Exploiting landing in multi-UAV systems for enhanced cooperative localization

A Gaussian Belief Propagation Perspective

S.H. Molenkamp

Delft University of Technology

TU Delft

Exploiting landing in multi-UAV systems for enhanced cooperative localization

A Gaussian Belief Propagation Perspective

by

S.H. Molenkamp

to obtain the degree of Master of Science

at the Delft University of Technology,

to be defended publicly on Thursday June 4th, 2026 at 16:00 hours.

Project duration:	September 1, 2025 – Jun 4, 2026	
Thesis committee:	Prof. dr. K.G. Langendoen,	TU Delft, supervisor
	Dr. ir. R. T. Rajan,	TU Delft
	Ir. P.S. de Vries	External Member, TNO
	Ir. G.B.G. Potter	External Member, TNO

Cover: Generated by Ideogram AI
Style: TU Delft Report Style, with modifications by Daan Zwaneveld

An electronic version of this thesis is available at http://repository.tudelft.nl/.

TU Delft TNO

Abstract

Multi-agent Systems increasingly rely on cooperative localisation to navigate GNSS-denied environments using only relative measurements. However, these purely relative networks inevitably accumulate global trajectory drift over time. This thesis investigates whether strategically landing a UAV to serve as a stationary anchor can mitigate this accumulated drift and improve the absolute localization performance of the overarching MAS.

Unlike traditional filtering methods that only estimate the current state, this work adopts Gaussian Belief Propagation to achieve scalable, fully distributed estimation, solving factor graphs locally to jointly optimize the trajectory history. Within this distributed architecture, low-cost IMU and range-bearing constraints were simulated to evaluate various landed anchor mathematical representations (Unary, Persistent Variable, and ZUPT) across continuous and multi-group flight topologies. The results demonstrate that in single-group, continuous flight scenarios, landing a drone fails to bound global drift because the dense network topology becomes highly overconfident and rigidly locks the anchor into a drifted state. Conversely, when applied to multi-group deployments, representing the landed anchor with a Zero-Velocity Update (ZUPT) model mathematically combats this overconfidence through artificial covariance inflation, allowing the anchor to absorb geometric corrections and successfully reset the drift of subsequent passing groups.

The findings imply that autonomous MAS should not simply deploy stationary anchors during continuous, dense flight without risking algorithmic divergence; instead, operators must explicitly structure missions into temporal, multi-wave batches to safely exploit this relative infrastructure. While the baseline GBP architecture was verified using real-world datasets, the conclusions regarding landed anchors rely on synthetic evaluations within a 2D simulation, meaning the proposed system must still be validated against the physical hardware and 3D flight complexities of real-world deployments.

Preface

First and foremost I thank my daily supervisors. Pieter was always enthusiastic and ready to help, and showed unwaivering positive reinforcement, even sitting down next to me to figure things out when I got lost. He always had ideas about what next steps could be taken and kept me inspired through tough times. I want to thank George for keeping me sharp on details, on setting intermediate deadlines and (trying) to keep to them, and always responding super quickly with feedback on my work. I thank him for his general positive outlook.

Next, I would like to thank my TU supervisor Koen, for useful discussions and especially for his emphasis on precision in every word or figure, that improved the quality of my writing greatly.

I would like to thank all the other colleagues at TNO as well, as they gave me a warm welcome. TNO is the ideally suited environment to pose hard problems or engineering issues during coffee or lunch, and I had many interesting discussions during my time there, even though many times unrelated to my work. At the start of my project, I had talks with a few colleagues, who guided me on what was possible and achievable and what would not be. I thank in particular Gabriele, for his specific guidance on factor graphs and his expertise in sensor fusion and kalman filtering, as talks with him greatly aided my understanding.

Finally, I thank my family, for all the support throughout this academic journey. My mom, dad and both my sisters were always ready to have a discussion, proofread my work, or simply listen. I also want to give special thanks to Kim, for guiding me through the daily struggle during my thesis, always pushing me to try and do just a little better and put in that last hour when it's necessary. I couldn't have done it without you.

S.H. Molenkamp
Delft, May 2026

Contents

Nomenclature

Abbreviations

Abbreviation	Definition
ADMM	Alternating Direction Method of Multipliers
BLE	Bluetooth Low Energy
BP	Belief Propagation
CCL	Centralized Cooperative Localization
CI	Covariance Intersection
CL	Cooperative Localization
DCL	Distributed Cooperative Localization
DR	Dead Reckoning
EKF	Extended Kalman Filter
FG	Factor Graph
FGO	Factor Graph Optimization
GBP	Gaussian Belief Propagation
GNSS	Global Navigation Satellite Systems
IMU	Inertial Measurement Unit
INS	Inertial Navigation System
KF	Kalman Filter
LoS	Line of Sight
LIDAR	Light Detection and Ranging
MAS	Multi-Agent Systems
MSCKF	Multi-state Constraint Kalman Filter
NBP	Non-parametric Belief Propagation
PF	Particle Filter
PGO	Pose Graph Optimization
RADAR	Radio Detection and Ranging
SDP	Semi-Definite Programming
SLAM	Simultaneous Localization And Mapping
SLR	Statistical Linear Regression
SPBP	Sigma-Point Belief Propagation
UAV	Unmanned Aerial Vehicle
UKF	Unscented Kalman Filter
UWB	Ultra-Wide Band

1

Introduction

Accurate localization is essential to the effective operation of cooperative missions by multi-agent systems (MAS), in particular for unmanned aerial vehicles (UAVs)[1]. Knowing your location relative to the world and other agents lays the fundament for all other tasks in MAS: perception, mapping, planning & control. Nowadays, as drone technology is maturing, MAS are deployed increasingly in complex and adverse scenarios, such as environmental monitoring, exploration, search and rescue or military operations [2], [3], [4], [5]. However, the typical environment for military operations contains little to no signal from the Global Navigation Satellite Systems (GNSS). The GNSS signal is compromised by multipath effects, or entirely lost due to obstruction in urban canyon or forest canopy [1]. Even more so, electromagnetic shielding, signal jamming and spoofing attacks, [6] all induce GNSS failure. Jamming of GNSS happens on a widespread level in present-day conflicts, spreading its effect far beyond the borders of the conflict [7]. All these factors necessitate localization and positioning based on on-board equipment.

There is an extensive body of research on the on-board localization of single agents [8]. However, when assuming adversarial conditions, especially signal jamming and spoofing, solutions must rely on relative positioning solutions. Relative solutions determine location in relation to a reference location in a local coordinate system. GNSS-denied solutions such as Wi-Fi, 5G, BLE or UWB are susceptible to interference from external factors [8]. Even with the rapid advance of computer vision, and consequently camera-based localization [9], [10], these methods still struggle in feature-poor environments, such as those that occur in or near battlefield situations. In these areas, there is no guarantee on lasting presence of features such as buildings, trees or even open plains as the battlefield is changing on an hour to hour basis. Therefore, relative positioning solutions rely on Inertial Navigation Systems (INS), Simultaneous Localization and Mapping (SLAM) type odometers and Dead Reckoning (DR) based techniques leading to errors drifting unboundedly over time [11].

To counter the unbounded drift, the topic of Cooperative Localization (CL) has attracted much attention [12]. CL has proven to improve positioning accuracy and availability between multiple agents [13]. The idea of CL was introduced more than three decades ago [14] and has seen significant progress since then. CL relies on relative measurements, measuring range only, bearing only, range-bearing and full relative transformation between agents. Information is then exchanged between agents through communication and data fusion is performed [13].

CL can be divided roughly into two methods, depending on the computation happening at a central point, making it Centralized Cooperative Localization (CCL), or at each agent locally, making it Distributed Cooperative Localization (DCL). The advantage of CCL is that since the inference takes place at a central point, that point carries all available information and can therefore make the optimal overall prediction. However, CCL has severe limitations, being a single-point-of-failure architecture and incurring high communication bandwidth requirements and high computation costs. This limits the scalability of CCL and makes it unfit for larger scale MAS.

This lack of scalability is particularly problematic because the current operational trend is towards mul-

titudes of low-cost units with limited sensors and accuracy [15]. This trend is replacing the single high-precision units, especially in military applications, which substantiates the need for a more scalable alternative to traditional centralized high-compute methods. Furthermore, the large number of units, combined with limited communication and sensing range, also calls for a more dynamic structure, where agents can enter and leave the cooperative network at any time. The CL in this scenario induces the need for an estimation method that has no single point of failure, is robust enough to rely only on localized communication, and can back-propagate updates when communication is intermittent/unreliable.

DCL, as opposed to CCL, refers to a distributed architecture. In DCL, each unit keeps its own, limited view of the MAS, determined by local connections, and information is propagated through the network hop by hop. Therefore, DCL has no single point of failure [16], communication is often local and thus limited and computation is distributed over all units in the MAS, providing robustness and scalability.

1.1. Problem definition

This thesis concerns the scenario where a relatively long-range flight (10km) needs to be undertaken with a MAS of low-cost UAVs (with limited sensor performance) under adversarial conditions, as visualised in Figure 1.1. The MAS will use distributed cooperative localization (DCL) to reach the end location with the highest possible localization precision for each of the units in the MAS.

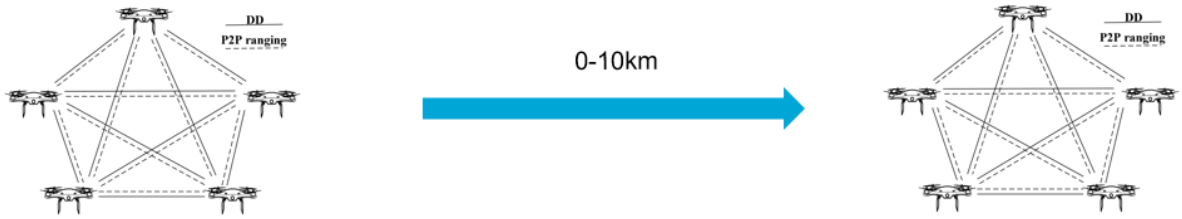


Figure 1.1: Schematic view of operational scenario. Flying a group of drones, in formation, over a distance of 10km. The star formation is for visualisation and not necessarily the formation used.

However, using DCL for reliable high-accuracy localization in mobile UAV MAS introduces challenges. Specifically, performance is critically determined by the network's lack of access to absolute reference information, or *anchors* [17], and its ability to handle heterogeneous agent capabilities and state complexities inherent in 2D/3D aerial navigation.
Therefore, there is research to be done on the impact of anchors: How the existence, deployment and use of absolute reference nodes, or rather, the lack of them, forcing anchor-free operation, constrain achievable localization performance [1].
The core subject requiring dedicated investigation is the possibility of using the moving units as mobile anchors. Specifically, by landing UAVs out of the MAS, making them physically stationary, they could be used as anchors, as detailed in Figure 1.2. Even more so focusing on the operational scenario with low-cost UAVs, there is limited literature, or literature does make use of absolute reference such as GPS [18].

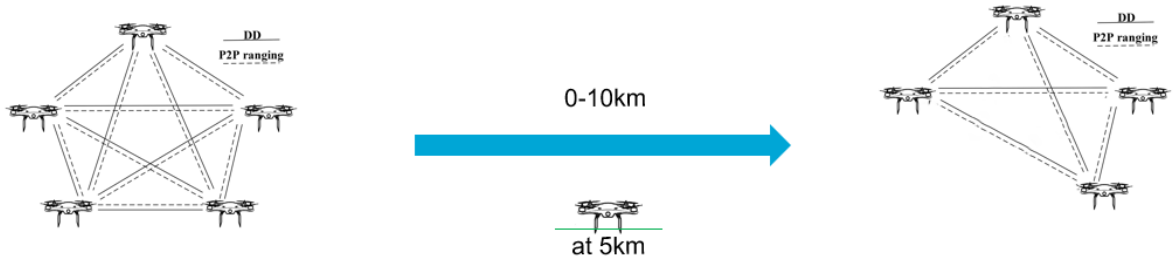


Figure 1.2: Schematic view of operational scenario with proposed solution with landed drone as anchor. Flying a group of drones, in formation, landing one (or more) over a distance of 10km. Formation and distance at which landed are purely for visualisation.

The operational relevance of figuring out how to improve MAS localization, over individual localization, comes from the potential of increased effectivity of the overall system. The increase in system effectivity rests on the principle of averaging out noise/error over multiple units. Next to that, sacrificing units by landing them, and increasing localization precision in this way through knowing that the unit is stationary, could further increase effectivity. If normally only half would make it to the target individually, but one could increase the location estimation accuracy by sacrificing UAVs, after which 75% makes it to the target, the trade-off would be worthwile. Furthermore, putting a statistically sound number on how many drones with which sensor sets, payloads and abilities one needs to execute exploration mission X or defense mission Y is more relevant than ever for the military, because of the developments and trends in defense and warfare, both in Ukraine and the entire European landscape.

1.2. Research goal

To the best of our knowledge, no prior research has addressed the question of whether landing a unit out of the MAS can improve the location estimate of the remainder of the MAS. Although historically the idea of creating stationary anchors forms the foundation for CL literature [14], recent works have focused on this idea in the context of sensing heterogeneity between units, e.g. using aerial-ground collaboration [1], [19], or using mobile anchors with improved sensor quality [17]. Next to that there are those using only relative measurements, but with no attempt to create anchors at any point [16]. The idea of sacrificing part of one's system to more consistently reach the end goal, and thus produce a more efficient yield, has not been investigated. Although cooperative localization with the use of units in MAS as mobile anchors is investigated to some extent [17], [1], the specific scenario of landing a unit out of the MAS is a new venture. The objective is to consistently improve the final location estimate by landing units out of the MAS along the flight/journey.

This work will investigate the landing phenomenon by analyzing specialized distributed optimization formulations, focusing on how external positioning references and internal variation among agents impact the convergence, accuracy, and efficiency of distributed multi-UAV localization.

1.3. Research questions

Main RQ: To what extent does landing drones in a MAS improve absolute localization performance when performing cooperative localization, under low-cost IMU and relative range and bearing measurements?

In order to properly answer this main research question, the following subquestions need to be addressed:

- What representation and algorithm can be used to model the MAS dynamics and measurements, and why is this choice appropriate for the problem and sensors considered?
- What is the improvement in absolute localization accuracy as the number of landed drones (N) increases, and at what point do these performance gains diminish?
- Given specific mission constraints (MAS size, sensor noise levels, and graph connectivity), what is the minimum number of landed drones required to guarantee a maximum allowable Final Drift threshold?

1.4. Thesis outline

The rest of this thesis is structured into six chapters. Chapter 2 provides an overview of previous work related to CL, especially DCL, and the role of (mobile) anchors in CL. Chapter 3 describes the mathematical notation and prerequisites to understand the algorithm used in this thesis. Chapter 4 dives into the specifics of the operational scenario this thesis is concerned with, and the models used to describe the scenario. Chapter 5 describes the simulation environment and tools and values used to evaluate the methods, Chapter 6 provides all the results of the different models investigated. Chapter 7 provides a more in-depth discussion of the presented results. Finally, Chapter 8 concludes the work with conclusions to the research questions and insights into future improvements.

