mu, t) = -\frac{1}{2}\|\mu\|_{Q^{-1}}^2 + \frac{1}{2}\|y - Hx\|_R^2 - \mu^\top(Fx + Bu)} \tag{C.16}$$

The Hamiltonian is a useful concept as it relates to the Hamilton-Jacobi-Bellman Equation

$$\mathcal{H}^*(x, \nabla_x V(x, t), t) - \frac{\partial}{\partial t}V(x, t) = 0 \tag{C.17}$$

In the literature on the HJB equation, it is common to see a plus sign instead of the minus sign in the equation above. To paraphrase Saccon, Trumpf *et al.* [77]; the optimal control filtering problem of (C.6) can be thought of as a standard optimal control problem which is solved backwards in time. In this interpretation, the term $\|x(0) - \hat{x}_0\|_{K_0}^2$ in (C.4) can be considered the terminal

[3] Wikipedia contains a list of many useful vector calculus identities

cost, and $V(x,t)$ as the cost-to-go. This justifies the presence of the minus sign in (C.17) rather than the standard form of the HJB equation.

For ease of notation, we let $\mu(x,t) := \nabla_x V(x,t) = \frac{\mathrm{d}}{\mathrm{d}x} V(x,t)^\top$ which equivalently gives

$$\mathcal{H}^*(x,\mu(x,t),t) = \frac{\partial}{\partial t} V(x,t) \tag{C.18}$$

C.2.2 Derivative of the Optimal Hamiltonian

In subsequent steps of the derivation, we will need to calculate the derivative of the Optimal Hamiltonian with respect to x where μ is also a function of x. Applying the chain rule, we have

$$\frac{\mathrm{d}}{\mathrm{d}x}\mathcal{H}^*(x,\mu(x,t),t) = \frac{\partial}{\partial x}\mathcal{H}^*(x,\mu(x,t),t) + \frac{\partial}{\partial \mu}\mathcal{H}^*(x,\mu(x,t),t)\frac{\mathrm{d}}{\mathrm{d}x}\mu(x,t) \tag{C.19}$$

Evaluating just the first term in (C.19), we have

$$\frac{\partial}{\partial x}\mathcal{H}^*(x,\mu(x,t),t) = \frac{\partial}{\partial x}\left(-\frac{1}{2}\|\mu(x,t)\|^2_{Q^{-1}} + \frac{1}{2}\|y - Hx\|^2_R - \mu(x,t)^\top(Fx + Bu)\right) \tag{C.20}$$

$$= -(y - Hx)^\top RH - \mu(x,t)^\top F \tag{C.21}$$

Similarly, evaluating the second term in (C.19) gives

$$\frac{\partial}{\partial \mu}\mathcal{H}^*(x,\mu(x,t),t) = \frac{\partial}{\partial \mu}\left(-\frac{1}{2}\|\mu(x,t)\|^2_{Q^{-1}} + \frac{1}{2}\|y - Hx\|^2_R - \mu(x,t)^\top(Fx + Bu)\right) \tag{C.22}$$

$$= -\mu(x,t)^\top Q^{-1} - (Fx + Bu)^\top \tag{C.23}$$

Substituting (C.21) and (C.23) back into (C.19) gives

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}x}\mathcal{H}^*(x,\mu(x,t),t) &= -(y - Hx)^\top RH - \mu(x,t)^\top F \\ &\quad - \left(\mu(x,t)^\top Q^{-1} + (Fx + Bu)^\top\right)\frac{\mathrm{d}}{\mathrm{d}x}\mu(x,t)\end{aligned} \tag{C.24}$$

C.2.3 Solving for the State Estimate

Consider the derivative of the value function with respect to both x and t, and apply the chain rule;

$$\frac{\mathrm{d}}{\mathrm{d}x}\frac{\mathrm{d}}{\mathrm{d}t}V(x,t) = \frac{\mathrm{d}}{\mathrm{d}x}\left(\frac{\partial}{\partial t}V(x,t) + \frac{\partial}{\partial x}V(x,t)\frac{\mathrm{d}}{\mathrm{d}t}x\right) \tag{C.25}$$

Substituting the first term with the HJB equation (C.18) and applying the product rule to the second term gives

$$= \frac{\mathrm{d}}{\mathrm{d}x}\mathcal{H}^*(x,\mu(x,t),t) + \frac{\partial}{\partial x}V(x,t)\frac{\mathrm{d}}{\mathrm{d}x}\left(\frac{\mathrm{d}}{\mathrm{d}t}x\right) + \left(\frac{\mathrm{d}}{\mathrm{d}t}x\right)^\top \frac{\mathrm{d}}{\mathrm{d}x}\left(\frac{\partial}{\partial x}V(x,t)\right)^\top \tag{C.26}$$

Recall from (C.7) that, at the optimal trajectory, the derivative w.r.t. x of the value function is zero. Therefore,

$$\frac{\mathrm{d}}{\mathrm{d}t}\frac{\mathrm{d}}{\mathrm{d}x}V(x,t)\bigg|_{x=\hat{x}(t)} = 0 \tag{C.27}$$

Substituting (C.24) into (C.26) and then evaluating at $x = \hat{x}(t)$ gives

$$\begin{aligned} 0 = -(y - H\hat{x})^\top RH - \mu(\hat{x},t)^\top F - \left(\mu(\hat{x},t)^\top Q^{-1} + (F\hat{x} + Bu)^\top\right)\frac{\mathrm{d}^2}{\mathrm{d}x^2}V(\hat{x},t) \\ + \mu(\hat{x},t)^\top \frac{\mathrm{d}}{\mathrm{d}x}\dot{\hat{x}} + \dot{\hat{x}}^\top \frac{\mathrm{d}^2}{\mathrm{d}x^2}V(\hat{x},t) \end{aligned} \tag{C.28}$$

where we have used the shorthand notation $\frac{\mathrm{d}^2}{\mathrm{d}x^2}f(x) = \frac{\mathrm{d}}{\mathrm{d}x}\left(\frac{\mathrm{d}}{\mathrm{d}x}f(x)\right)^\top$ and $\dot{x} = \frac{\mathrm{d}}{\mathrm{d}t}x$. From (C.7), we note that $\mu(\hat{x},t) = 0$, and we can simplify to

$$0 = -(y - H\hat{x})^\top RH - (F\hat{x} + Bu)^\top \frac{\mathrm{d}^2}{\mathrm{d}x^2}V(\hat{x},t) + \dot{\hat{x}}^\top \frac{\mathrm{d}^2}{\mathrm{d}x^2}V(\hat{x},t) \tag{C.29}$$

Solving for $\dot{\hat{x}}$ and introducing $K = \frac{\mathrm{d}^2}{\mathrm{d}x^2}V(\hat{x},t)$, we have

$$\dot{\hat{x}}^\top K = (y - H\hat{x})^\top RH + (F\hat{x} + Bu)^\top K \tag{C.30}$$

$$\boxed{\dot{\hat{x}} = F\hat{x} + Bu + K^{-1}H^\top R(y - H\hat{x})}. \tag{C.31}$$

We also need to determine the initial condition of the ODE, which is

$$\begin{aligned} \hat{x}(0) &= \arg\min_x V(x,0) && \text{(C.32)} \\ &= \arg\min_x \frac{1}{2}\|x - \hat{x}_0\|^2_{\Sigma_0} && \text{(C.33)} \\ &= \hat{x}_0. && \text{(C.34)} \end{aligned}$$

C.2.4 Determining the Hessian of the Value Function

Note in (C.31) that the term K, the Hessian of the value function, is still undetermined. We can describe K in terms of an ODE and an initial condition.