2

Related work

Distributed cooperative location estimation in multi-agent systems is an active research field, presenting no definitive best practice as of yet. Special attention is being paid to significant real life challenges such as communication constraints [20], [21], low-cost platforms, thus using only partial relative pose measurements and measuring with reduced accuracy [15], and consequently heterogeneity [1]. This chapter will first define some core concepts and terminology in navigation/localization in Section 2.1, and then dive into the history of CL in Section 2.2. Next, in Section 2.3 the state of the art distributed algorithms for CL will be discussed. Afterwards, there is an elaboration on the sensors used in CL in Section 2.4, then the role of anchor points is discussed in Section 2.5. To close off, Section 2.6 gives concluding remarks on the presented literature.

2.1. Navigation/localization preliminaries

Before detailing the history and state-of-the-art of cooperative localization, it is necessary to establish the foundational concepts of spatial navigation and localization. This section introduces some of the core terms used in navigation, marked in bold, starting from a scenario with autonomous agents.

An autonomous agent fundamentally perceives the world through its on-board sensors in its **local frame**, the coordinate system attached to its body that translates and rotates with the agent. To operate effectively within a wider environment, or cooperatively, this local data must mathematically be transformed into the **global frame** - a fixed, stationary coordinate system representing the absolute map.

The core objective of navigation/localization is estimating the agent's **pose**, which represents the state of an agent at a given time. The **pose** is typically its 2D or 3D position, alongside its orientation (e.g. $x_i = [\theta, x, y]$ in 2D). Because calculating heading (2D) or orientation (3D) inherently involves trigonometric functions, which are non-linear functions, navigating in the real world is fundamentally a **non-linear optimization problem**.

To estimate the pose continuously, systems traditionally rely on **dead-reckoning** (or **odometry**). **Dead-reckoning** is the process of integrating local sensor data over time to estimate the current position from a known starting point. However, because integration inherently accumulates small measurement errors into unbounded drift, modern systems rely on **sensor fusion**, algorithmically combining data from multiple sensors to create a more reliable and accurate estimate of the state.

To correct the drift caused by dead-reckoning, localization systems also rely on spatial constraints to bound the error. In some graph-based models, these constraints are generated via **loop closures**, instances where an agent recognizes a previously visited state, mathematically tightening its estimate. In feature-poor or adversarial environments, systems must rely on **anchors**, specific nodes or agents within the network that possess highly certain state information, either through absolute global frame sensors (e.g. GPS) or by being physically stationary, effectively grounding the relative network to reality.

2.2. History of cooperative relative localization

The idea of cooperative localization was first proposed in 1994 by Kurazume *et al.* [14], stemming from the problem of absolute positioning of multiple robots. The traditional method for robot localization without external help, which is still the method used today, is that given their initial positions in a common reference frame, the next positions are obtained using an on-board system for dead reckoning, most commonly odometry for ground vehicles, or IMU/INS for aerial vehicles.

However, positions calculated in this way accumulate errors over longer times and distances as the onboard sensors bring noise and bias issues, especially in low-cost platforms. In order to mitigate the accumulated errors without external facilities such as GNSS and fixed beacons, [14] proposed for the first time the idea of cooperative localization where the mobile robots are treated as beacons themselves. In their idea there are two groups of robots that alternate movement, while using the other stationary group as landmarks (anchors), thus providing location reference. In this way an increased number of robots also increases the redundancy of position information. Then, using the weighted least squares method, the redundancy leads to a decrease in localization error.

Although this pioneered idea by [14] inspired the research into infrastructure-free localization, it lacks concrete experiments and sensing device configurations. The terminology of cooperative localization was only first introduced later, in 2000, by Rekleitis *et al.* [22], who also added real world experiments to the picture to validate that the idea can indeed reduce the inaccuracies produced by accumulated dead reckoning errors. They propose a method where robots visually observe each other, with one robot carrying a camera and the other a marker, to improve localization. They propose trapezoidization and triangulation for large and small-scale spaces, respectively.

However, this early work is still limited as it assumes that the robots can always see each other and the landmark robots are stationary. This imposes extra constraints:

- Motion planning is constrained by one of the groups having to be stationary to act as landmark.
- Always seeing each other prevents the capability to work in obstacle-filled spaces.
- Always seeing each other also prevents scalability to scenarios with more robots where observability is intermittent (due to sensing range constraints) or dynamic (with new robots coming in at later times)

Another limitation of the work is that it assumes a measurement of full relative pose between different robots is available. It does not take into account limited relative measurements imposed by sensing devices. That is, most sensors measure range-only, bearing-only, or range-bearing, especially in low-cost platforms [12]. Today, measuring the full relative pose is becoming increasingly possible with the use of commodity cameras combined with advances in computer vision and spatial AI [23], [9], [10]. However, this is not a definitive solution yet, presenting challenges in feature-poor environments and real-time processing speeds.

In 2002, Roumeliotis *et al.* [24] improve on the motion planning constraint by localizing a group of robots that are all mobile. They discuss the centralized way to do this, performing sensor fusion with one Kalman filter that processes the available positioning information from all team members and then produces a pose estimate for all. They then go on to show that the equations for this can be rewritten in a decentralized manner, allowing for the decentralization of computation to a number of smaller communicating filters. Information is only exchanged when robots are within each others' range and measure relative pose.
As observing a relative pose is not often feasible in real world scenarios, the state-of-the-art field addresses the relative pose constrained and relaxes it to range-only, bearing-only and range-bearing based solutions [12].
The always-seeing-each-other constraint is continuously combatted by the development of new sensor technologies (UWB), as far as Line of Sight (LoS) goes. As far as observability goes, graph-based solutions present a distributed and dynamic structure that can handle agents connecting and leaving at different points in time, i.e. intermittent observability.
Although the history thus presents the foundational ideas for cooperative localization, limited observ-

ability and low-cost platforms still draw the attention of research to develop new CL solutions. Moreover, the rise of MAS has driven research towards distributed solutions.

2.3. Distributed CL state-of-the-art

The field of distributed multi-agent cooperative localization is expanding in two directions regarding sensor fusion, with both filter-based methods and graph-based methods being actively sought after. Algebraic methods exist, but assume limited or no noise on the measurement [15]. The main difference between the filter-based and graph-based methods is that filters have an active optimization window of only 1 sample: the current state. On the contrary, graph implementations keep all states, or a sliding window of states, in the active optimization window, trading off accuracy, through retained information in graphs, for efficiency and ease of implementation, through optimizing only the current state estimate in filtering.

2.3.1. Filter based methods

The Extended Kalman Filter (EKF), where the non-linear optimization problem is linearized locally to perform inference and give an estimate, is a popular algorithm and widely applied in CL problems [16], [8], [25]. This popularity is especially due to ease of implementation, satisfactory estimation accuracy and low computational complexity [16]. Papers such as Chenchana *et al.* [26] extend this to a Multi-State Constraint Kalman Filter (MSCKF) by saving the current state and a history of states in a sliding time window.

Some problems are inherently not Gaussian, such as underwater cooperative localization, due to the propagation medium, water, varying with salinity, temperature, depth, not to mention the multipath severity, where signals bounce off surfaces and arrive at the target multiple times, distorting the measurement [27]. For those problems, Kalman filter based approaches are not suitable, and algorithms like Particle Filter (PF) can be used. In PF, a number of particles are drawn from a probabilistic distribution, after which a weighted combination of the particles is made that determines the hypothesis on the location of the agent. In this way, it provides feasibility in nonlinear non-Gaussian cases and maintains the possibility of optimal results through its diverse set of state hypotheses. While resampling techniques exist to mitigate issues like particle depletion and Li *et al.* [27] detail a method with low communication cost, PF based methods incur high computation costs, having to evaluate the measurement models on each of the particles, rendering them impractical for low-cost, resource-constrained systems.

Covariance Intersection (CI) filtering (CIF) is another method in distributed CL problems [13]. CIF uses convex combinations to fuse information from two sets of unknown variables that are correlated. CI is a conservative algorithm, assuming correlations are present and thus using less of the available information for prediction to decouple this correlation, instead of Kalman Filtering that assumes independency of information and might lead to overconfidence. However, CI is a distributed single-step fusion technique [13]. Because it does not track state history, like other filtering techniques, and it assumes worst-case cross-correlations between agents, it inherently disregards useful information.

2.3.2. Algebraic approaches

Recently, Chen *et al.* [15] developed mathematics for algebraic determination of relative position through measurement of interior angles in multi-agent triangles. Although this provides full mathematical guarantees and is a complete formulation, it assumes perfect measurement and is thus still a limited approach. Liang *et al.* [28] build on top of this, accounting for measurement uncertainty, essentially transforming it into a Bayesian inference problem, specifically a MAP estimator, thereby Bayesian Inference (BI), which then makes it similar to much of the Graph-Based Methods discussed next.

2.3.3. Graph-based methods

Graph-based methods, in contrast to filter methods, keep a history of states in the optimization window. By building a graph with nodes for states and measurements, they can optimize over the entire history, or a sliding window of the history, rather than just the single present state variable.

A significant amount of recent advancements in multi-robot cooperative localization use the advancements made in distributed Pose-Graph Optimization (PGO). PGO is the specialized problem formulation from the field of Simultaneous Localization and Mapping (SLAM), where the nodes in the graph represent robot poses and the edges represent relative pose constraints between two poses (from e.g. odometry or loop closures). Each edge encodes a relative measurement: $z_{ij} = f(x_i, x_j) + noise$, and the goal is to find the poses x_i, in order to minimize the error over all the edges:

$$\min_{x} \sum_{(i,j)} \|f(x_i, x_j) - z_{ij}\|^2_{\Sigma_{ij}} \tag{2.1}$$

Therefore, PGO is a nonlinear least-square problem built specifically for the graph of poses. The graph optimization problem can be solved by numerical optimization methods. For example the Gauss-Newton and Levenberg-Marquardt methods are widely adopted "due to open source graph optimization tools, such as g2o, ceres, gtsam, SE-sync" [12].
However, recent papers choose to perform relaxation to make the PGO problem linear, and then solve them with linear solvers. This implies trading off the geometric accuracy of the problem that one has in the nonlinear least-squares problem, for convexity (ensuring no local minima where the solution gets stuck), robustness (insensitive to initialization) and scalability (computational efficiency).
Todescato *et al.* [29] use a simple gradient based approach for this. Choudhary *et al.* [30] divide the problem in two stages, first solving a relaxed version of the problem to get a good estimate, and secondly do a Gauss-Newton iteration to improve. This is done in a distributed way using the Distributed Gauss-Seidel algorithm. In its subsequent work Choudhary *et al.* [31], use chordal-relaxation to make the PGO problem linear, and then use a Gauss-Seidel linear solver. On the other hand, Tian *et al.* [32], Tian *et al.* [33] use Semi-Definite Programming (SDP) relaxation and block coordinate descent on riemannian manifold to come to a solution. The difference is that the former chooses a more efficient relaxation, and thus scales well to large pose graphs, but abandons the formal global optimality that the SDP relaxation can achieve if it is tight. Tian *et al.* [33] for that reason thus also claim certifiable correctness , even though the solution functions in a distributed and asynchronous manner. Tian et al. [5] even built a SLAM system with this approach, arguing its practicality.
Alongside this, Alternating Direction Method of Multipliers (ADMM) has also been named as a "particularly attractive and versatile distributed optimization method for multi-robot systems" [34]. It naturally handles constrained problems by breaking the global map optimization into independent local tasks and mathematically penalizing disagreements between robots until they reach a consensus. Ultimately, whether solving via Gauss-Newton, block coordinate descent or ADMM, the formulation of the PGO problem is a specific case of a broader probabilistic framework known as a factor graph.

Factor graphs provide the general framework for multi-robot localization, such as in Cunningham *et al.* [35], Cunningham *et al.* [36], where they use factor graphs in combination with Gaussian elimination. In this approach robots exchange Gaussian marginals about shared variables; these marginals act as compressed messages, communicating summaries of the robot's state estimates and uncertainties without needing to send entire state histories. However, in [36] the communication increases quadratically with the number of shared variables. Wang *et al.* [21] propose an alternative message passing strategy to improve on iterations needed for convergence and thus reduce communication, while Murai *et al.* [37] proposes restructuring the graph to restrict external communication, but slightly increasing local computation in the process.
One of the drawbacks of using a factor graph is that it keeps the entire history in the optimization window, making it computationally intensive. Marginalizing out the old states to keep only a limited number of states in the window is common practice to combat this dimensional explosion [37], [38]. This marginalization means mathematically absorbing their information into the current state's prior so they can be dropped from the active graph without losing their historical influence.
Keeping the entire history also calls for a different approach as to the frequency with which one adds measurements, factors, to the graph and iterate, as opposed to the memoryless filter architecture, where ideally it runs at a much higher frequency. This reduced running frequency for the factor graph leads to new solution, mainly for sensors like IMU that typically output measurements at up to 1 kHz. State of the art sensor fusion implementations using factor graphs, such as those abundantly present in visual-inertial odometry domain, make use of a technique called IMU preintegration, which bundles hundreds of IMU measurements into a single relative motion constraint, preventing the factor graph

from being overwhelmed by too many nodes [39], [9], using the technique to “utilize the IMU precision fully”, by not throwing any measurement results away, but also not reintegrating the IMU in the factor graph.

Still factor graphs provide a generalized framework for distributed CL. Factor graphs exploit the inherent dependency structure of the complex estimation problems in CL [40]. A well-known message passing algorithm that operates on factor graphs is Belief Propagation (BP), also known as the sum-product method.
The literature reflects the widespread use of belief propagation. There are those who suggest Non-parametric Belief Propagation (NBP) [41], [27], assuming no prior knowledge about the positions. They then generate N_p weighted particles to approximate the beliefs of the agents. The calculation of the messages between the nodes in the factorgraph then involves approximating the belief particles of the transmitting node, and generating particles randomly from all incoming messages based on distance and angles. Then each particle is weighted equally by $1/N_p$. Particle-based methods work well for the case where the assumption of Gaussianity cannot hold, such as in underwater environments [27]. Furthermore, they allow for an accurate estimation of the uncertainty [1]. However, for a limited number of particles, performance is not guaranteed [41]. Nevertheless, if the number of particles is increased, the approximation is more accurate. This means NBP presents a trade-off between accuracy and computational burden. Therefore, it is interesting to look at the design of computationally efficient alternatives, based on parametrization of BP.
What remains to be solved in this computationally efficient parametrized BP is dealing with nonlinear measurement models. There are two main ways to deal with this nonlinearity. The first being sigma-point methods, such as the unscented transform, which handles non-linearities by passing a small, chosen set of sample point (sigma points) through the non-linear function instead of trying to linearise the function itself. The second being analytic linearization. The first procedure linearizes by approximate Statistical Linear Regression (SLR). Therefore it is convenient to use sigma-point belief propagation (SPBP) [42] or SLR [43]. However, sigma-point BP does not work properly if the number of neighbors is high, as this yields a high-dimensional space, rendering the sigma-points outside the region of interest [43]. Nonetheless, in the proposed scenario for this thesis, the number of neighbors would not cause issues, but rather the computational demand of evaluating non-linear measurement functions multiple times per update.
The second way to deal with nonlinearity is analytical linearization, where one linearizes around the prior mean in the presence of nonlinear factors/functions via first-, or higher order Taylor expansion, transforming the non-linear measurement model into a linear system, maintaining computational efficiency. In the case where there is a sufficiently accurate prior or sufficient number of anchors or a large number of neighbors, beliefs are much more likely to be unimodal. This means they have one peak in the belief distribution, so only one hypothesis about where the node is. In this case Gaussian Belief Propagation (GBP), with analytic linearizaton, can be used satisfactorily [43].

2.3.4. GBP for distributed localization

GBP is a special case of general loopy Belief Propagation [23]. GBP computes approximate marginal distributions, known as beliefs, via an iterative process using purely local algorithmic message passing between the nodes of the graph. Note that this message passing on graph nodes is the term used in these algorithms, and does not necessarily imply actual physical communication.

GBP is a popular choice because of the following favorable characteristics:

- GBP is naturally distributed, and thus suited for distributed computation.
- GBP provides good scalability as the communication cost grows linearly with the node degree.
- The algorithm handles asynchronicity well. With messages from only part of the neighbors it can already re-iterate and improve the estimate. Using inbox / message caching asynchronous clocks can also be handled.
- It has proven to be robust to communication failures and outliers [44].
- According to A. Davison [23], the founder of Imperial College dyson robotics lab that is leading the world in SLAM efforts with little resources combined with AI, the algorithm also bears a forward-looking architectural fit (spatial AI) due to compatibility with graph processors, incremental and

dynamic structure, flexibility to incoming constraints.

- Last, and foremost, It provides a unified framework, that can also be applied to path planning, global map forming (in SLAM) and consensus [45][46], and even object avoidance [3].

2.3.5. Conclusions

To summarize, in DCL one can use both filter-based and graph-based methods. Graph-based methods are preferred because of their inherent distributedness. Factor graphs then provide a general and dynamic framework. To perform inference on factor graphs iteratively, Belief Propagation is common, and Gaussian Belief Propagation is chosen because parametrization makes the solution computationally efficient, and Gaussian is a valid assumption for many real world problems including our scenario.

2.4. Sensor suite

Apart from the algorithm choice for CL, there is always the discussion on what kind of sensor set to use. According to Yao *et al.* [8], sensor technologies can roughly be divided into two categories: First, absolute positioning, fixing a state/location to a specific coordinate frame. The most common example of this is GNSS. In the absence of GNSS, local communication sensors, such as Wi-Fi, Bluetooth Low Energy, or ultrawideband (UWB), can act as absolute positioning references, when using fixed beacons. However, all of these technologies are ”susceptible to interference from external factors” [8]. Therefore, they should not be the primary or only sensor used in a scenario under adversarial conditions.

Second, the more flexible and self-dependent option, relative positioning solutions, calculating position relative to some initial location that acts as a reference in the local coordinate frame. This offers a wide range of technologies, with millimeter wave Radio Detection And Ranging (RADAR) as the most established and most robust to interference, owing to longer wavelengths. More recently, radar is also used to measure velocity via doppler frequency shift measurement [47]. Light Detection And Ranging (LIDAR) which can give very high accuracy given sufficient measurements, up to sub-centimeter level using terrestrial laser scanning [48]. Optical-vision based methods, deducing position from a camera, such as [9], [10] is still vulnerable to feature-poor environments and illumination changes, making it unfit for battlefield use. The use of IMU/INS, comprising of sensors measuring forces, such as accelerometer, gyroscope, magnetometer and barometer is very common [8], [9], [16], as it provides a low-cost, short-term accurate solution, which can hardly be influenced by external factors. However, the IMU/INS suffers from drift over longer periods of use, as do most relative solutions. Therefore, technologies are combined using algorithmic fusion of these different sensors, by means of the aforementioned filtering or graph-based methods. The local communication sensors, such as UWB in [49], as mentioned before, can be used for determining relative distances and thus cooperative localization in a MAS scenario.
In view of the above, a combination of IMU/INS with a local communication sensor is most sensible for the operational scenario with multiple low-cost UAVs in an adversarial, and thus feature-poor, environment.

2.5. The role of anchor points and heterogeneity

Research stresses that many localization algorithms, both CCL and DCL, still depend on fixed anchors [17]. Even recent studies into cooperative localization in GNSS-denied environments still employ ground stations in some cases [6]. This supports looking into anchor points, the approach taken in this thesis, or in the specific case where fixed beacons are not possible, such as hostile environments or large distance environments, looking into device heterogeneity that can serve as a method of anchoring the estimations.

The question then arises whether the geometry of anchor points is relevant. Looking at the well established research field of wireless sensor networks, nowadays called IoT, we see that its relevance has been stressed [50]. Recent work [1], supports this, investigating optimal formation and motion to maintain accurate localization. Although they do not use classical fixed beacons, they employ agent heterogeneity, specifically decreased uncertainty in Unmanned Ground Vehicles (UGV) over UAV, to act as anchoring and improve UAV estimates. Namely, UGVs benefit from more stable dead-reckoning, through ground-based odometry, compared to drifting flight dynamics for UAVs.

There are multiple sources employing this heterogeneity. For example the UAV-UGV combination, exploiting the fact that UGV's are less weight and power constrained, leading to better sensing and localization quality [19].
This idea of heterogeneity can even work the other way around, as investigated in e.g. [51] where UAVs help UGV localize, because the former has valuable Line of Sight (LoS) observations, where the UGV cannot observe, leading to better overall localization.
In cases where agent A is clearly better at localization than agent B and C, the choice can also be made for a top down approach, as in [52]. Wanasinghe *et al.* [52] exploit this heterogeneity and top-down structure through a leader-assistive framework, thereby limiting communication to one-way. This could be beneficial in hostile environments.

The generalizability to more problem instances is identified as it also appears in the context of phone localization in urban canyon environments, which are also GNSS-denied, where the same philosophy is employed. Sou *et al.* [53] exploit device heterogeneity in wearable devices such as smartwatch and smartphone, to improve person localization. Zhou *et al.* [17] use mobile robots as anchors to improve person/phone localization.
Overall, the literature clearly indicates that in the absence of traditional fixed beacons, strategically using the geometric and sensing heterogeneity of different agents provides an effective and adaptable method for anchoring cooperative localization across domains.

2.6. Concluding remarks

In conclusion, the literature presents a clear trajectory for facing the challenges of distributed cooperative localization in GNSS-denied, adversarial environments. To mitigate the inherent drift of low-cost dead-reckoning sensors, sensor fusion methods can strategically combine strenghts and weaknesses over different sensors. While filter methods are well established in literature, graph-based methods, specifically gaussian belief propagation, emerge as the most robust framework. GBP provides the necessary distributed architecture, scalability and resilience to fuse the relative and dead-reckoning measurements, establishing an infrastructure-free sensing baseline. Furthermore, the literature illustrates circumvention of traditional fixed anchors by exploiting heterogeneity in agent roles or abilities. Together, resilient relative sensing, sensor fusion through GBP and exploiting heterogeneous roles for anchoring form the architectural foundation for the multi-agent system approach proposed in this thesis.

3

Technical Background

This chapter establishes the necessary theoretical prerequisites for distributed cooperative localisation. It gives an overview of the technical background of Gaussian Models, used to model measurement uncertainty, factor graphs, as a representation of the full system, and Gaussian Belief Propagation, as a distributed algorithm operating on factor graphs. However, for a more in-depth and especially visual interactive explanation, we encourage the reader to view the work of J. Ortiz, T. Evans, and A.J. Davison [54].

3.1. Gaussian models

We are interested in modeling the uncertainty on the measurements, as by factoring these in, we can make an improved estimate, especially for high uncertainties that low-cost platforms yield. Uncertainty is commonly represented by Gaussian Models for a number of reasons:

- Accurate reflection of reality: Gaussian Models are an accurate representation of many physical phenomena and sensor measurements in reality. This is, as long as the distributions are unimodal [55].
- Mathematical simplicity: the distribution has a simple mathematical form, making it easy to understand and to work with.
- Simplifying complex operations: complex operations can be expressed in simple formulae as Gaussians are easily summarized by their parameters.
- Preserved mathematical form: Gaussian models are closed under common statistical operations of marginalization, conditioning and taking products.

The Gaussian distribution can be expressed exponentially $p(x) \propto e^{-E(x)}$ with a quadratic energy function E(x). This representation is convenient for use in factor graphs, as it makes variable estimate inference straightforward, as wil be explained later, per Equation 3.6.
There are then two ways to write the quadratic energy, which also happen to be the two common parametrizations for multivariate Gaussian distributions, as shown in Equation 3.1.

$$\underbrace{\mathcal{N}(\mathbf{X};\boldsymbol{\mu},\boldsymbol{\Sigma})}_{\text{Moments form}} = \underbrace{\mathcal{N}^{-1}(\mathbf{X};\boldsymbol{\eta},\boldsymbol{\Lambda})}_{\text{Canonical form}} \tag{3.1}$$

With their respective energy equations:

$$E_{moments}(x) = \frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu) \tag{3.2}$$

$$E_{canonical}(x) = \frac{1}{2}x^T\Lambda x - \eta^T x \tag{3.3}$$

The moments form in Equation 3.2 is more computationally efficient and simple for marginalization. Nevertheless, the canonical form in Equation 3.3 is often preferred in inference [54]. Namely, the

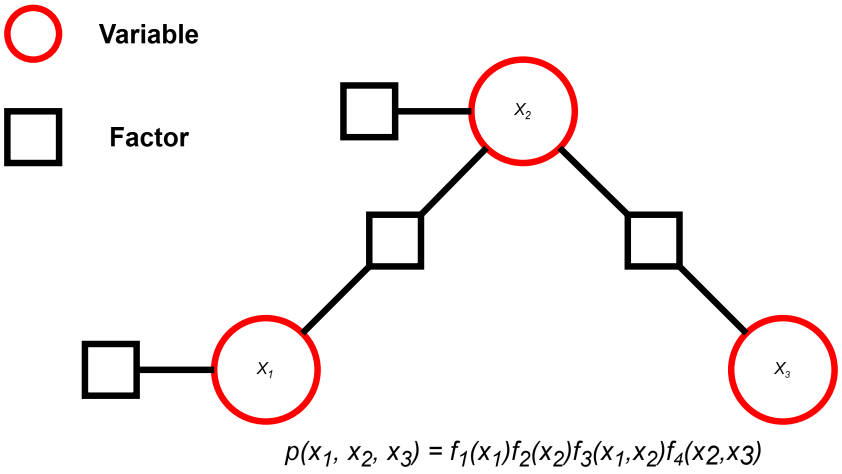


Figure 3.1: Visualization of a factor graph. Figure adapted from [54].

canonical form makes taking a product straightforward, and so computing the posterior from the factors is easy. Secondly, in the canonical form one makes use of the precision matrix, which is sparse, reducing computational load, and reflects the structure of the factor graph, facilitating understanding.

3.2. Factor graphs

Factor graphs are used in order to break down the global estimation problem into local, manageable pieces. Factor Graphs are a general and robust model for systems involving interdependent variables and constraints, particularly insightful in probabilistic inference and optimization. A factor graph is a type of bipartite graphical model $G = (X, F, E)$ consisting of two types of nodes: variable nodes $X = \{x_i\}_{i=1:N_v}$, which represent the variables of interest, and factor nodes $F = \{f_s\}_{s=1:N_f}$, which represent the functions (i.e. factors) that define the relationships between the variables. The variables and factors are connected by edges E. The factor graph represents the factorization of the joint distribution $p(X)$, as per Equation 3.4. An example factor graph is visualized in Figure 3.1.

$$p(X) \propto \prod_{s=1}^{N_f} f_s(X_s), \quad \text{where} \quad X_s = n(f_s) \tag{3.4}$$

In this, $n(v)$ is the set of neighboring nodes connected via edge to the node v. In Gaussian factor graphs, the factors f_s take the form of Gaussian distributions. Now taking the energy form of the Gaussian distribution form Equation 3.2, the factor then looks like Equation 3.5.

$$f_{\boldsymbol{s}}(\boldsymbol{X}_s) \propto e^{-\frac{1}{2}(\boldsymbol{z}_s - \boldsymbol{h}_s(\boldsymbol{X}_s))^\top \Lambda_s (\boldsymbol{z}_s - \boldsymbol{h}_s(\boldsymbol{X}_s))} \tag{3.5}$$

Here, $h_s(X_s)$ represents the measurement function of the factor. z_s represents the actual observed or the expected value depending on the context, and Λ_s is the precision matrix (inverse covariance) of the factor.
Representing the factor graph with energy-based models, as in Equation 3.5, the most likely variable configuration, the Maximum A Posteriori (MAP) estimate, is found as the sum of the factor energies, making variable estimate inference straightforward:

$$X_{MAP} = argmin_X - log\ p(X) = argmin_X \sum_i E_i(X_i) \tag{3.6}$$

$h_s(X_s)$ thus determines the expected measurement based on the involved variables X_s. Take for example in the case that a drone observes another drone, then h_s function/factor could be the Euclidean distance between the first drones' position and the second drones' position. Meanwhile z_s would be the actual range measurement obtained from an onboard sensor.
In this way, one can also use factors to form priors, which are not sensor measurements, but rather

assumptions on the certainty of the initial positions, or external knowledge. A parallel could be drawn between this factor formulation and a non-holonomic constraints in Kalman filtering. In this case, the measurement, or rather expectation is $z_s = 0$, meaning the factor energy is purely a function of the states.

3.3. Belief propagation

Belief Propagation is a naturally distributed algorithm, thus architecturally fit for our operational scenario. It performs marginal inference, i.e. it computes the marginal posterior distribution of each variable from the set of factors that make up the joint posterior. BP is linked to factor graphs by the following property: BP can be implemented as iterative message passing on the posterior factor graph. The algorithm operates by iteratively updating a node's locally stored belief through sending and receiving messages from its connected neighboring nodes. Each iteration has three phases, as shown in Figure 3.2.