Recall that

$$K = \frac{\mathrm{d}^2}{\mathrm{d}x^2} V(x,t)\bigg|_{x=\hat{x}(t)} \tag{C.35}$$

Taking the total time derivative of K and applying the chain rule,

$$\frac{\mathrm{d}}{\mathrm{d}t} K = \frac{\partial}{\partial x}\left(\frac{\mathrm{d}^2}{\mathrm{d}x^2} V(x,t)\right)\frac{\mathrm{d}}{\mathrm{d}t}x + \frac{\partial}{\partial t}\left(\frac{\mathrm{d}^2}{\mathrm{d}x^2} V(x,t)\right) \tag{C.36}$$

$$= \frac{\mathrm{d}}{\mathrm{d}x} K \frac{\mathrm{d}}{\mathrm{d}t} x + \frac{\mathrm{d}^2}{\mathrm{d}x^2}\frac{\partial}{\partial t} V(x,t) \tag{C.37}$$

Considering just the second term

$$\frac{\mathrm{d}^2}{\mathrm{d}x^2}\frac{\partial}{\partial t} V(x,t) = \frac{\mathrm{d}^2}{\mathrm{d}x^2}\mathcal{H}(x,\mu(x,t),t) \tag{C.38}$$

$$= \frac{\mathrm{d}}{\mathrm{d}x}\left(\frac{\mathrm{d}}{\mathrm{d}x}\mathcal{H}(x,\mu(x,t),t)\right)^\top \tag{C.39}$$

We can use the results from (C.24)

$$\frac{\mathrm{d}^2}{\mathrm{d}x^2}\frac{\partial}{\partial t} V(x,t) = \frac{\mathrm{d}}{\mathrm{d}x}\Bigg(-(y-Hx)^\top RH - \mu(x,t)^\top F - (\mu(x,t)^\top Q^{-1} + (Fx+Bu)^\top)\frac{\mathrm{d}}{\mathrm{d}x}\mu(x,t)\Bigg)^\top \tag{C.40}$$

$$\frac{\mathrm{d}^2}{\mathrm{d}x^2}\frac{\partial}{\partial t} V(x,t) = \frac{\mathrm{d}}{\mathrm{d}x}\Bigg(-H^\top R(y-Hx) - F^\top \mu(x,t) - (\frac{\mathrm{d}}{\mathrm{d}x}\mu(x,t))^\top\left(Q^{-1}\mu(x,t) + (Fx+Bu)\right)\Bigg) \tag{C.41}$$

Evaluating the derivative, we have

$$= H^\top RH - F^\top \frac{\mathrm{d}}{\mathrm{d}x}\mu(x,t) - (\frac{\mathrm{d}}{\mathrm{d}x}\mu(x,t))^\top F - (\frac{\mathrm{d}}{\mathrm{d}x}\mu(x,t))^\top Q^{-1}\frac{\mathrm{d}}{\mathrm{d}x}\mu(x,t) + \mu(x,t)^\top Q^{-1}\frac{\mathrm{d}}{\mathrm{d}x}\left(\frac{\mathrm{d}}{\mathrm{d}x}\mu(x,t)\right) - (Fx+Bu)^\top\frac{\mathrm{d}}{\mathrm{d}x}\left(\frac{\mathrm{d}}{\mathrm{d}x}\mu(x,t)\right) \tag{C.42}$$

$$= H^\top RH - F^\top K - KF - KQ^{-1}K + \mu(x,t)^\top Q^{-1}\frac{\mathrm{d}}{\mathrm{d}x}K - (Fx+Bu)^\top\frac{\mathrm{d}}{\mathrm{d}x}K \tag{C.43}$$

Substituting back into (C.37) and evaluating at $x = \hat{x}(t)$ results in

$$\dot{K} = H^\top RH - F^\top K - KF - KQ^{-1}K + O(\frac{\mathrm{d}}{\mathrm{d}x}K) \tag{C.44}$$

where $O(\frac{\mathrm{d}}{\mathrm{d}x}K)$ represents terms containing the derivative of K, which is the third derivative of the value function. For a linear system, it is possible to show that the value function is quadratic in x, and thus the third-order derivative is identically zero. Sontag [143] shows this property in Chapter 8, specifically in Theorem 38 and Equation 8.47. Given this, the resulting ODE for K is

$$\boxed{\dot{K} = H^\top RH - F^\top K - KF - KQ^{-1}K} \tag{C.45}$$

The initial condition for K is straightforward to derive

$$\begin{aligned} K(0) &= \frac{\mathrm{d}^2}{\mathrm{d}x^2}V(x,0) && \text{(C.46)}\\ &= \frac{1}{2}\|x - x_0\|^2_{K_0} && \text{(C.47)}\\ &= K_0 && \text{(C.48)} \end{aligned}$$

We observe that, in (C.31), the K term is inverted. Instead of performing this inversion, we can also just define an ODE for K^{-1}.

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t}K^{-1} &= -K^{-1}\frac{\mathrm{d}}{\mathrm{d}t}[K]K^{-1} && \text{(C.49)}\\ &= -K^{-1}\left(H^\top RH - F^\top K - KF - KQ^{-1}K\right)K^{-1} && \text{(C.50)} \end{aligned}$$

If we substitute $\Sigma = K^{-1}$, we get the familiar form of the filter;

$$\boxed{\begin{aligned} \dot{\hat{x}} &= F\hat{x} + Bu + \Sigma H^\top R(y - H\hat{x}) && \text{(C.51)}\\ \dot{\Sigma} &= Q^{-1} + \Sigma F^\top + F\Sigma - \Sigma H^\top RH\Sigma && \text{(C.52)} \end{aligned}}$$

C.3 Conclusions

C.3.1 Comparison with the Kalman-Bucy Filter

It should come as no surprise that the minimum-energy filter estimate has the same form as the Kalman-Bucy filter [47] which is the optimal minimum-variance estimator for the same linear system.

Recall that the formulation of our problem is purely deterministic, with no concept of random variables or covariance matrices. However, if we match up all the terms in the minimum energy filter with the Kalman-Bucy filter, however we can observe the parallels to the stochastic interpretation. Σ is equivalent to the covariance of the estimate, while Q^{-1} and R^{-1} map to the process noise covariance and the measurement noise covariance respectively. This matches up well to the intuitions in the deterministic system. Σ describes the inverse of the Hessian of the value function. If the Hessian is large, it means that a small change in the estimate in any direction is going to significantly increase the cost functional, and that new measurement data is unlikely to alter the state estimate significantly. This is equivalent to the covariance of the state estimate being small, which also indicates a high degree of confidence in the current estimate.

C.3.2 Extending the Minimum Energy Filter

One might come to the conclusion that the minimum energy filter is just a reinterpretation or an alternative derivation of the Kalman-Bucy filter, and that there's nothing particularly different about the two approaches. However, this is only true in the linear system case. Where the two approaches begin to differ is in the non-linear case.

The standard approach for dealing with non-linear systems in the stochastic framework is to linearise the system about the current state estimate, and then apply a standard linear Kalman filter, an approach known as the Extended Kalman Filter (EKF). This simple approach is often very effective for systems that are close to being linear, but when the linearisation error is high the filter behaviour is erratic and can diverge.

The derivation presented in this paper works follows through in a very similar process for non-linear systems, and does not require the same linearisation process that is used in the EKF. The key change that occurs when moving to non-linear systems is that we can no longer guarantee that the Value function is quadratic in x, which means that we cannot cancel the third-order terms out of (C.44). We could instead find a differential equation to describe the third order derivative, but this would contain terms of fourth-order. And solving for the fourth-order term requires the fifth-derivative and so on. Instead, one approach is to assume that these higher-order derivatives are negligible and can be discarded from the ODE anyway. If we do this, the filter no longer provides the optimal minimum-energy solution to the problem, and is called the 'second-order optimal minimum energy filter'.