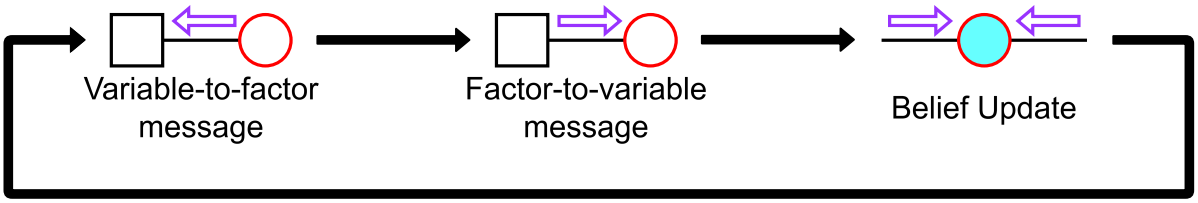


Figure 3.2: Belief Propagation in 3 phases. Figure adapted from [54]

1. Variable-to-factor message: A variable takes the product over the incoming messages from all connected factors except the target factor for the message. This message thus tells the target factor what the belief of the variable would be, were the target factor to not exist.
2. Factor-to-variable message: A factor takes the product over the incoming messages from all variables except the target variable for the message. It then marginalizes over all the other nodes' variables to produce a message that expresses the factors' belief over the receiving nodes' variables.
3. Variable Belief Update: The variable node beliefs are updated by taking the product over the incoming messages from all connected factors.

In short, BP was originally developed by J. Pearl [56] for graphs that are trees. BP for loopy graphs was therefore at first empirical. Theoretical grounds have however been developed. "Loopy BP minimizes the KL divergence between posterior and a variational distribution, which we use as a proxy for the marginals after optimization" [54]. However, this results only holds when the algorithm converges, i.e. after optimization. Loopy BP gives no theoretical guarantees on convergence. Neverthless, it was empirically determined that loopy BP generally converges to the true marginals, if graphs are not very loopy [57], [58].

3.4. Gaussian belief propagation

GBP is a specific case of loopy BP, where all factors and therefore the joint posterior distribution are Gaussian distributions. It was originally introduced by Weiss and Freeman [59].

1. *Variable to Factor Message*: a message from variable x_k to factor f_j is the product of all incoming factor to variable messages for x_k, excluding the message from the target factor f_j:

$$\vec{m}_{\mathbf{x}_k \to j}(f_j) = \prod_{f \in n(\mathbf{x}_k) \setminus f_j} \vec{m}_{f \to k}(\mathbf{x}_k) \tag{3.7}$$

2. *Variable Belief Update*: a variable x_k updates its belief by taking the product over all incoming messages from its connected factors:

$$b(\mathbf{x}_k) = \prod_{f \in n(\mathbf{x}_k)} \vec{m}_{f \to k}(\mathbf{x}_k) \tag{3.8}$$

3. *Factor to Variable message*: a factor f_s to variable x_k message is the product of all incoming variable to factor messages for f_s excluding the message from the target variable x_k, times the factor likelihood function, which then in turn marginalizes out all variables excluding x_k:

$$\vec{m}_{f\to k}(\mathbf{x}_k) = \sum_{\mathbf{x}\in\mathbf{X}_s\setminus\mathbf{x}_k} f_s(\mathbf{X}_s) \prod_{\mathbf{x}\in\mathbf{X}_s\setminus\mathbf{x}_k} \vec{m}_{\mathbf{x}\to f}(\mathbf{x}) \tag{3.9}$$

4. Lastly, there is the option to extend Gaussian Belief Propagation to the non-linear case. Using the first-order taylor expansion, the non-linear factor measurement function is linearized around the current estimate X^0. For a factor, its likelihood $f_s(X_s)$, with observation z_s and precision matrix Λ_s, is then represented as:

$$\boldsymbol{\eta}_f = \mathbf{J}_s^\top \boldsymbol{\Lambda}_s(\mathbf{J}_s\mathbf{X}_s^0 + \mathbf{z}_s - \mathbf{h}_s(\mathbf{X}_s^0)), \quad \boldsymbol{\Lambda}_f = \mathbf{J}_s^\top \boldsymbol{\Lambda}_s\mathbf{J}_s \tag{3.10}$$

3.5. Concluding remarks

This chapter has established the necessary theoretical prerequisites for distributed cooperative localization. By structuring the joint probability of the swarm as a factor graph, the global estimation problem is broken down into tractable, local pieces. Furthermore, by parametrizing the messages as Gaussians and linearizing the non-linear observations via Taylor expansion, inference can be performed efficiently on resource-constrained agents. The next chapter applies this algorithmic baseline to the proposed multi-UAV scenario, explicitly defining the measurement models and the architectural mechanics of the landed drone anchor.

4

Methodology

This chapter elaborates on the precise scope of the project and the contributions made by this work to extend the current literature. First the detailed formulation of the problem is introduced in Section 4.1, and its accompanying architecture and coordinate system in Section 4.2 and Section 4.3. This is then translated into the mathematics of the model in Section 4.4. Afterwards, in Section 4.5 we elaborate on how the model functions distributedly and evolves over time. Finally, we discuss our proposed model for the landed drone in Section 4.6.

4.1. Scenario Formulation

An operational scenario with a MAS of UAVs navigating through a GNSS-denied, adversarial environment is targeted. In this featureless terrain, the drones are forced to rely on distributed cooperative localization. The ultimate objective is for the drones to reach a specific target area, designated by a basket range (x meters in a circular perimeter from the target location). Because these drones are using IMU to perform dead reckoning and exchanging relative measurements, their position estimates will deviate from reality over time. Therefore, the primary metric to evaluate success of the objective is the Final Drift (FD) experienced per drone, as this value must remain bounded within the basket range for the objective to be met. Next to that, overall path fidelity is evaluated through the Absolute Translation Error (ATE) and Absolute Rotation Error (ARE) with the formulation as in [31], and communication load is monitored to ensure the viability of the solution on low-bandwidth hardware.

The core idea of this methodology is driven by the promise of achieving higher overall system effectivity through a calculated tactical sacrifice. Instead of attempting to bring every single drone to the target with high uncertainty, the system can deliberately land one unit out of the group mid-flight. By doing so, the landed drone becomes a physical, stationary anchor. If sacrificing just 1 drone out of a swarm of 100 improves the localization accuracy of the remaining 99 by 10%, the overall mission effectivity increases, making the sacrifice a worthwhile operational trade-off.

This concept further evolves into a multi-group tactic, as visualised in Figure 4.1, raising the possibility of planting beacons in a featureless environment by sending drones batch by batch. In this extended scenario, the first wave of drones can land one of its units to establish a stationary reference point. This newly planted beacon then serves all subsequent batches flying overhead, with the hypothesized effect of limiting their accumulated drift.

4.2. System Architecture

Each UAV in the system follows the homogeneous architecture as in Figure 4.2. Each UAV operates autonomously and is equipped with a specific low-cost sensor suite comprising of IMU for dead-reckoning and UWB/Radar for communications and relative range-bearing measurements. This also implies that sensing range and communication range coincide. Of course, each UAV also has a local processor that stores the measurement data in a factor graph representation and runs the GBP algorithm on it.

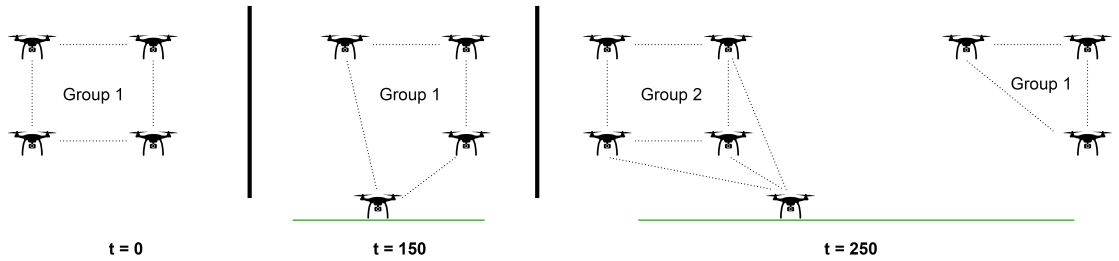


Figure 4.1: Visualisation of multi-group scenario. At t=0, the first group flies in formation. At t=150, a drone from the first group lands, breaking the formation. The second group leaves from the starting point later, and reaches the landed drone at t=250, forming communications to exchange beliefs.

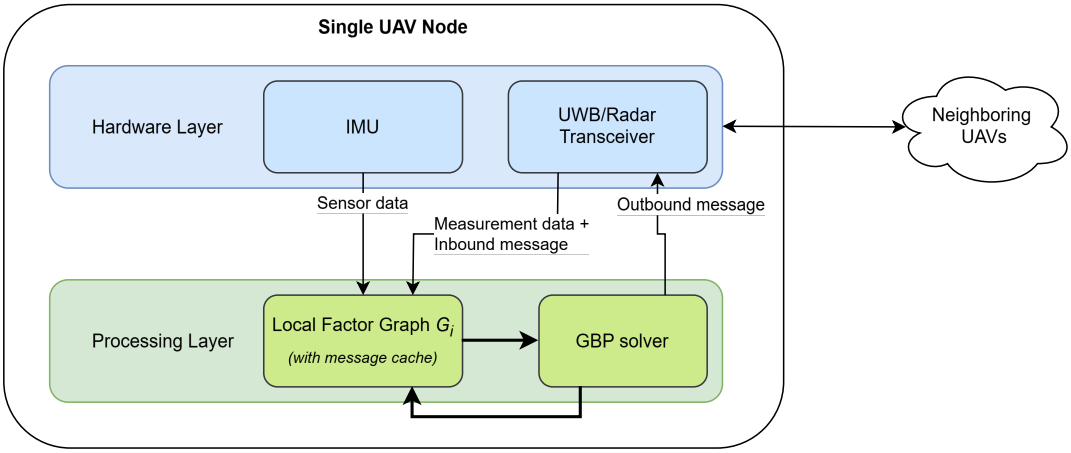


Figure 4.2: Block diagram view of UAV architecture.

It is assumed that IMU data is pre-integrated to obtain a low-frequency (1 Hz) relative pose estimate that is fed into the factor graph. The range-bearing measurements come directly from the UWB/Radar module.
The local factor graph G_i and the information that the outbound/external message holds, as well as the concept of message cache, will be detailed further in Subsection 4.5.1.

4.3. Coordinate system

The operational scenario concerns 10km-range formation flight for drones. Two simplifying assumptions are made:

- First, while UAVs operate in a 3D environment, this thesis simplifies to a 2D environment. This simplifying assumption is justified in long range flight for low-cost units, as barometers are excellent at detecting small changes in air pressure, especially at medium high altitudes [60], providing a more certain and less drifted height estimate, as compared to translation and orientation estimates provided by IMU.
- Secondly, because the scenario is long-range flight, one can assume a flight path with little turning. Consequently, using Euclidean coordinates and Euclidean math to investigate the research questions proposed in this thesis is justified. This forfeits the need to make use of alternate coordinate systems such as Quaternions or Lie Groups, and considerably simplifies the mathematics. This choice was made in conjunction with the choice to build our own simulator from the ground up.

Furthermore, the state of the drones is represented in the global coordinate frame, while the measurements are of course made in the local coordinate frame. This is visualised in Figure 4.3.

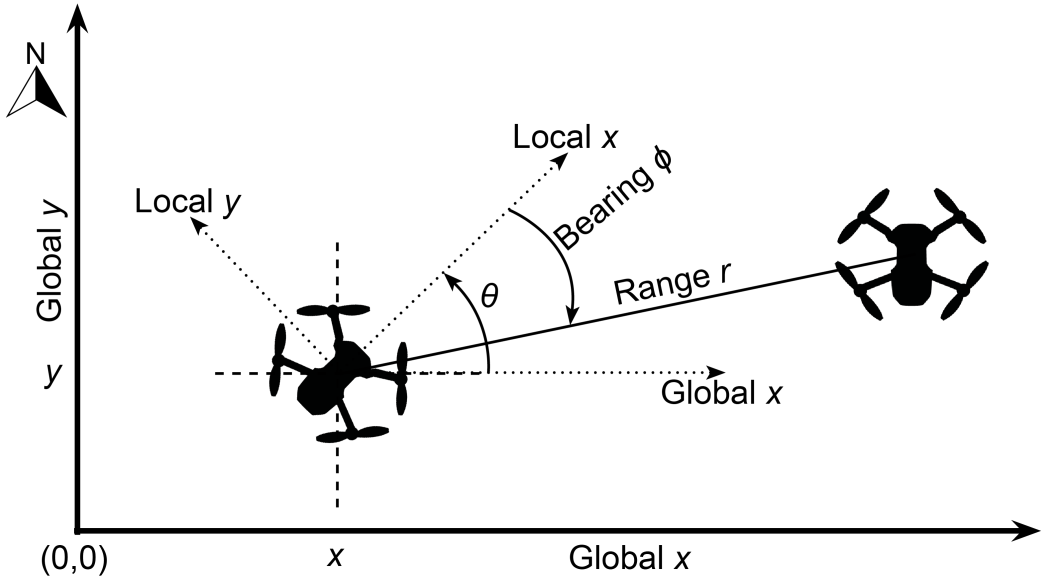


Figure 4.3: A visualisation of the Euclidean, 2D coordinate system that is used. The state of the drone is represented as $[\theta, x, y]$ in the global frame. Measurements are made in the local frame, where only the range-bearing measurement is shown.

4.4. Sensor and factor formulations

For different sensors, different measurement models and noise models apply. In this thesis we assume that dead-reckoning measurements come from an IMU in the form of full relative pose measurements. Furthermore, range-bearing measurements between UAVs are performed using UWB or RADAR based sensing. This is representative of many low-cost real-world scenarios.
All measurements will be modeled as:

$$z = h(X) + \epsilon \tag{4.1}$$

where $h(.)$ is the measurement prediction function, and ϵ is assumed to be Additive White Gaussian Noise (AWGN):

$$\epsilon \sim \mathcal{N}\big(0, \Sigma\big) \tag{4.2}$$

4.4.1. IMU relative pose measurements

We assume that the IMU provides us with relative pose measurements $z = [d\theta, dx, dy]$.The simplifying assumption is made that a unit moves on its local frame x-axis, meaning that the measurement is reduced to $z = [d\theta, dx, 0]$.
Now, we assume that for an IMU the noise is dependent on the magnitude of the motion. Therefore we can model the noise on the IMU measurements as:

$$\epsilon_{IMU} = \begin{bmatrix} \epsilon_\theta \\ \epsilon_x \\ \epsilon_y \end{bmatrix} \sim \begin{bmatrix} \mathcal{N}\big(0, \sigma_\theta\big) \\ \mathcal{N}\big(0, dx * \sigma_x\big) \\ 0 \end{bmatrix} \tag{4.3}$$

4.4.2. IMU factor

Now, having defined the noise model for the IMU, a manner of including the IMU measurement into the factor graph is still needed. Each factor is defined as a gaussian distribution as in Equation 3.5, Now, specifically for the IMU factor the mathematical definition is the following:

$$f_{IMU}\left(\mathbf{X}_{t-1}^{A},\, \mathbf{X}_{t}^{A};\, \bar{\mathbf{z}}_{t-1,t}\right) \propto \exp\left(-\frac{1}{2}\left\|\bar{\mathbf{z}}_{t-1,t} - h_s\left(\mathbf{X}_{t-1}^{A}, \mathbf{X}_{t}^{A}\right)\right\|_{\Sigma_o}^{2}\right) \tag{4.4}$$

where the measurement is related to the state variables $\mathbf{X}_{t-1}^{A} = [\theta_a, x_a.y_a]$ and $\mathbf{X}_{t}^{A} = [\theta_b, x_b, y_b]$ using the following measurement prediction function h_{IMU}:

$$h_{IMU}(\mathbf{x}) = \begin{bmatrix}\theta \\ x \\ y\end{bmatrix} = \begin{bmatrix}\theta_b - \theta_a \\ cos\,\theta_a \Delta x_w + sin\,\theta_a \Delta y_w \\ -sin\,\theta_a \Delta x_w + cos\,\theta_a \Delta y_w\end{bmatrix} \; where\; \Delta x_w = x_b - x_a.\Delta y_w = y_b - y_a \tag{4.5}$$

Now, since the measurement model is non-linear, it requires linearization to be compatible with GBP. This is achieved through first-order Taylor expansion, as in Equation 3.10. However, this requires the computation of Jacobian for the factor. Although baseline code was provided with jacobian calculation through small angle perturbation, the choice was made to determine the analytical jacobian for each factor. Namely, using the analytical jacobian is much more computationally efficient than the aforementioned numerical technique. The analytical jacobian is derived as in Section A.1.

4.4.3. Range-bearing measurements

We assume that a range-bearing measurement is provided in the local frame of the observing UAV. Furthermore, the range-bearing measurement is made by UWB or RADAR. In these sensor technologies, the noise is characterized as constant for the bearing (ϕ), but scaled with the magnitude for the range (r), In addition, the noise for the range also has a constant baseline noise component. This leads to the following noise model:

$$\epsilon_{RB} = \begin{bmatrix}\epsilon_\phi \\ \epsilon_r\end{bmatrix} \sim \begin{bmatrix}\sigma_\phi \\ \sigma_c + r * \sigma_r\end{bmatrix} \tag{4.6}$$

4.4.4. Range-bearing factor

In our setup, drones observe other drones using range-bearing sensors. We use 2D spherical coordinates (r, ϕ), i.e. radial distance and azimuthal angle.
The range-bearing factor relating the sensing drone A and the observed drone B at time t is:

$$f_{rb}\big(\mathbf{X}_t^A, \mathbf{X}_t^B; \bar{\mathbf{z}}^{AB}\big) \propto \exp\left(-\frac{1}{2}\left\|\bar{\mathbf{z}}^{AB} \;-\; h_{rb}\big(\mathbf{X}_t^A, \mathbf{X}_t^B\big)\right\|^2_{\Sigma_s}\right) \tag{4.7}$$

where $\mathbf{X}_t^A = [\theta_a, x_a, y_a]$ and $\mathbf{X}_t^B = [\theta_b, x_b, y_b]$:

$$h_{rb}(\mathbf{x}) = \begin{bmatrix}\phi \\ r\end{bmatrix} = \begin{bmatrix}atan2(y_b - y_a, x_b - x_a) - \theta_a \\ \sqrt{(x_b - x_a)^2 + (y_b - y_a)^2}\end{bmatrix} \tag{4.8}$$

The analytical Jacobian for linearisation is again derived and is included in Section A.2.

4.4.5. Prior factor

The prior factor is used in order to model the prior knowledge about a state variable. For example, the initial position of the agents in the system is known, and one can model this with a prior factor. The prior factor is a unary anchor factor, i.e. connecting to one variable and constraining it to an absolute reference point. It has the same formulation as the GPS factor with the only difference that it also constrains the initial heading.

$$\epsilon_{prior} = \begin{bmatrix}\mathcal{N}(0, \sigma_\theta) \\ \mathcal{N}(0, \sigma_x) \\ \mathcal{N}(0, \sigma_y)\end{bmatrix} \tag{4.9}$$

$$h_{prior}(\mathbf{x}) = \begin{bmatrix}\theta \\ x \\ y\end{bmatrix} \quad where\; J(\mathbf{x}) = \mathbb{I}_{3\times 3} \tag{4.10}$$

4.5. GBP for distributed inference

Gaussian Belief Propagation, as detailed in Section 3.4, is an iterative message passing algorithm on a factor graph. However, in order for it to function distributedly, the factor graph needs to be split up over the drones in the system. Next to that, for the algorithm to function online on a drone, the factor graph needs to evolve over time, fusing in new measurements as time continues.

4.5.1. Splitting up the factor graph

Firstly, we split up the graph as follows. Let G be the full, global factor graph. In distributed GBP, each UAV owns a part of the factor graph G_i, and the union of all these graphs is $G = \cup_{i \epsilon \Omega} G_i$.
Each UAV owns its own pose variables and the factors for the observations that it has made. Organising it in this way ensures that the details of measurement factors, that can depend on the type of sensor or calibration, need only be known by the drone making the observation.
The pose variables and measurement factors are the nodes that make up the local graph of each drone G_i, causing the graph to be split up as shown in Figure 4.4. This architectural choice is significant as the entire graph G is partitioned among the UAVs Ω. This means that the marginal estimates obtained by distributed GBP are exactly the same as the marginal estimates obtained via centralised GBP on the global graph, under the assumption of perfect communication.
Distributed inference is achieved by each UAV $a \epsilon \Omega$ performing GBP message passing on their local

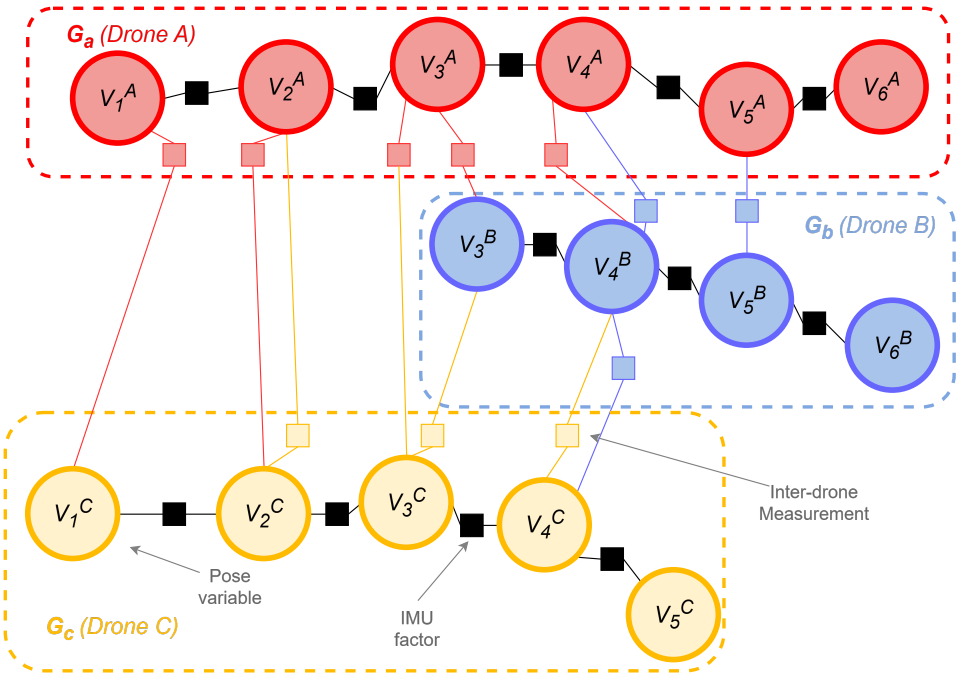


Figure 4.4: The factor graph is distributed over the drones. Each drone's local graph G_i owns its pose variables and factors for the measurements that the drone makes. Only inter-drone measurements require inter-drone communications. Figure adapted from [44].

graph G_i. This implies that computation load is thus also partitioned over the MAS.
Along the edges of factors f_{ab} that connect multiple UAVs together, a factor-to-variable message is sent via inter-drone communication. This is the only type of message that requires actual communication in real-life.
A factor-to-variable message takes the form of a Gaussian distribution in canonical form. A variable-to-factor message takes the same form, with the addition of sending a linearisation point $\bar{x} = \mu$.

$$\vec{m}_{f \to v}(\mathbf{x}_k) = \mathcal{N}(\eta, \Lambda) \qquad \vec{m}_{\mathbf{x}_k \to j}(f_j) = \mathcal{N}(\eta, \Lambda), \bar{x} \tag{4.11}$$

A variable to factor message always sends the mean μ in addition as linearisation point for the factor, to account for the situation where the variable has only just been initialised. Namely, the variable then has a precision Λ of 0 and an information η of 0, leaving a deadlock where the mean μ cannot be calculated from the standard Equation 4.12, because lambda is singular.

$$\mu = \Lambda^{-1} \eta \tag{4.12}$$

A pseudo-inverse or linear system solve of Equation 4.12 in this case leads to a mean $\mu = 0$, which would lead to incorrect results. This is prevented by sending the linearisation point $\bar{x}$ (the mean μ) with the message explicitly.

Given now that the graph G is split up, at each iteration, each unit performs on its own graph G_i thus the following steps:

1. Message computation: Locally on each of the UAVs, variable-to-factor messages are computed using Equation 3.7, and factor-to-variable messages are computed using Equation 3.9.
2. Message exchange: Messages are exchanged in the graph **internally** within a UAV, and **externally** between UAVs. An **internal** message is a message which is exchanged between nodes of the same local graph G_i, while an **external** message is a message which is sent from one UAV to another via actual communication media. An **internal** message thus actually entails only a computation on the processor local to the UAV, while an **external** message requires actual communication, thus incurring much higher costs in terms of time and energy consumption.
3. Variable belief update: once messages are exchanged, the beliefs of the variables are updated via Equation 3.8.

The process is repeated iteratively, until convergence.

Message passing schemes

Now, the literature reflects the development of various message passing schemes over the graph [21], which can be either parallel or sequential. In sequential message passing, nodes compute their local optimizations and transmit messages in a specific coordinated order, whereas in the parallel version, all nodes compute their local optimizations and broadcast their messages simultaneously. The advantage of a sequential message passing scheme is that new information on one end of the graph can propagate much more efficiently to the other end, leading to convergence in fewer iterations and higher accuracy. However, sequentiality incurs high latency per iteration and poor scalability. Parallel message passing, on the other hand, has the advantage of being much faster in single iterations, and no global coordination of the scheme is needed, ensuring decentralization and simplicity [21].
This thesis uses the fully parallel message passing scheme, in order to provide a fully distributed solution. Theoretically, the graph converges after one message passing iteration when only an IMU factor is instantiated in a timestep. Therefore, the faster single iteration nature of parallel message passing is also favorable.

4.5.2. Evolution of the graph over time

The factor graph in this scenario grows and evolves as time progresses. Drones are initialised with just one variable node. As time progresses, the drones measure, and consequently calculate through dead-reckoning, their own displacement incrementally, through integrating the acceleration and angular velocities from IMU, adding new IMU factors to their internal graph. These IMU factors connect each of the consecutive pose variables.
The GBP algorithm runs continuously on each drone's internal graph, producing updating marginal distributions for each variable. The internal message passing can take on different patterns. Since the internal message passing in reality comes down to processor computations, it is preferred to choose the pattern that leads to most rapid convergence possible. As mentioned before, to adhere strictly to a distributed system, the message passing scheme chosen is the fully parallel scheme.

Each of the factors that is connected to another drone's graph, caches the latest message it received from the other drone. This way, the local graph can keep updating even when communication is lost, just only updating that particular factors' message when inter-drone communication allows for it again. This architecture also enables asynchronicity of communication.
It is assumed for now that the drones have synchronized clocks for timestamping of measurements, so that the graph does not suffer from time-biasing [61].
As the factor graph grows with each measurement, the storage, computation and communication load grow with it. Now, windowed GBP is the commonly used solution to this, where instead of keeping the full history of the drone in the graph, a sliding window of variables is retained. This allows the size, and thus computation and communication needed, to be fixed.

4.6. Model for landed drone

The main addition of this thesis is a factor graph formulation for modelling the UAV in the MAS that is landed. The hypothesis is that landing a drone, and knowing that it is stationary, can improve localization accuracy of the landed drone, and in turn improve localization of the MAS. Multiple models were deemed viable and evaluated. Below we elaborate on these models.

4.6.1. Unary anchor factor

The most trivial way to model the fact that a drone has become stationary in the factor graph is by adding a unary anchor factor. This is a factor only connecting to the current state variable of the landed drone in the graph. The factor constrains the variable state to the most recent calculated belief mean, i.e. to the pose/state that it has just before landing, as shown in Figure 4.5.
However, this method is limited, as it directly constrains the pose $[\theta, x, y]^T$. Therefore, the pose is immovable and the landed drone cannot improve its location estimate once it is stationary.
The hypothesis is that in using this factor, the performance will be proportional to the drift of the landed anchor at the time it lands, and therefore it does not lead to performance gain.

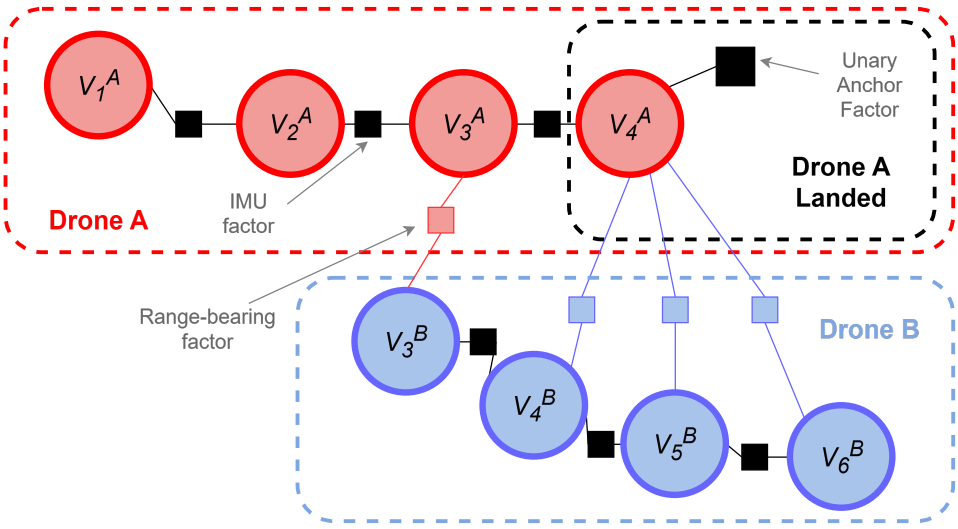


Figure 4.5: The unary anchor factor connects only to one variable, constraining it to an absolute pose $[\theta, x, y]^T$.