Another advantage of the minimum energy filter is that it allows us to work directly with non-Euclidean state spaces such as Lie groups. For example, rather than x being an element of $\mathbb{R}^n$, we can consider state spaces such as rotation matrices, $\mathrm{SO}(3)$, or poses, $\mathrm{SE}(3)$. Zamani [73] shows an example of a minimum energy filter on $\mathrm{SO}(3)$, and Saccon, Trumpf *et al.* [77] generalises the filter for arbitrary Lie Groups. These derivations require an understanding of differential geometry, but there are no fundamental differences with the derivation presented in this paper.

Appendix D

Errors Identified in the MRCLAM Dataset

Despite its popularity, the University of Toronto Institute for Aerospace Studies (UTIAS) Multi-Robot Cooperative Localisation and Mapping (MRCLAM) dataset [101] contains a number of errors that do not appear to have been identified by the community. In the brief appendix, we detail some of the errors that we identified when analysing the data.

File Mislabelling

In the `MRCLAM4.zip` file downloaded from the website, the folder containing the data is mislabelled as `MRSLAM_Dataset4`. This becomes an issue when writing automated scripts to process the data which is expecting a certain format.

Landmark Ground Truth Data

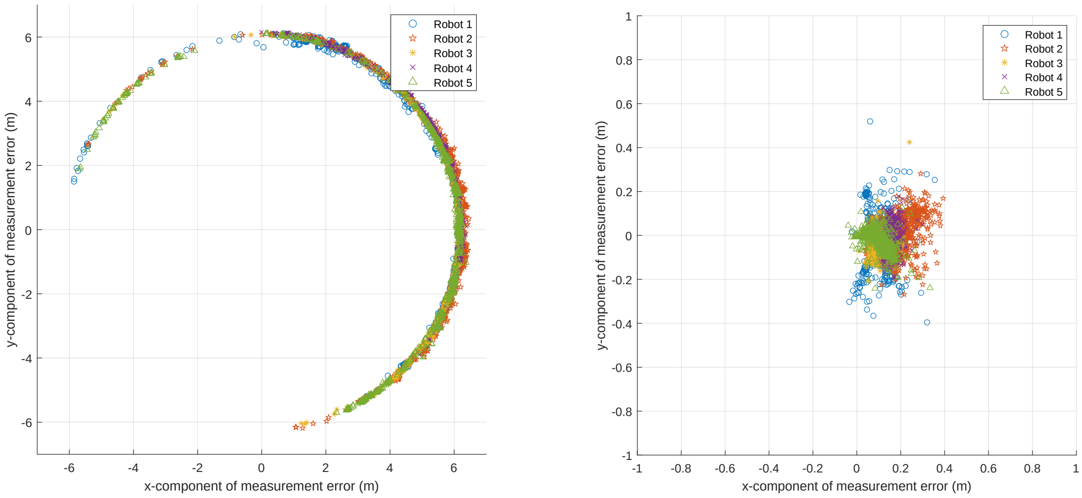
In Dataset 1, it is highly likely that the ground truth positions for Landmark 11 and Landmark 17 have been transposed. If we examine the relative position measurement errors for these landmarks, shown in Figure D.1a, it is clear there is some misconfiguration in the dataset. Swapping the ground truth positions for Landmark 11 and 17 appears to rectify the issue, as seen in Figure D.1b, where the errors show a much more consistent behaviour.

Range Measurement Interpretation

There appears to be ambiguity around the components of the measurements recorded by the camera on each vehicle. From the paper explaining the dataset structure, they state

> The measurement data file for each robot contains the timestamped range and bearing (r, θ) measurements to particular subjects. [101]

and the standard interpretation of this is that the range value represents the distance from the optical centre of the camera to the target, which is illustrated in Figure D.2. However, if we analyse the sensor errors, we can see a clear



(A) Using landmark ground truth from the dataset

(B) Landmark 11 and 17 ground truth positions swapped

FIGURE D.1: A plot of relative measurement errors in MRCLAM Dataset 1, showing only measurements made to landmarks 11 and 17.

relationship between the range error and the bearing value, which is shown in Figure D.3a. This exact same behaviour was identified by Sullivan, Grainger and Cazzolato [102], and they attempt to normalise the error by fitting a polynomial to the trend.

Instead, the more likely source of this error is due to a misinterpretation of the data. Instead of range measurements, what the dataset actually contains is depth measurements, which is the distance from the image plane to the target, as illustrated in Figure D.2. If we instead interpret the data in this way, the dependence between the range error and bearing value disappears, as shown in Figure D.3b.

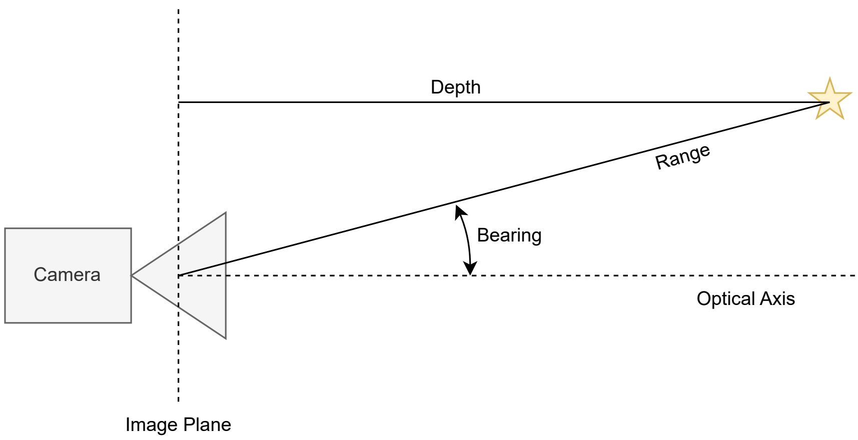
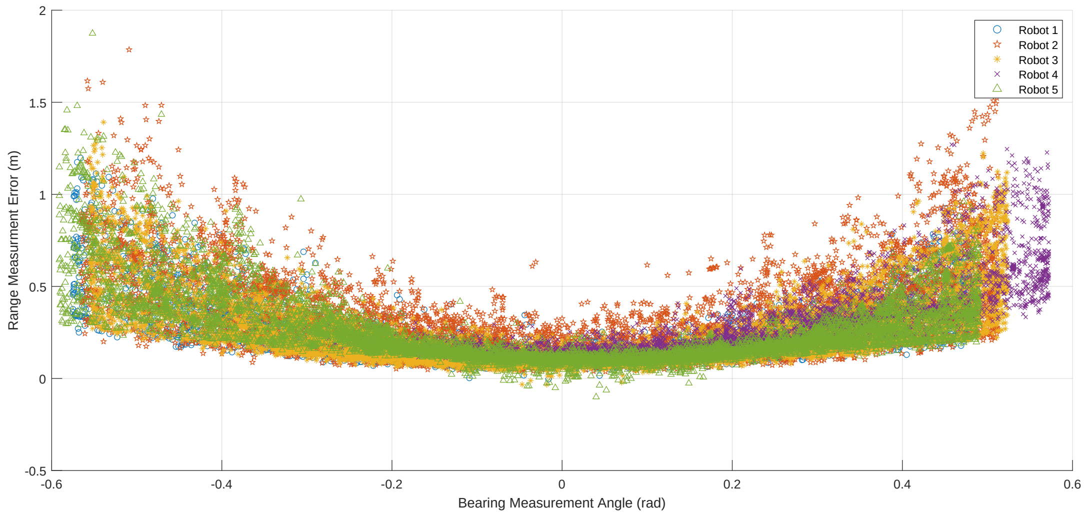
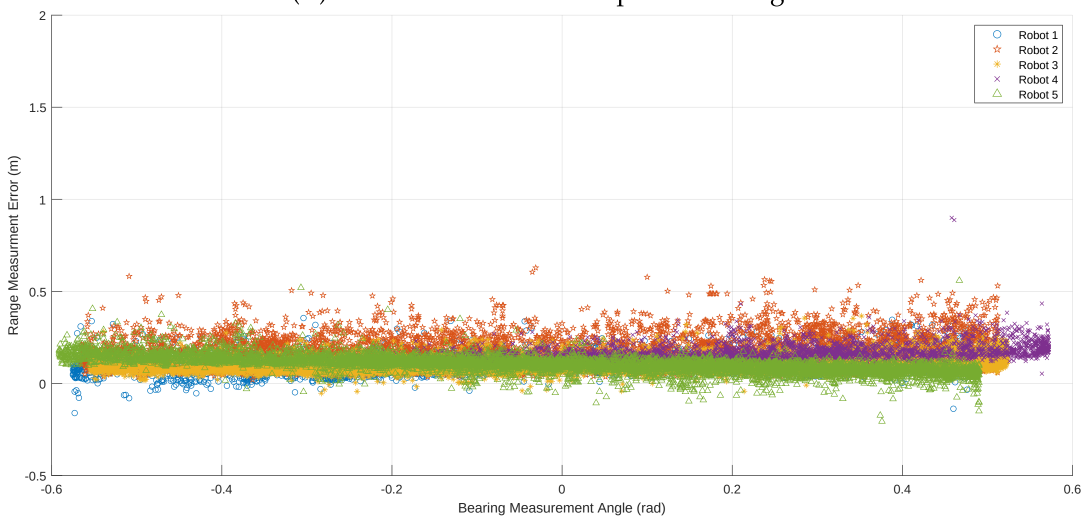


FIGURE D.2: Diagram illustrating the difference between range and depth measurements.



(A) Measurement data interpreted as range



(B) Measurement data interpreted as depth

FIGURE D.3: A plot of the range measurement error vs the bearing angle of all the measurements in MRCLAM Dataset 1.

TABLE D.1: Number of measurements for which the decoded barcode is invalid

	Robot 1	Robot 2	Robot 3	Robot 4	Robot 5
Dataset 1			2		1
Dataset 2			4		7
Dataset 3	6	1			9
Dataset 4	5	1			3
Dataset 5	5		3		1
Dataset 6	1		2	3	
Dataset 7			9		
Dataset 8	605	897	1975	828	1301
Dataset 9	1				1

Landmark Identification Errors

Within each dataset, there are several measurements where the decoded barcode does not match with any of the known landmarks or robots. The number of invalid measurements in each dataset is shown in Table D.1. Note the high number of invalid measurements in Dataset 8. This is likely because Landmark 20 is not included in the ground truth for this dataset, but from the measurement data, it was clearly visible by all the robots. Without the ground truth for the landmark, over 5500 measurements in the dataset must be discarded.

Bibliography

[1] J. Henderson, J. Trumpf and M. Zamani, "A Minimum Energy Filter for Distributed Multirobot Localisation," in *IFAC-PapersOnLine*, vol. 53, 2020, pp. 4916–4922. DOI: 10.1016/j.ifacol.2020.12.1068.

[2] J. Henderson, M. Zamani, R. Mahony and J. Trumpf, "A Minimum Energy Filter for Localisation of an Unmanned Aerial Vehicle," in *2020 59th IEEE Conference on Decision and Control (CDC)*, Jeju Island, Korea (South): IEEE, Dec. 2020, pp. 4188–4193. DOI: 10.1109/CDC42340.2020.9303730.

[3] J. Henderson, M. Zamani, R. Mahony and J. Trumpf, *Inertial Collaborative Localisation for Autonomous Vehicles using a Minimum Energy Filter*, Apr. 2021. arXiv: 2104.05897. [Online]. Available: https://arxiv.org/abs/2104.05897v1.

[4] Z. Wang, Y. Ng, J. Henderson and R. Mahony, "Smart Visual Beacons with Asynchronous Optical Communications using Event Cameras," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct. 2022, pp. 3793–3799. DOI: 10.1109/IROS47612.2022.9982016.

[5] V. Zhou, "BHP to roll out autonomous trucks at QLD coal mine," *Australian Mining*, Nov. 2019.

[6] K. Sheikh, "A Growing Presence on the Farm: Robots," *The New York Times*, Feb. 2020.

[7] Zipline International Inc., *Zipline Widens Access to Malaria Vaccine for Children in Ghana*, May 2022. [Online]. Available: https://web.archive.org/web/20221202094629/https://www.flyzipline.com/post/zipline-widens-access-to-malaria-vaccine-for-children-in-ghana (visited on 06/10/2022).

[8] D. Gleeson, "BHP looks at phased rollout of autonomous trucks at WA iron ore ops," *International Mining*, Feb. 2019.

[9] M. Murison, *Agtech: How robotics could dig farmers out of Brexit hole*, Apr. 2018. [Online]. Available: https://internetofbusiness.com/agtech-robotics-farmers-brexit-hole/ (visited on 07/10/2022).

[10] Zipline International Inc., *Zipline takes off in Japan!* Apr. 2022. [Online]. Available: https://web.archive.org/web/20221202083615/https://www.flyzipline.com/post/zipline-takes-off-in-japan (visited on 07/10/2022).

[11] E. Hecht, "Drones in the Nagorno-Karabakh War: Analyzing the Data," *Military Strategy Magazine*, vol. 7, no. 4, 2022.

[12] M. Hellyer, *US to 'pause' production of Australia's Triton drones*, Feb. 2020. [Online]. Available: https://www.aspistrategist.org.au/us-to-pause-production-of-australias-triton-drones/ (visited on 27/10/2022).

[13] D. Sheridan, "Inside Ukrainian drone school where being a PlayStation guru counts as a useful skill," *The Telegraph*, Aug. 2022.

[14] M. Hieslmair, *Drone 100: A World Record with 100 Points*, Dec. 2016. [Online]. Available: https://ars.electronica.art/aeblog/en/2016/01/12/drone100/ (visited on 19/10/2023).

[15] B. Barrett, "Inside the Olympics Opening Ceremony World-Record Drone Show," *Wired*, Sep. 2018.

[16] Guinness World Records Limited, *Most remote controlled multirotors/drones airborne simultaneously*, May 2021. [Online]. Available: https://www.guinnessworldrecords.com/world-records/373319-most%C2%A0unmanned-aerial%C2%A0vehicles%C2%A0uavs-airborne%C2%A0simultaneously%C2%A05-kg-or-less (visited on 19/10/2023).

[17] Z. Yan, N. Jouandeau and A. A. Cherif, "A Survey and Analysis of Multi-Robot Coordination," *International Journal of Advanced Robotic Systems*, vol. 10, no. 12, p. 399, Dec. 2013. DOI: 10.5772/57313.

[18] C. Gartenberg, *Intel's Winter Olympics light show featured a record-breaking 1,218 drones*, Feb. 2018. [Online]. Available: https://www.theverge.com/2018/2/9/16994638/winter-olympics-2018-intel-drone-show-world-record (visited on 07/10/2022).

[19] J. A. Boyd, *A Discourse on Winning and Losing*, G. T. Hammond, Ed. Air University Press, 2018, ISBN: 978-1-58566-279-1.

[20] L. E. Navarro-Serment, C. J. J. Paredis and P. K. Khosla, "A Beacon System for the Localization of Distributed Robotic Teams," in *Proceedings of the International Conference on Field and Service Robotics*, vol. 6, 1999.

[21] F. Thomas and L. Ros, "Revisiting trilateration for robot localization," *IEEE Transactions on Robotics*, vol. 21, no. 1, pp. 93–101, Feb. 2005. DOI: 10.1109/TRO.2004.833793.

[22] R. I. Hartley and P. Sturm, "Triangulation," *Computer Vision and Image Understanding*, vol. 68, no. 2, pp. 146–157, Nov. 1997. DOI: 10.1006/cviu.1997.0547.

[23] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2003, ISBN: 978-0-521-54051-3.

[24] M. Kayton and W. R. Fried, Eds., *Avionics Navigation Systems*, 2nd ed. New York: Wiley, 1997, ISBN: 978-0-471-54795-2.