4.6.2. Zero-update (ZUPT) factor

In contrast to the unary anchor factor, the zero-update factor does not directly constrain the pose. The zero-update is a factor with a measurement of 0 velocity, connecting the previous and current state variable of the drone. It is comparable to the IMU factor in that it provides a measurement of full relative pose, i.e. velocities in the local frame of the drone. However, this zero-update factor is given a measurement of exactly 0 in all directions. Zero-velocity update has been common practice in kalman filtering [62], and has also been implemented in factor graphs before, although to a limited extent [63]. Conceptually, by constraining the measurement to be zero rather than the state itself, the state is left malleable. This way, the landed drone can still improve its state estimate, instead of freezing it with an absolute unary anchor factor.
Furthermore, the zero-update factor differs from a regular IMU measurement factor, in the fact that there's no sensor-determined uncertainty for it. Therefore, one can play with the optimal uncertainty σ, where increasing σ leads to a looser constraint and more movement in consecutive variables, while decreasing σ constrains the variables to be more closely clustered, as visualised in Figure 4.6.
The hypothesis is that this model will outperform the unary anchor model. However, since it artificially adds uncertainty at every timestep, the covariance of the state will not permanently shrink, limiting its applicability as an anchor to the rest of the system.

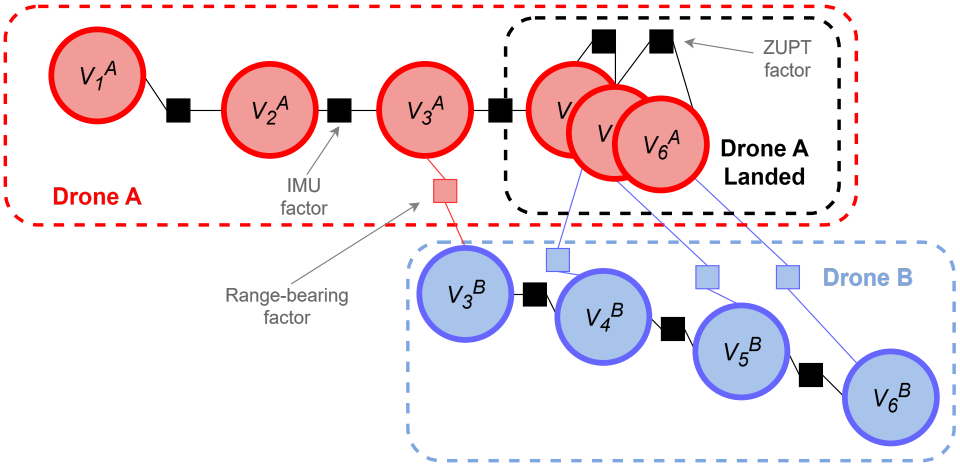


Figure 4.6: Zero Update factor model of landed drone in factor graph. The zero update factor constrains the movement of the pose variables based on its uncertainty σ, making consecutive estimates overlap spatially.

4.6.3. Persistent variable

The persistent variable model for the landed drone works as shown in Figure 4.7. Normally, at each timestep, the drone instantiates a new variable in the factor graph, and connects it to its previous variable via an in-between factor (being IMU factor or 0-update factor in this case).
In this model, the drone stops creating new variables in the graph once it lands. This way, all range-bearing factors from other drones, also from timesteps in the future, connect to this same variable of the landed drone. Consequently, the persistent variable grows into a densely connected hub. The

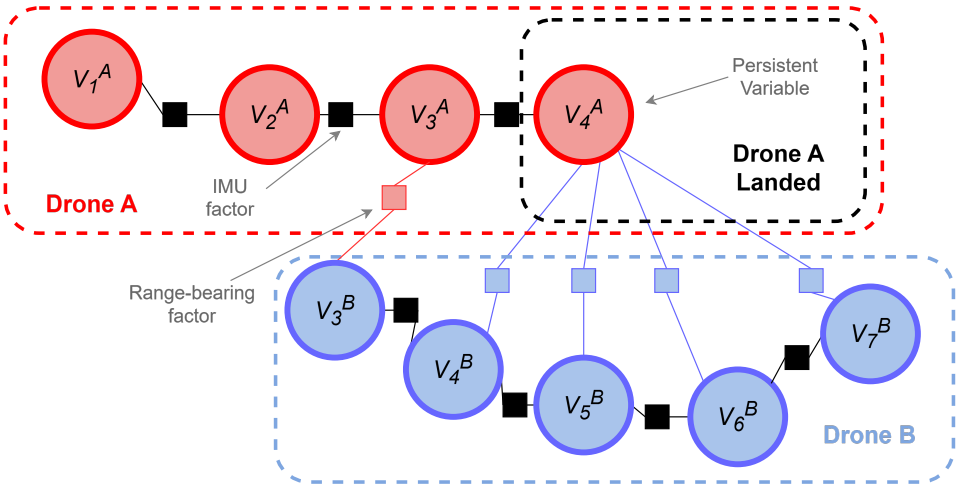


Figure 4.7: Modelling the landed drone in the factor graph by not creating new IMU factors or variables for this drone. All inter-drone factors then connect to the latest variable, creating a densely connected hub in the graph.

hypothesis is that the belief of this variable is going to converge to a small error, as all the independent range-bearing measurements will each add information pulling the pose towards ground truth. Furthermore, in this model, information and precision are strictly additive and its covariance will shrink until it functions as a rigid anchor.

4.7. Concluding remarks

This chapter defined the architectural and mathematical methodology for the proposed multi-UAV system. It details the standard relative factors and proposes three distinct models for a landed anchor drone, the unary anchor, ZUPT and persistent variable. Having established the mathematical framework, these models must now be subjected to simulated testing to evaluate their effectiveness mitigating global drift. The next chapter details the experimental setup and simulation environment designed to

perform this evaluation.

5

Experimental Setup

To validate the mathematical models proposed in Chapter 4, a robust simulation environment was required. This chapter outlines the architecture of that experimental setup. First, Section 5.1 details the kinematic generation of the ground-truth trajectories and swarm formations. Section 5.2 then discusses the custom Python simulation environment used to execute the distributed GBP algorithm. Finally, Section 5.3 defines the baseline constants, parameter sweeps, and Key Performance Indicators (KPIs) used to measure the system's success.

5.1. Trajectory generation

To evaluate the performance of the proposed solution formulation of GBP on distributed factor graphs, a robust ground-truth generation framework was developed. This framework simulates multi-agent mission profiles with customizable trajectories/states, providing a noise-free baseline against which the Gaussian Belief Propagation (GBP) estimates are compared.

5.1.1. Kinematic integration

Unlike standard Euclidean integration which is used in the proposed solution, and can suffer from drift in rotational consistency over long durations, the trajectory generation utilizes the Special Euclidean Group $SE(2)$ for state propagation. The agent state is integrated using the exponential map, which maps a velocity twist $\xi = [\omega, v_x, v_y]^\top$ in the Lie algebra $\mathfrak{se}(2)$ to the manifold $SE(2)$:

$$T_{t+1} = T_t \cdot \exp(\xi \Delta t) \tag{5.1}$$

where $T \in SE(2)$ represents the rigid-body transformation of the UAV. This approach ensures that the resulting trajectories are kinematically feasible and that the heading (θ) remains naturally wrapped within the $[-\pi, \pi]$ interval without the need for manual clipping during the generation phase.

5.1.2. Leader-follower group formations

The simulation supports coordinated movement of the system through a leader-follower structure, as shown in Figure 5.1. A designated leader agent follows a predefined path, for example a constant-curve turn or a straight-line sprint. Then, $N-1$ follower agents maintain a specific geometric formation around the leader:

- Geometric formations: The framework supports arbitrary formation shapes, including V-shape, lattice, and diamond configurations. These are defined by local coordinate offsets $\mathbf{x}_{off}$ relative to the leader's frame.
- Rotation-invariant offsets: To maintain formation during turns, local offsets are rotated into the global frame using the leader's current heading θ_L:

$$\mathbf{x}_{des} = \mathbf{x}_L + \mathbf{R}(\theta_L)\mathbf{x}_{off},\ where\ \mathbf{R}(\theta) = \begin{bmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{bmatrix} \tag{5.2}$$

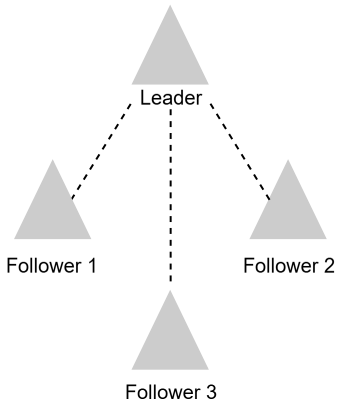


Figure 5.1: Visualization of simple leader-follower formation. Followers use proportional control to keep a pre-determined distance to the leader.

- Follower Control: Follower agents utilize proportional control to minimize the error between their current position and their designated formation slot, integrated via the same Lie group kinematics to maintain swarm consistency.
- Velocity constrained: The drones are given a maximum possible velocity, to adhere to realistic constraints. If the proposed control step exceeds this velocity, the maximum velocity is taken instead. This simulates the movement and control of low-cost drones realistically, accounting for skewing of the formation over turns and long-range trajectories.
- Initial heading: Each drone initially has the exact same heading. This provides a realistic starting scenario where followers still need to settle into the formation. This leads to ripple in the heading at the start, as can be observed from Figure 5.2b, and is an important test of whether the algorithm can deal with the start-up conditions of the system.

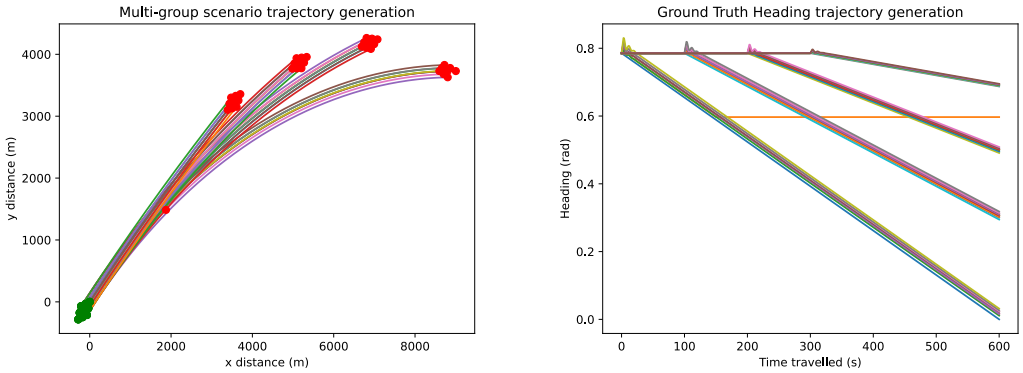
5.1.3. Multi-group and landing scenarios

To simulate the delayed spawn and stationary anchor concepts central to this thesis, the script allows for the generation of multiple independent groups. Each group is assigned a starting timestep, allowing for the simulation of secondary and consequent groups of drones passing over the landed drone out of a primary group. Furthermore, the framework implements landing logic in which specific drones can be transitioned to a stationary state in a predefined timestep t_{land}. Once landed, the agent's velocity twist ξ is set to zero, and its state remains constant for the remainder of the simulation. This transition provides the mobile anchor baseline used to investigate marginal improvements in absolute localization performance.

5.2. Simulation environment

To evaluate the proposed landing strategies, a custom discrete-time kinematic simulation environment was developed in Python. The architecture is highly modular, primarily divided into a global Scenario controller and localized Drone agents, mimicking the distributed nature of the target multi-agent system (MAS).

- Global Scenario Controller: Manages the true global clock, the ground-truth trajectories (generated mathematically or pulled from datasets), and the orchestration of Monte Carlo batches. It strictly acts as an evaluation oracle; agents do not have access to global states during optimization.
- Agent Initialization (Delayed Spawning): To accurately model complex mission profiles (e.g., subsequent UAV groups joining a mission later or passing over landed drones), the environment supports delayed spawning. Agents maintain an internal clock that only advances once they reach their designated start_timestep, preventing artificial dilution of accumulated error metrics.
- UI: An optional visual User Interface (UI) was developed, to aid rapid prototyping and visual iden-



(a) generated trajectories plotted over distance

(b) generated headings plotted over time

Figure 5.2: These plots show the ground truth trajectory and heading generation for a configuration with 4 groups of 9 drones flying in a diamond formation, where the first group lands a drone at $t = 150, [x, y] \approx [2000, 1600]$. Each drone flies according to a constant velocity model, leading to a small distortion of the diamond pattern over the curved trajectory. Each drone is initialised with the same heading, leading to ripple in the heading plot at the start when the drones fly into formation.

tification of phenomena. It includes all kinds of diagnostic plotting tools, a visual of the current estimates compared to ground truth and dead-reckoning, connectivity graph and live error tracking, as shown in Figure 5.3. This UI can also be disabled, enabling fast numerical simulation for Monte Carlo (MC) batch runs.

- Dataset: The simulation environment provides the option to use datasets from .npz files, making it a modular approach and able to accept any dataset. It also includes the possibility to run the specific UTIAS MR CLAM dataset [64], for verification of the design on real-world data.

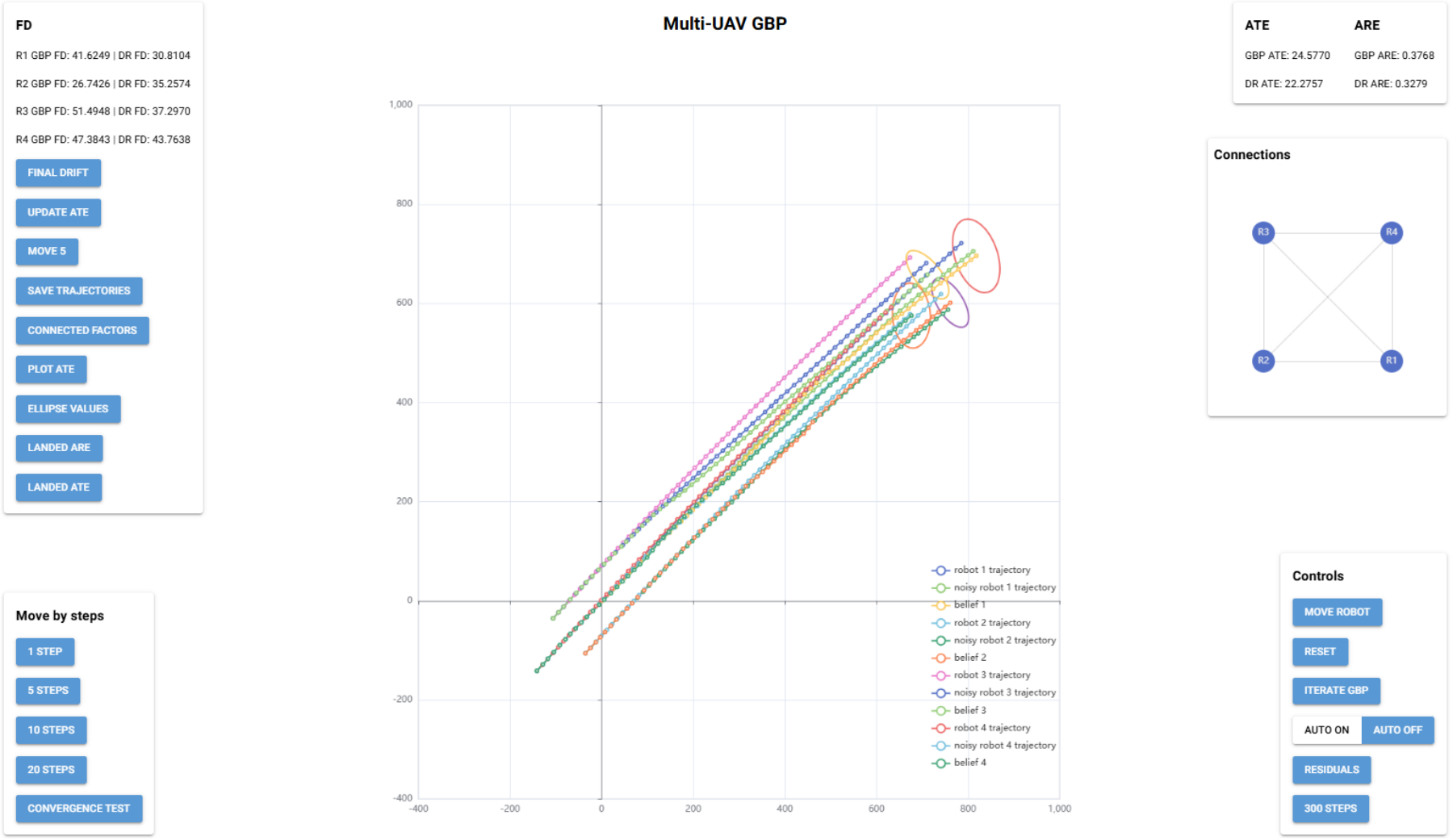


Figure 5.3: Graphical User Interface for the simulation environment. The graph in the center tracks the ground truth trajectories, dead-reckoning trajectories and GBP belief trajectories, with covariance ellipse plotted for the most recent variable's belief. The top right tracks the overall ATE and ARE live, below that a graph showing the active communication connections. The bottom right window and left windows provide buttons to trigger movement, algorithm iteration and plotting.