[25] *Vicon — Award Winning Motion Capture Systems*, Jan. 2023. [Online]. Available: https://www.vicon.com/ (visited on 20/01/2023).

[26] OptiTrack, *Motion Capture Systems*, Jan. 2023. [Online]. Available: http://optitrack.com/index.html (visited on 20/01/2023).

[27] J. Crowley, "Asynchronous control of orientation and displacement in a robot vehicle," in *1989 International Conference on Robotics and Automation Proceedings*, May 1989, 1277–1282 vol.3. DOI: 10.1109/ROBOT.1989.100156.

[28] D. H. Titterton and J. L. Weston, *Strapdown Inertial Navigation Technology* (IEE Radar, Sonar, Navigation, and Avionics Series 17), 2nd ed. Stevenage: Institution of Electrical Engineers, 2004, ISBN: 978-0-86341-358-2.

[29] B. Barshan and H. Durrant-Whyte, "Inertial navigation systems for mobile robots," *IEEE Transactions on Robotics and Automation*, vol. 11, no. 3, pp. 328–342, Jun. 1995. DOI: 10.1109/70.388775.

[30] A. D. King, "Inertial Navigation - Forty Years of Evolution," *GEC REVIEW*, vol. 13, no. 3, p. 10, 1998.

[31] International Civil Aviation Organisation, "Destruction of Korean Airlines Boeing 747 on 31 August 1983," Tech. Rep., 1993, p. 298.

[32] R. Witkin, "Air Experts Work on New Safety Plan," *The New York Times*, Sep. 1983.

[33] Mgarin73, *Flight map of Korean Airlines Flight 007*, Jan. 2011. [Online]. Available: https://commons.wikimedia.org/wiki/File:KAL007.svg (visited on 16/09/2022).

[34] P. D. Groves, *Principles of GNSS, Inertial, and Multisensor Integrated Navigation Systems* (GNSS Technology and Applications Series). Boston: Artech House, 2008, ISBN: 978-1-58053-255-6.

[35] D. Hall and J. Llinas, Eds., *Multisensor Data Fusion*. CRC Press, Jun. 2001, ISBN: 978-0-429-12747-2. DOI: 10.1201/9781420038545.

[36] J. Moore, "Persistence of excitation in extended least squares," *IEEE Transactions on Automatic Control*, vol. 28, no. 1, pp. 60–68, Jan. 1983. DOI: 10.1109/TAC.1983.1103142.

[37] B. D. Anderson and J. B. Moore, *Optimal Filtering* (Prentice-Hall Information and System Sciences Series). Englewood Cliffs, NJ: Prentice-Hall, 1979, ISBN: 978-0-13-638122-8.

[38] A. H. Jazwinski, *Stochastic Processes and Filtering Theory* (Mathematics in Science and Engineering 64), 12. [print]. San Diego: Acad. Press, 1997, ISBN: 978-0-12-381550-7.

[39] R. E. Mortensen, "Maximum-likelihood recursive nonlinear filtering," *Journal of Optimization Theory and Applications*, vol. 2, no. 6, pp. 386–394, Nov. 1968. DOI: 10.1007/BF00925744.

[40] R. Mahony, T. Hamel and J.-M. Pflimlin, "Nonlinear Complementary Filters on the Special Orthogonal Group," *IEEE Transactions on Automatic Control*, vol. 53, no. 5, pp. 1203–1218, Jun. 2008. DOI: 10.1109/TAC.2008.923738.

[41] M. Dissanayake, P. Newman, S. Clark, H. Durrant-Whyte and M. Csorba, "A solution to the simultaneous localization and map building (SLAM) problem," *IEEE Transactions on Robotics and Automation*, vol. 17, no. 3, pp. 229–241, Jun. 2001. DOI: 10.1109/70.938381.

[42] S. Thrun, W. Burgard and D. Fox, *Probabilistic Robotics* (Intelligent Robotics and Autonomous Agents). Cambridge, Mass: MIT Press, 2005, ISBN: 978-0-262-20162-9.

[43] S. Thrun, Y. Liu, D. Koller, A. Y. Ng, Z. Ghahramani and H. Durrant-Whyte, "Simultaneous Localization and Mapping with Sparse Extended Information Filters," *The International Journal of Robotics Research*, vol. 23, no. 7-8, pp. 693–716, Aug. 2004. DOI: 10.1177/0278364904045479.

[44] D. Fox, W. Burgard and S. Thrun, "Markov Localization for Reliable Robot Navigation and People Detection," in *Sensor Based Intelligent Robots*, G. Goos, J. Hartmanis, J. van Leeuwen, H. I. Christensen, H. Bunke and H. Noltemeier, Eds., vol. 1724, Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, pp. 1–20, ISBN: 978-3-540-66933-3. DOI: 10.1007/10705474_1.

[45] P. Del Moral, "Nonlinear filtering: Interacting particle resolution," *Comptes Rendus de l'Académie des Sciences - Series I - Mathematics*, vol. 325, no. 6, pp. 653–658, Sep. 1997. DOI: 10.1016/S0764-4442(97)84778-7.

[46] R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *Journal of Basic Engineering*, vol. 82, no. 1, p. 35, 1960. DOI: 10.1115/1.3662552.

[47] R. E. Kalman and R. S. Bucy, "New Results in Linear Filtering and Prediction Theory," *Journal of Basic Engineering*, vol. 83, no. 1, pp. 95–108, Mar. 1961. DOI: 10.1115/1.3658902.

[48] J. Willems, "Deterministic least squares filtering," *Journal of Econometrics*, vol. 118, no. 1-2, pp. 341–373, Jan. 2004. DOI: 10.1016/S0304-4076(03)00146-5.

[49] O. B. Hijab, "Minimum Energy Estimation," Ph.D. dissertation, University of California, Berkeley, United States – California, 1980, ISBN: 9798205262507. [Online]. Available: https://www.proquest.com/docview/303005105/abstract/37BACE22E23241FBPQ/1.

[50] L. A. McGee and S. F. Schmidt, "Discovery of the Kalman Filter as a Practical Tool for Aerospace and Industry," NASA, Technical Memorandum 86847, Nov. 1985, p. 25.

[51] S. F. Schmidt, "Application of State-Space Methods to Navigation Problems," in *Advances in Control Systems*, vol. 3, Elsevier, 1966, pp. 293–340, ISBN: 978-1-4831-6716-9. DOI: 10.1016/B978-1-4831-6716-9.50011-4.

[52] G. L. S. Smith, "Application of Statistical Filter Theory to the Optimal Estimation of Position and Velocity on Board a Circumlunar Vehicle," Tech. Rep. NASA-TR-R-135, Jan. 1962. [Online]. Available: https://ntrs.nasa.gov/search.jsp?R=20190002215 (visited on 09/01/2020).

[53] S. F. Schmidt, "The Kalman filter - Its recognition and development for aerospace applications," *Journal of Guidance and Control*, vol. 4, no. 1, pp. 4–7, Jan. 1981. DOI: 10.2514/3.19713.

[54] W. T. Higgins, "A Comparison of Complementary and Kalman Filtering," *IEEE Transactions on Aerospace and Electronic Systems*, vol. AES-11, no. 3, pp. 321–325, May 1975. DOI: 10.1109/TAES.1975.308081.

[55] J. K. Uhlmann, "Dynamic Map Building and Localization: New Theoretical Foundations," Ph.D. dissertation, University of Oxford, 1995.

[56] S. J. Julier and J. K. Uhlmann, "New extension of the Kalman filter to nonlinear systems," in *Signal Processing, Sensor Fusion, and Target Recognition VI*, vol. 3068, International Society for Optics and Photonics, Jul. 1997, pp. 182–193. DOI: 10.1117/12.280797.