5.3. Parameters & associated KPI's

Now in order to simulate, the parameters that are constant and the parameters that are variable need to be defined.
The constant parameters stem from the operational scenario, with low-cost hardware and quadcopter drones, leading to the high noise values for IMU and UWB ranging as in Table 5.1. Furthermore, the quadcopter drones define the nominal achievable flight distance and achievable flight velocity. The factor graph rate is limited to 1 Hz, to simulate a resource-constrained system that cannot process a factor graph representation at high frequency. Now, the parameters that are interesting to sweep are

Table 5.1: Simulation constants

Constant	Value	Unit
Nominal flight distance	10	kilometers
Nominal flight velocity	16.7	m/s
Sampling rate of factor graph	1	Hz
Initial position uncertainty	0.1	m
IMU translation noise (σ_x)	0.1	m / m travelled
IMU rotation noise (σ_θ)	0.5	degrees
UWB ranging range base noise	1	m
UWB ranging range dependent noise (σ_r)	0.08	m / m distance
UWB ranging bearing noise (σ_b)	5	degrees

related to the landed anchors. A landed anchor is only interesting intermittently, therefore defined as dropping only 1-5 times over the full flight distance of 10km. The number of drones one lands is directly related to the number of drones in the full MAS, as this presents the trade-off between MAS effectivity and size of the sacrifice. The inter-drone distance has to adhere to a minimum that will not let the drones crash into each other, and its maximum is limited by the communication & sensing module, as is the communication range. The communication rate will provide insight into the minimum required **external** communication bandwidth for the MAS to function. The geometry of the formation is also explored, as we want to verify that e.g. parallelism in the geometry does not impact the performance negatively. This is all shown comprehensively in Table 5.2.

Table 5.2: Simulation variables

Variable/Parameter	Baseline	Variation range	Associated Evaluation
Landed anchors	0 anchors	1-5 anchors	Mitigating Final Drift (FD)
MAS swarm size	4 UAVs	2-100 UAVs	Scaling limits and graph dimensionality
# of distinct groups in MAS	1	1-10	Effect of MAS topology on landed anchor effectivity
Inter-drone distance	100 m	50 - 2000 m	Impact of ranging geometry on ATE
Communication range	1000 m	50 - 1000 m	Impact of limited communication on ATE
Communication rate	Unlimited	max. 1 other agent per step - unlimited	Impact of limited communication on ATE
Formation of MAS	Diamond	Diamond, V, lattice	Impact of MAS geometry on ATE

5.4. Concluding remarks

This chapter outlined the comprehensive experimental setup required to validate the proposed GBP architecture. By defining a kinematic trajectory generator and establishing a custom, discrete-time Python simulation environment, the conditions for multi-agent flight simulation have been set. Furthermore, the baseline sensor noise parameters and operational variables have been set to reflect the realities of low-cost hardware. With the simulation environment and parameters fully defined, the next chapter will systematically evaluate the system's performance.

6 Evaluation

6.1. Introduction

This chapter presents the quantitative evaluation of the proposed Gaussian Belief Propagation (GBP) cooperative localisation system. Using the custom discrete-time simulation environment explained in Chapter 5, a series of Monte Carlo experiments were conducted to isolate and analyze the effects of deploying stationary anchors within a moving multi-agent system (MAS).

The performance of the system is evaluated against the primary key performance indicators established in Section 4.1. Final Drift (FD) to measure ultimate mission success, Absolute Trajectory Error (ATE) and Absolute Rotation Error (ARE) to quantify overall path fidelity, and Communication Load (CL) to ensure viable low-bandwidth solutions on cost-limited hardware.

The evaluation systematically progresses from baseline dead-reckoning to cooperative but anchor-free operation, to validating the mathematical formulation of a single landed drone, and finally evaluating the performance scaling and communication overhead of multi-anchor deployments.

All shown results are averaged over 50 monte carlo realisations, unless stated otherwise. The noise values used in the synthetic data are as stated in Section 5.3. The uncertainties passed to the GBP factors are based on the same noise values. When noise values scale with the magnitude of the inputs (e.g. range-dependent range noise), the uncertainty values passed to GBP are scaled with the measurements. So, range-dependent range uncertainty is scaled with the noisy range measurement.

6.2. Baseline anchor-free performance

Before evaluating the impact of stationary anchors, it is necessary to establish baseline performance under purely relative, anchor-free conditions.

Two distinct configurations were tested:

1. Dead-reckoning: Agents rely solely on the integration of noisy IMU measurements, simulating a complete absence of inter-agent communication or outside information.
2. Standard Cooperative Localization: Agents perform distributed GBP using both IMU measurements and inter-agent range-bearing measurements, but without any absolute reference points beyond the initial pose.

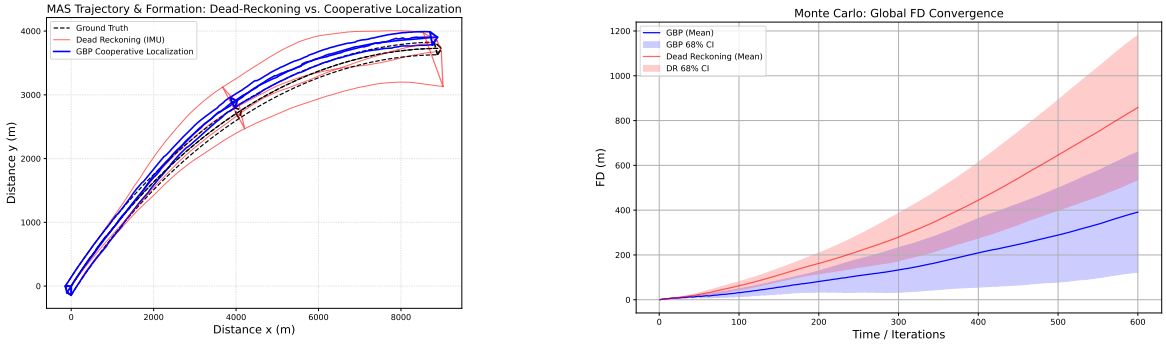
The configurations were tested first on synthetic data with Gaussian noise, as detailed in Section 4.4 and Section 5.1. After success on synthetic data, to verify the simulation and the applicability to real-world sensor data, the performance was also evaluated on the UTIAS MR.CLAM real-world dataset with real-world noise [64].

6.2.1. Dead-reckoning vs cooperative localization on synthetic dataset

Figure 6.1a shows that dead-reckoning integration quickly leads to position errors in the order of hundreds of meters, and the loss of the shape of the relative formation of the system.
On the other hand, CL using GBP is effective in keeping the relative formation tight. On top of that, as can be seen in the right figure, CL reduces the mean FD to half of the value of the dead-reckoning solution.
This result is in line with theory, as in optimal multi-sensor data fusion, combining measurements from N independent, unbiased sensors reduces the estimator's variance to $\frac{\sigma^2}{N}$, meaning the standard deviation of the estimator is reduced by a factor of $\sqrt{N}$.
Since each of the drones follows a random walk pattern, with the error in $X \sim \mathcal{N}(0, \sigma_x)$ and error in $Y \sim \mathcal{N}(0, \sigma_y)$, the $FD \sim \sqrt{X^2 + Y^2}$ follows a Rayleigh distribution:

$$Mean\ FD = \sigma \cdot \sqrt{\frac{\pi}{2}} \tag{6.1}$$

Therefore, a system with N = 4 drones, should lead to a two-fold reduction in mean FD: $\frac{\sigma}{\sqrt{N}} = \frac{\sigma}{2}$, as shown in Figure 6.1b.



(a) (b)

Figure 6.1: Comparison of dead-reckoning localization performance versus cooperative localization. (Left) comparison of ground thuth, DR and GBP CL trajectories for one particular realization of the experiment. (Right) Comparison of mean Final Drift (FD) over time of DR versus CL for 4 drones flying in formation.

The ARE shows the same trend as the ATE and is therefore included in Section B.1.
This baseline testing also evaluated the impact of flight formation (diamond, lattice, and V-shape) on localization performance. The results indicate that formation geometry has a negligible impact on the relative improvement of the system. Across all three shapes, the ATE ratio remained stable between 2.31 and 2.44, while the Final Drift ratio remained bounded between 2.05 and 2.20. Consequently, all subsequent evaluations were conducted using the diamond formation.
The effect of increased ATE, ARE and FD ratio scales in the same way for larger group of drones, with ATE and ARE averaged over 50 monte carlo runs. Moreover, as the number of drones increases, the ratio of ATE to DR increases more rapidly.
Now the standard discrete-time kinematic equations for the drones used are:

$$\begin{aligned} \theta_t &= \theta_{t-1} + \omega_t \Delta t \\ x_t &= x_{t-1} + v_t \Delta t\ cos(\theta_{t-1}) \\ y_t &= y_{t-1} + v_t \Delta t\ sin(\theta_{t-1}) \end{aligned} \tag{6.2}$$

After first order Taylor expansion, and uncertainty propagation laws, for which the derivation is included in Section A.3, this implies that for the covariance propagation, the equations are:

$$\begin{aligned} P_{\theta\theta,t} &= P_{\theta\theta,t-1} + Q_{\theta\theta} \\ P_{xx,t} &= P_{xx,t-1} - 2(v_t \Delta t \sin(\theta_{t-1})) P_{\theta x} + (-v_t \Delta t \sin(\theta_{t-1}))^2 P_{\theta\theta} + Q_{xx} \\ P_{yy,t} &= P_{yy,t-1} + 2(v_t \Delta t \cos(\theta_{t-1})) P_{\theta y} + (v_t \Delta t \cos(\theta_{t-1}))^2 P_{\theta\theta} + Q_{yy} \end{aligned} \tag{6.3}$$

Table 6.1: Ratio of GBP performance to DR performance for different numbers of drones flying in diamond formation, after 600 timesteps.

Number of drones	ATE ratio	ARE ratio	FD ratio
4	2.44	2.06	2.20
5	2.46	2.31	2.17
6	2.67	2.57	2.46
7	3.14	2.89	2.87
8	3.19	3.03	2.90
9	3.78	3.40	3.48
10	4.25	3.74	3.95

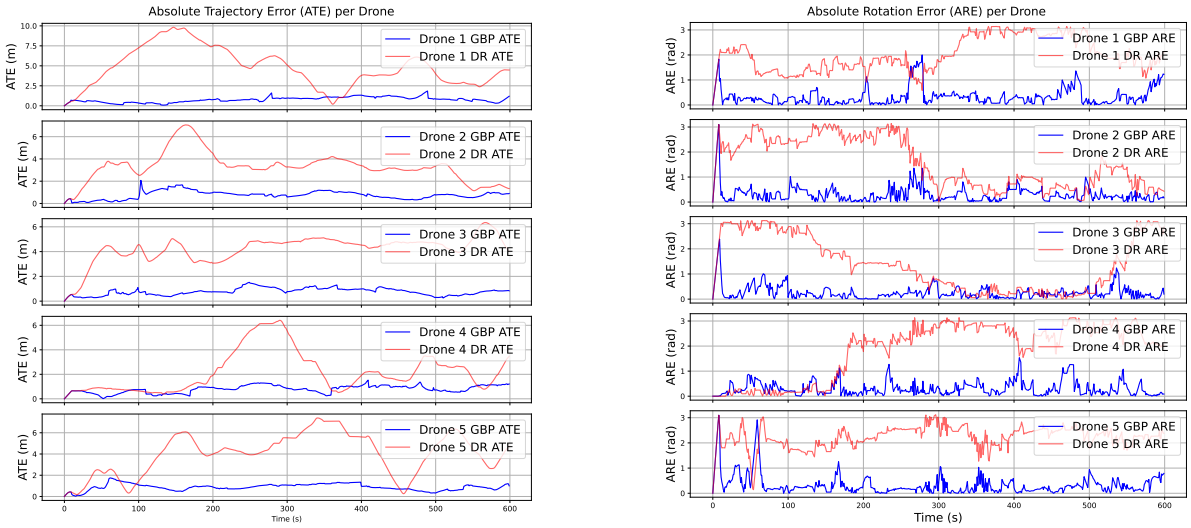
These equation in Equation 6.3 show that a decrease in the covariance of the heading already leads to a decrease in the covariance on the position. This is even without taking into account the additional benefit of averaging the position error. This explains the results in the table, where the ATE is reduced by a larger factor than the ARE. As the ARE, governed by $P_{\theta\theta,t}$ already decreases with the statistical $\sqrt{N}$, the ATE, governed by $P_{xx,t}$ and $P_{yy,t}$ decreases more rapidly, profiting from the decreased coupling term of $P_{\theta\theta}$.
On top of that, the ARE shows a trend of performing consistently better than the $\sqrt{N}$. This is a consequence of the factor graph holding states for multiple timesteps in the optimization window, and thus being able to back-correct older states with new information.