[57] J. R. Carpenter and C. N. D'Souza, "Navigation Filter Best Practices," NASA, Technical Publication TP–2018–219822, Apr. 2018, p. 149. [Online]. Available: https://ntrs.nasa.gov/api/citations/20180003657/downloads/20180003657.pdf.

[58] J. L. Crassidis, F. L. Markley and Y. Cheng, "Survey of Nonlinear Attitude Estimation Methods," *Journal of Guidance, Control, and Dynamics*, vol. 30, no. 1, pp. 12–28, Jan. 2007. DOI: 10.2514/1.22452.

[59] E. Lefferts, F. Markley and M. Shuster, "Kalman Filtering for Spacecraft Attitude Estimation," *Journal of Guidance, Control, and Dynamics*, vol. 5, no. 5, pp. 417–429, Sep. 1982. DOI: 10.2514/3.56190.

[60] F. L. Markley, "Attitude Error Representations for Kalman Filtering," *Journal of Guidance, Control, and Dynamics*, vol. 26, no. 2, pp. 311–317, Mar. 2003. DOI: 10.2514/2.5048.

[61] F. L. Markley and J. L. Crassidis, *Fundamentals of Spacecraft Attitude Determination and Control* (Space Technology Library 33). New York: Springer, 2014, ISBN: 978-1-4939-0801-1.

[62] N. Filipe, M. Kontitsis and P. Tsiotras, "Extended Kalman Filter for Spacecraft Pose Estimation Using Dual Quaternions," *Journal of Guidance, Control, and Dynamics*, vol. 38, no. 9, pp. 1625–1641, 2015. DOI: 10.2514/1.G000977.

[63] P. Martin and E. Salaün, "Generalized Multiplicative Extended Kalman Filter for Aided Attitude and Heading Reference System," in *AIAA Guidance, Navigation, and Control Conference*, Toronto, Ontario, Canada: American Institute of Aeronautics and Astronautics, Aug. 2010. DOI: 10.2514/6.2010-8300.

[64] J. Solà, *Quaternion kinematics for the error-state Kalman filter*, Nov. 2017. arXiv: 1711.02508 [cs]. [Online]. Available: http://arxiv.org/abs/1711.02508.

[65] S. Bonnabel, P. Martin and E. Salaün, "Invariant Extended Kalman Filter: Theory and application to a velocity-aided attitude estimation problem," in *Proceedings of the 48th IEEE Conference on Decision and Control (CDC)*, Dec. 2009, pp. 1297–1304. DOI: 10.1109/CDC.2009.5400372.

[66] S. Bonnabel, "Left-invariant extended Kalman filter and attitude estimation," in *2007 46th IEEE Conference on Decision and Control*, New Orleans, LA, USA: IEEE, 2007, pp. 1027–1032. DOI: 10.1109/CDC.2007.4434662.

[67] S. Bonnabel, P. Martin and P. Rouchon, "Symmetry-Preserving Observers," *IEEE Transactions on Automatic Control*, vol. 53, no. 11, pp. 2514–2526, Dec. 2008. DOI: 10.1109/TAC.2008.2006929.

[68] M. Brossard, S. Bonnabel and A. Barrau, "Invariant Kalman Filtering for Visual Inertial SLAM," in *2018 21st International Conference on Information Fusion (FUSION)*, Jul. 2018, pp. 2021–2028. DOI: 10.23919/ICIF.2018.8455807.

[69] A. Barrau and S. Bonnabel, *An EKF-SLAM algorithm with consistency properties*, Sep. 2016. arXiv: 1510.06263 [cs]. [Online]. Available: http://arxiv.org/abs/1510.06263.

[70] A. Barrau and S. Bonnabel, "The Invariant Extended Kalman Filter as a Stable Observer," *IEEE Transactions on Automatic Control*, vol. 62, no. 4, pp. 1797–1812, Apr. 2017. DOI: 10.1109/TAC.2016.2594085.

[71] R. Mahony, T. Hamel and J.-M. Pflimlin, "Complementary filter design on the special orthogonal group SO(3)," in *Proceedings of the 44th IEEE Conference on Decision and Control*, Dec. 2005, pp. 1477–1484. DOI: 10.1109/CDC.2005.1582367.

[72] G. Baldwin, R. Mahony, J. Trumpf, T. Hamel and T. Cheviron, "Complementary filter design on the Special Euclidean group SE(3)," in *2007 European Control Conference (ECC)*, Kos: IEEE, Jul. 2007, pp. 3763–3770. DOI: 10.23919/ECC.2007.7068746.

[73] M. Zamani, "Deterministic Attitude and Pose Filtering, an Embedded Lie Groups Approach," Ph.D. dissertation, Australian National University, 2013.

[74] M. Zamani, J. Trumpf and R. Mahony, "Minimum-Energy Filtering for Attitude Estimation," *IEEE Transactions on Automatic Control*, vol. 58, no. 11, pp. 2917–2921, Nov. 2013. DOI: 10.1109/TAC.2013.2259092.

[75] M. Zamani and J. Trumpf, "Discrete update pose filter on the special Euclidean group SE(3)," in *Proceedings of the 55th IEEE Conference on Decision and Control (CDC)*, 2019.

[76] J. Berger, A. Neufeld, F. Becker, F. Lenzen and C. Schnörr, "Second Order Minimum Energy Filtering on SE(3) with Nonlinear Measurement Equations," in *Scale Space and Variational Methods in Computer Vision*, vol. 9087, Cham: Springer International Publishing, 2015, pp. 397–409.

[77] A. Saccon, J. Trumpf, R. Mahony and A. P. Aguiar, "Second-Order-Optimal Minimum-Energy Filters on Lie Groups," *IEEE Transactions on Automatic Control*, vol. 61, no. 10, pp. 2906–2919, Oct. 2016. DOI: 10.1109/TAC.2015.2506662.

[78] D. Fox, W. Burgard, H. Kruppa and S. Thrun, "A Probabilistic Approach to Collaborative Multi-Robot Localization," *Autonomous Robots*, vol. 8, no. 3, pp. 325–344, Jun. 2000. DOI: 10.1023/A:1008937911390.

[79] S. Roumeliotis and G. Bekey, "Distributed multirobot localization," *IEEE Transactions on Robotics and Automation*, vol. 18, no. 5, pp. 781–795, Oct. 2002. DOI: 10.1109/TRA.2002.803461.

[80] J. Zhu and S. S. Kia, "Cooperative Localization Under Limited Connectivity," *IEEE Transactions on Robotics*, vol. 35, no. 6, pp. 1523–1530, Dec. 2019. DOI: 10.1109/TRO.2019.2930404.

[81] A. Bahr, M. R. Walter and J. J. Leonard, "Consistent cooperative localization," in *2009 IEEE International Conference on Robotics and Automation*, May 2009, pp. 3415–3422. DOI: 10.1109/ROBOT.2009.5152859.

[82] J. Aspnes, T. Eren, D. Goldenberg *et al.*, "A Theory of Network Localization," *IEEE Transactions on Mobile Computing*, vol. 5, no. 12, pp. 1663–1678, Dec. 2006. DOI: 10.1109/TMC.2006.174.

[83] S. Julier and J. Uhlmann, "General Decentralized Data Fusion with Covariance Intersection (CI)," in *Handbook of Multisensor Data Fusion: Theory and Practice*, 2nd ed, Boca Raton, FL: CRC Press, 2009, ISBN: 978-1-4200-5309-8.

[84] V. Crespi, A. Galstyan and K. Lerman, "Top-down vs bottom-up methodologies in multi-agent system design," *Autonomous Robots*, vol. 24, no. 3, pp. 303–313, Apr. 2008. DOI: 10.1007/s10514-007-9080-5.

[85] A. Howard, M. Mataric and G. Sukhatme, "Putting the 'I' in 'team': An ego-centric approach to cooperative localization," in *2003 IEEE International Conference on Robotics and Automation*, Sep. 2003, pp. 868–874. DOI: 10.1109/ROBOT.2003.1241702.

[86] S. J. Julier and J. K. Uhlmann, "A non-divergent estimation algorithm in the presence of unknown correlations," in *Proceedings of the 1997 American Control Conference (Cat. No.97CH36041)*, vol. 4, Jun. 1997, 2369–2373 vol.4. DOI: 10.1109/ACC.1997.609105.

[87] L. C. Carrillo-Arce, E. D. Nerurkar, J. L. Gordillo and S. I. Roumeliotis, "Decentralized multi-robot cooperative localization using covariance intersection," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Nov. 2013, pp. 1412–1417. DOI: 10.1109/IROS.2013.6696534.

[88] M. Zamani and R. Hunjet, "Collaborative Pose Filtering Using Relative Measurements and Communications," in *2019 12th Asian Control Conference (ASCC)*, Jun. 2019, pp. 919–924.

[89] H. Li, F. Nashashibi and M. Yang, "Split Covariance Intersection Filter: Theory and Its Application to Vehicle Localization," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 4, pp. 1860–1871, Dec. 2013. DOI: 10.1109/TITS.2013.2267800.

[90] T. R. Wanasinghe, G. K. Mann and R. G. Gosine, "Decentralized Cooperative Localization for Heterogeneous Multi-robot System Using Split Covariance Intersection Filter," in *2014 Canadian Conference on Computer and Robot Vision*, May 2014, pp. 167–174. DOI: 10.1109/CRV.2014.30.

[91] S. Roumeliotis and G. Bekey, "Collective localization: A distributed Kalman filter approach to localization of groups of mobile robots," in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, vol. 3, Apr. 2000, 2958–2965 vol.3. DOI: 10.1109/ROBOT.2000.846477.

[92] A. Basu, J. Gao, J. S. B. Mitchell and G. Sabhnani, "Distributed localization using noisy distance and angle information," in *Proceedings of the Seventh ACM International Symposium on Mobile Ad Hoc Networking and Computing - MobiHoc '06*, Florence, Italy: ACM Press, 2006, p. 262. DOI: 10.1145/1132905.1132934.

[93] Y. Dieudonne, O. Labbani-Igbida and F. Petit, "On the solvability of the localization problem in robot networks," in *2008 IEEE International Conference on Robotics and Automation*, May 2008, pp. 480–485. DOI: 10.1109/ROBOT.2008.4543253.

[94] A. Martinelli and R. Siegwart, "Observability analysis for mobile robot localization," in *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Aug. 2005, pp. 1471–1476. DOI: 10.1109/IROS.2005.1545153.

[95] A. Mourikis and S. Roumeliotis, "Performance analysis of multirobot Cooperative localization," *IEEE Transactions on Robotics*, vol. 22, no. 4, pp. 666–681, Aug. 2006. DOI: 10.1109/TRO.2006.878957.

[96] S. S. Kia, S. Rounds and S. Martinez, "Cooperative Localization for Mobile Agents: A Recursive Decentralized Algorithm Based on Kalman-Filter Decoupling," *IEEE Control Systems Magazine*, vol. 36, no. 2, pp. 86–101, Apr. 2016. DOI: 10.1109/MCS.2015.2512033.

[97] K. Y. K. Leung, T. D. Barfoot and H. H. T. Liu, "Decentralized Localization of Sparsely-Communicating Robot Networks: A Centralized-Equivalent Approach," *IEEE Transactions on Robotics*, vol. 26, no. 1, pp. 62–77, Feb. 2010. DOI: 10.1109/TRO.2009.2035741.

[98] L. Luft, T. Schubert, S. I. Roumeliotis and W. Burgard, "Recursive decentralized localization for multi-robot systems with asynchronous pairwise communication," *The International Journal of Robotics Research*, vol. 37, no. 10, pp. 1152–1167, Sep. 2018. DOI: 10.1177/0278364918760698.

[99] K. M. Brink, "Partial-Update Schmidt–Kalman Filter," *Journal of Guidance, Control, and Dynamics*, vol. 40, no. 9, pp. 2214–2228, Sep. 2017. DOI: 10.2514/1.G002808.

[100] R. Jung, C. Brommer and S. Weiss, "Decentralized Collaborative State Estimation for Aided Inertial Navigation," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, May 2020, pp. 4673–4679. DOI: 10.1109/ICRA40945.2020.9197178.

[101] K. Y. Leung, Y. Halpern, T. D. Barfoot and H. H. Liu, "The UTIAS multi-robot cooperative localization and mapping dataset," *The International Journal of Robotics Research*, vol. 30, no. 8, pp. 969–974, Jul. 2011. DOI: 10.1177/0278364911398404.

[102] N. Sullivan, S. Grainger and B. Cazzolato, "Analysis of cooperative localisation performance under varying sensor qualities and communication rates," *Robotics and Autonomous Systems*, vol. 110, pp. 73–84, Dec. 2018. DOI: 10.1016/j.robot.2018.09.010.

[103] J. Hartzer and S. Saripalli, "Vehicular Teamwork: Collaborative localization of Autonomous Vehicles," in *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, Sep. 2021, pp. 1077–1082. DOI: 10.1109/ITSC48978.2021.9564981.

[104] S. Güler, M. Abdelkader and J. S. Shamma, "Infrastructure-free Multi-robot Localization with Ultrawideband Sensors," in *2019 American Control Conference (ACC)*, Jul. 2019, pp. 13–18. DOI: 10.23919/ACC.2019.8814678.

[105] M. Burri, J. Nikolic, P. Gohl *et al.*, "The EuRoC micro aerial vehicle datasets," *The International Journal of Robotics Research*, vol. 35, no. 10, pp. 1157–1163, Sep. 2016. DOI: 10.1177/0278364915620033.

[106] D. Schubert, T. Goll, N. Demmel, V. Usenko, J. Stückler and D. Cremers, "The TUM VI Benchmark for Evaluating Visual-Inertial Odometry," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct. 2018, pp. 1680–1687. DOI: 10.1109/IROS.2018.8593419.

[107] A. Chakraborty, C. N. Taylor, R. Sharma and K. M. Brink, "Cooperative localization for fixed wing unmanned aerial vehicles," in *2016 IEEE/ION Position, Location and Navigation Symposium (PLANS)*, Apr. 2016, pp. 106–117. DOI: 10.1109/PLANS.2016.7479689.

[108] T. Baca, M. Petrlik, M. Vrba *et al.*, "The MRS UAV System: Pushing the Frontiers of Reproducible Research, Real-world Deployment, and Education with Autonomous Unmanned Aerial Vehicles," *Journal of Intelligent & Robotic Systems*, vol. 102, no. 1, p. 26, Apr. 2021. DOI: 10.1007/s10846-021-01383-5.

[109] M. Vrba, D. Heřt and M. Saska, "Onboard Marker-Less Detection and Localization of Non-Cooperating Drones for Their Safe Interception by an Autonomous Aerial System," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3402–3409, Oct. 2019. DOI: 10.1109/LRA.2019.2927130.

[110] T. Krajník, M. Nitsche, J. Faigl *et al.*, "A Practical Multirobot Localization System," *Journal of Intelligent & Robotic Systems*, vol. 76, no. 3-4, pp. 539–562, Dec. 2014. DOI: 10.1007/s10846-014-0041-x.

[111] M. Fiala, "ARTag, a fiducial marker system using digital techniques," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 2, Jun. 2005, 590–596 vol. 2. DOI: 10.1109/CVPR.2005.74.

[112] N. Sullivan, "Task Allocation and Collaborative Localisation in Multi-Robot Systems," Ph.D. dissertation, University of Adelaide, Jul. 2019. [Online]. Available: https://hdl.handle.net/2440/120578.

[113] E. Olson, "AprilTag: A robust and flexible visual fiducial system," in *2011 IEEE International Conference on Robotics and Automation*, Shanghai, China: IEEE, May 2011, pp. 3400–3407. DOI: 10.1109/ICRA.2011.5979561.

[114] V. Walter, N. Staub, M. Saska and A. Franchi, "Mutual Localization of UAVs based on Blinking Ultraviolet Markers and 3D Time-Position Hough Transform," in *2018 IEEE 14th International Conference on Automation Science and Engineering (CASE)*, Munich: IEEE, Aug. 2018, pp. 298–303. DOI: 10.1109/COASE.2018.8560384.

[115] V. Walter, M. Saska and A. Franchi, "Fast Mutual Relative Localization of UAVs using Ultraviolet LED Markers," in *2018 International Conference on Unmanned Aircraft Systems (ICUAS)*, Dallas, TX: IEEE, Jun. 2018, pp. 1217–1226. DOI: 10.1109/ICUAS.2018.8453331.

[116] N. Saeed, S. Guo, K.-H. Park, T. Y. Al-Naffouri and M.-S. Alouini, "Optical camera communications: Survey, use cases, challenges, and future trends," *Physical Communication*, vol. 37, p. 100 900, Dec. 2019. DOI: 10.1016/j.phycom.2019.100900.

[117] M. Ridolfi, A. Kaya, R. Berkvens, M. Weyn, W. Joseph and E. D. Poorter, "Self-calibration and Collaborative Localization for UWB Positioning Systems: A Survey and Future Research Directions," *ACM Computing Surveys*, vol. 54, no. 4, pp. 1–27, May 2022. DOI: 10.1145/3448303.

[118] A. Barrau, "Non-linear state error based extended Kalman filters with applications to navigation," Theses, Mines Paristech, Sep. 2015. [Online]. Available: https://hal.archives-ouvertes.fr/tel-01247723 (visited on 27/09/2019).

[119] J. W. Polderman and J. C. Willems, *Introduction to Mathematical Systems Theory: A Behavioral Approach* (Texts in Applied Mathematics 26). New York: Springer, 1998, ISBN: 978-0-387-98266-3.

[120] P.-A. Absil, R. Mahony and R. Sepulchre, *Optimization Algorithms on Matrix Manifolds*. Princeton, N.J. ; Woodstock: Princeton University Press, 2008, ISBN: 978-0-691-13298-3.

[121] R. Ramalingam, G. Anitha and J. Shanmugam, "Microelectromechnical Systems Inertial Measurement Unit Error Modelling and Error Analysis for Low-cost Strapdown Inertial Navigation System," *Defence Science Journal*, vol. 59, no. 6, pp. 650–658, Nov. 2009. DOI: 10.14429/dsj.59.1571.

[122] ublox, "ZED-F9P Module," Product Summary UBX-17005151 - R13, 2023. [Online]. Available: https://content.u-blox.com/sites/default/files/ZED-F9P_ProductSummary_UBX-17005151.pdf (visited on 20/04/2023).

[123] W. Henning, "User Guidelines for Single Base Real Time GNSS Positioning," National Geodetic Survey, National Oceanic and Atmospheric Administration, Tech. Rep., Aug. 2011. [Online]. Available: https://www.ngs.noaa.gov/PUBS_LIB/NGSRealTimeUserGuidelines.v2.1.pdf (visited on 20/04/2023).

[124] Intel Corporation, "USB 3.0* Radio Frequency Interference Impact on 2.4GHz Wireless Devices - White Paper," White Paper 327216-001, Apr. 2012.

[125] FLIR Integrated Imaging Solutions Inc., "FLIR Firefly Technical Reference for FFY-U3-04S2M," Technical Reference, May 2020.

[126] Z. Zivkovic and O. Booij, "How did we built our hyperbolic mirror omnidirectional camera - practical issues and basic geometry," University of Amsterdam, Technical Report IAS-UV A-05-04, 2005.

[127] National Renewable Energy Laboratory, *SMARTS MODTRAN ETR Spectra*. [Online]. Available: https://www.nrel.gov/grid/solar-resource/spectra.html (visited on 15/12/2022).

[128] O. Kreylos, *Hacking the Oculus Rift DK2*, Oct. 2014. [Online]. Available: http://doc-ok.org/?p=1095 (visited on 10/05/2020).

[129] R. C. Bose and D. K. Ray-Chaudhuri, "On a class of error correcting binary group codes," *Information and Control*, vol. 3, no. 1, pp. 68–79, Mar. 1960. DOI: 10.1016/S0019-9958(60)90287-4.

[130] U. Bauer, "Ripser: Efficient computation of Vietoris–Rips persistence barcodes," *Journal of Applied and Computational Topology*, vol. 5, no. 3, pp. 391–423, Sep. 2021. DOI: 10.1007/s41468-021-00071-5.

[131] J. Kannala and S. Brandt, "A generic camera model and calibration method for conventional, wide-angle, and fish-eye lenses," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 8, pp. 1335–1340, Aug. 2006. DOI: 10.1109/TPAMI.2006.153.

[132] V. Usenko, N. Demmel and D. Cremers, "The Double Sphere Camera Model," in *2018 International Conference on 3D Vision (3DV)*, Sep. 2018, pp. 552–560. DOI: 10.1109/3DV.2018.00069.

[133] Y. Lochman, K. Liepieshov, J. Chen, M. Perdoch, C. Zach and J. Pritts, "BabelCalib: A Universal Approach to Calibrating Central Cameras," in *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, Montreal, QC, Canada: IEEE, Oct. 2021, pp. 15 233–15 242. DOI: 10.1109/ICCV48922.2021.01497.

[134] Decawave, *Product Datasheet: DWM1001-DEV*, Dec. 2021. [Online]. Available: https://www.qorvo.com/products/d/da007952.

[135] A. S. Tanenbaum, *Computer Networks*, 4th ed. Upper Saddle River, NJ: Prentice Hall PTR, 2003, ISBN: 978-0-13-066102-9.

[136] M. Hillam, "Improving Ad Hoc Ultra-Wideband-Radio-Based Ranging for Unmanned Ground Vehicles," University of New South Wales Canberra, Tech. Rep., 2021, p. 11.

[137] Decawave, "Non Line of Sight Operation and Optimizations to Improve Performance in DW1000 Based Systems," Application Note APS006 Part 2, 2014, p. 31.

[138] Decawave, "Sources of Error in DW1000 Based Two-Way Ranging (TWR) Schemes," Application Note APS011, 2014, p. 22.

[139] K. Nomizu, "Invariant Affine Connections on Homogeneous Spaces," *American Journal of Mathematics*, vol. 76, no. 1, p. 33, Jan. 1954. DOI: 10.2307/2372398. JSTOR: 2372398.

[140] A. Barrau and S. Bonnabel, "Invariant particle filtering with application to localization," in *53rd IEEE Conference on Decision and Control*, Dec. 2014, pp. 5599–5605. DOI: 10.1109/CDC.2014.7040265.

[141] Y. Bar-Shalom, X.-R. Li and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation*. New York: Wiley, 2001, ISBN: 978-0-471-41655-5.

[142] D. Woodbury and J. Junkins, "On the Consider Kalman Filter," in *AIAA Guidance, Navigation, and Control Conference*, Toronto, Ontario, Canada: American Institute of Aeronautics and Astronautics, Aug. 2010. DOI: 10.2514/6.2010-7752.

[143] E. D. Sontag, *Mathematical Control Theory* (Texts in Applied Mathematics), J. E. Marsden, L. Sirovich, M. Golubitsky and W. Jäger, Eds. New York, NY: Springer New York, 1998, vol. 6, ISBN: 978-1-4612-6825-3. DOI: 10.1007/978-1-4612-0577-7.

[144] D. E. Kirk, *Optimal Control Theory: An Introduction.* Dover Publications, 2012, ISBN: 978-1-306-34995-6.

Cover page artwork by rawpixel.com