6.2.2. Dead-reckoning vs cooperative localization on UTIAS MR.CLAM dataset

To verify our approach on real-world data, localisation was performed on the UTIAS MR.CLAM dataset [64]. The data set provides five ground robots that drive in random patterns through a space, making range-bearing measurements of each other as they move. The dataset also provides landmarks, but these were **not** used for the results for the following. The solution was tested on UTIAS datasets 1 through 4, dataset 1 is shown in Figure 6.2. Datasets 2 through 4 show the same trends and are therefore included in Subsection B.1.1.
The connection of the robots through range-bearing measurements in this information rich scenario with many turns and varying formations through random pattern driving is effective in keeping the ATE lower than dead-reckoning, as can be seen in Figure 6.2a. The plots show the dead-reckoning having a much larger ATE as well as a much larger spread of the error. On the other hand, the GBP solution effectively smoothes out the trajectory, keeping ATE at a steady low value.
The ARE plot in Figure 6.2b shows a large spike in error at the start. The robots start out accelerating from stationary position. This reflects how dead-reckoning struggles when dealing with orientation drift, especially when robots are barely moving, yielding no additional information to correct for the orientation estimates. We see that once the first range-bearing measurement comes in, the GBP quickly back-corrects the orientation, reducing the ARE, which is calculated over the whole past trajectory, drastically.

6.2.3. Study on convergence

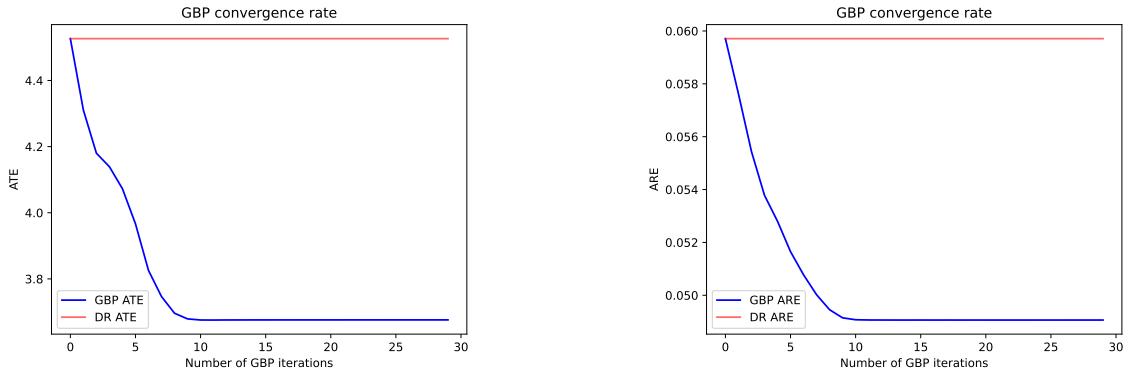
Now an important measure for the effectiveness and efficiency of the algorithm is of course convergence. Namely, the convergence of GBP is not guaranteed with loops in the system, so it needs to be empirically verified. Furthermore, the number of iterations until convergence indicate the computational efficiency of the algorithm.
In the proposed setup, the convergence is especially interesting on steps where range-bearing factors are added to the graph. In other cases, consensus is reached within 1 timestep, as the dead-reckoning measurement is propagated to the next variable node.
However, when two drones connect via the range-bearing factor, they have to negotiate on what their respective histories look like, given the new relative constraints. Figure 6.3 shows the ATE and ARE convergence of a scenario with 4 drones flying, at their first inter-drone connections. The plots show a clear asymptotic trend where the errors reach their minimum value after 10 iterations. To ensure that



(a) ATE **(b)** ARE

Figure 6.2: A comparison of the ATE and ARE of UTIAS MR.CLAM dataset 1 under dead-reckoning vs GBP cooperative localization without any landmark usage.

the algorithm has always converged before the next measurement factor and state variable is added to the graph, a standard amount of 15 iterations will be run for all experiments from here on out. Datasets



(a) ATE **(b)** ARE

Figure 6.3: Convergence of the ATE and ARE using GBP with time-window of 10 steps, evaluated at a step where inter-drone range-bearing measurements are performed.

2-4 show the same trends and are included in Subsection B.1.1 for completeness sake.

6.3. Evaluation of landed anchor models

Having established a baseline performance of the algorithm, we now move on to explore the different options to model the landed drone in the factor graph.

6.3.1. Unary anchor factor

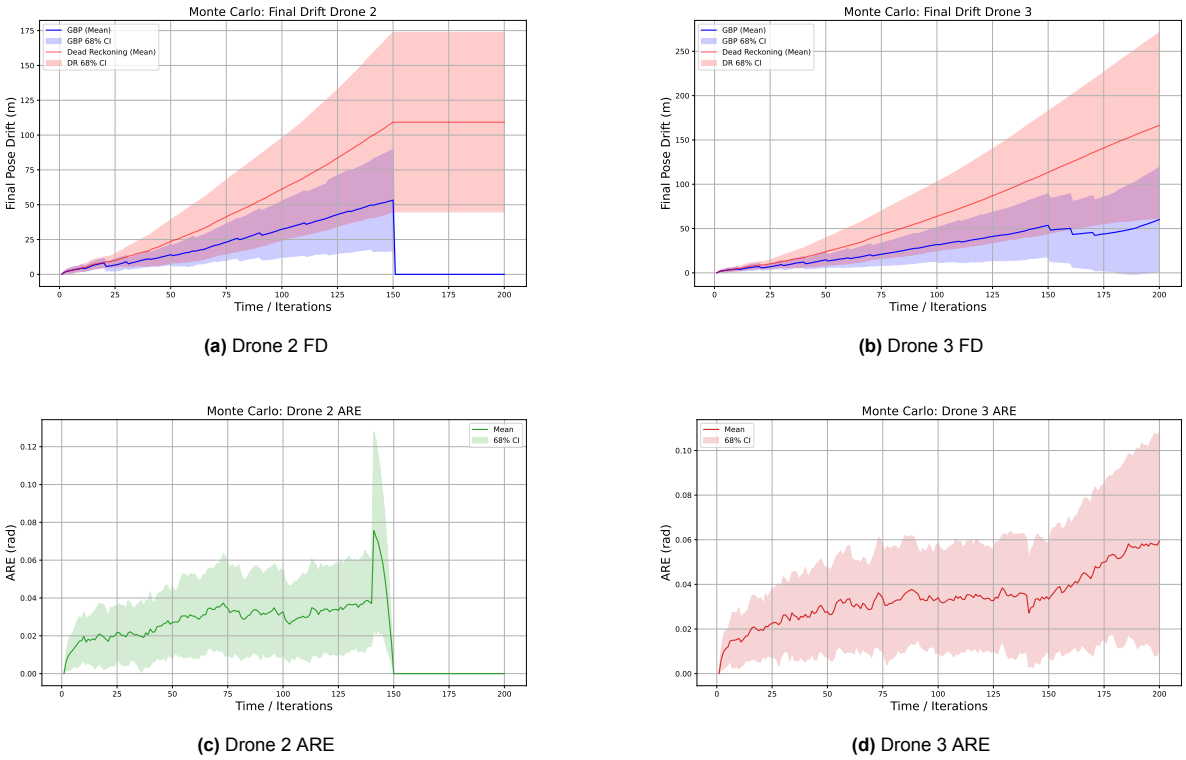
The unary anchor factor constrains the drone to a single absolute location. The hypothesis is that when the drone lands, its estimate does not improve anymore, and the performance will be directly related to the accumulated drift of the landed drone.

Ideal scenario

To test this hypothesis, the experiment is first tested in a ”signal of opportunity” scenario, injecting ground truth information into the landed drone with low uncertainty, to verify that this yields increased

localisation accuracy of all drones. Figure 6.4 shows the FD of the drone that gets the absolute information (drone 2) drops to 0. Furthermore, the other 3 drones in the system decrease their error at t=[150,160,170,180] via the range-bearing factors, as a consequence of the increased accuracy of drone 2.

However, a second order effect occurs in case of a large update such as drone 2 receiving absolute



(a) Drone 2 FD

(b) Drone 3 FD

(c) Drone 2 ARE

(d) Drone 3 ARE

Figure 6.4: ATE for drones in the unary anchor factor model with absolute information injected. Drone 1 and 4 are omitted, as they show the same trend as Drone 3.

information. GBP is inherently a smoothing algorithm, so it will attempt to translate a large position update (due to absolute information) at a single timestep, into small updates over all past timesteps. In this way, the algorithm back propagates an update through its belief of the trajectory, leading to a reduced error over the whole trajectory.

However, in case of a large update in combination with a narrow time window, or too few iterations to converge, this will lead to a spike/discontinuity in the error, and a worse long-term prediction as a result. The algorithm attempts to reconcile the large update by twisting the heading by a small amount at each variable node, leading to a steeper increase in heading error between t=150 and t=180 in Figure 6.4d. In turn, via the lever-arm effect, this increased heading error leads to worse long-term predictions.

This can be mitigated by using a larger time-window, in combination with more message passing iterations, as shown in Figure 6.5. Using a larger time-window, without increasing the number of message passing iterations does not yield a significant improvement, it is included in Subsection B.2.1.

Non-ideal scenario

Now, in the actual operational scenario, there is no signal of opportunity, and the best estimate of the drones' location is the current estimate. Thus, the next experiment tests the effect of a single landed drone using a unary anchor factor constraining it to the latest estimate of absolute position. Figure 6.6a shows the landed anchor drone stops drifting at t=150. However, the estimate does not improve anymore.

This yields a system where the performance of cooperative localization of the group depends directly on the accuracy of the location estimate of the landed drone, as this drone is given artificial certainty with the unary factor.

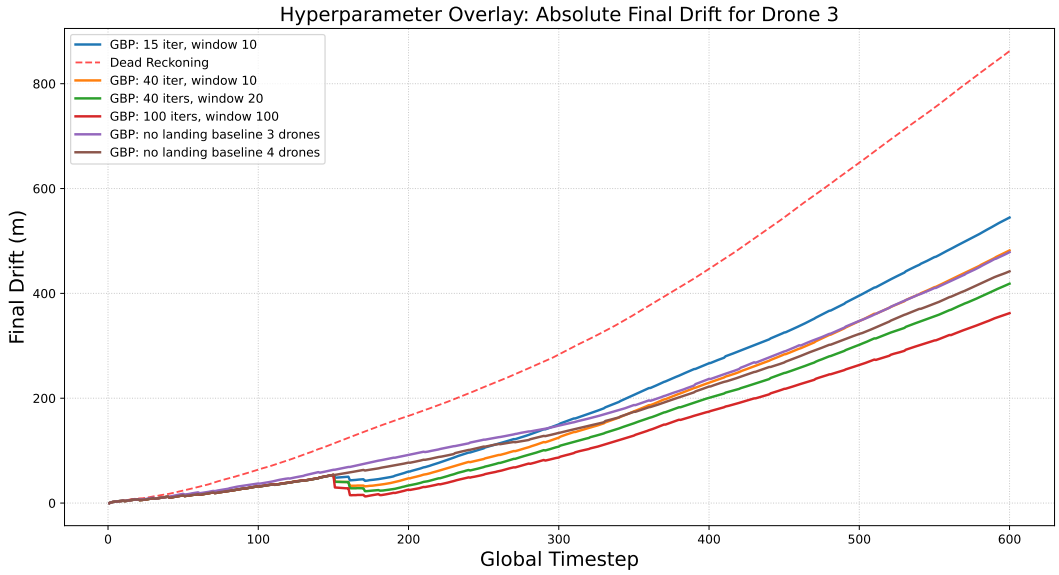


Figure 6.5: Final Drift characteristic under different number of iterations and time-window sizes for unary anchor model with absolute information.

6.3.2. ZUPT

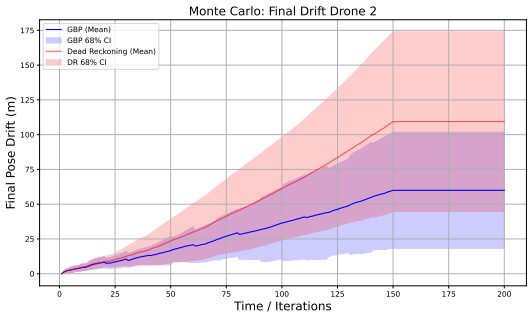
The zero-update factor only tells the landed anchor drone that it is not moving, without constraining it to an absolute position. The hypothesis would therefore be that this zero-update factor allows the landed anchor to improve its own location estimate and in turn relay this to the rest of the MAS.
Figure 6.6b illustrates how a zero-update with a certainty similar to that of the IMU factors can improve its estimate marginally at each 10 timesteps when relative measurements are present. Figure 6.6d Illustrates how the zero-update with much lower uncertainty (100x lower uncertainty), still allows for the convergence of the landed drone to a better estimate. However, as will be elaborated on in the next section, it performs worse than the zero-update with similar certainty in terms of covariance.
Although in the single-group topology, it does not meaningfully improve its estimate, nor that of the group, it shows promise in this manner as a solution in the multi-group scenario.

6.3.3. Persistent variable

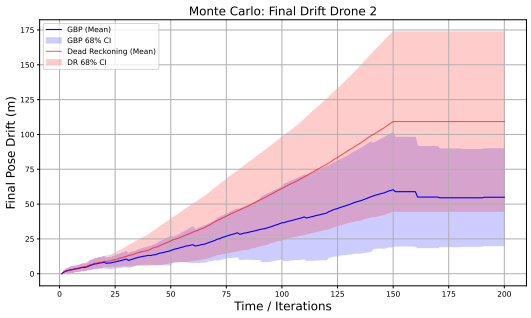
The persistent variable model for the landed anchor drone tries to leverage the fact that all independent measurements should lead to a reduction in the variance or error. Figure 6.6c shows how this is indeed the case. It shows a reduction in error just as the ZUPT at t=160 mainly. It is however clear for both models that the estimate does not improve significantly after that. This can be prescribed to the group reaching a mathematical consensus that preserves the relatively geometry, yet still drifts in the global frame. The main difference to the ZUPT is the behavior of the covariance matrix, which is explored in more detail in the following section.

6.4. Multi-group topologies

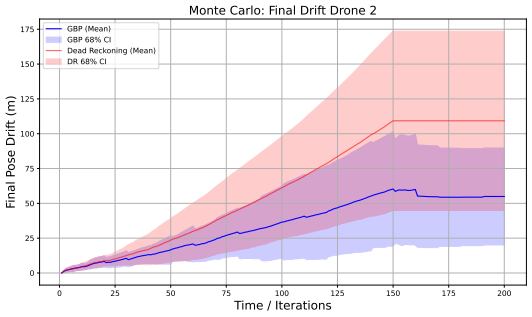
Since we can conclude from the previous that landing a drone does not improve the localisation within a single group, because the estimates of the group are correlated, we move on to a multi-group scenario. Here we test an operational scenario where a drone from the first group is landed, and see if it can improve its location estimate with the information of subsequent groups. In turn, we hypothesize a turning point where the landed drone goes from gathering new information from subsequent groups to improve its own estimate, to functioning as a rigid anchor and passing on its location certainty to the next group improving the group its estimate.
The experiment considers a scenario with groups of 4 drones, Each group starting 100 timesteps after the other and following the same path. The first group lands a drone at t=150. The spacing between the drones in a group is set to 100m, and the uncertainties are as defined in Table 5.1. The communication range is set to 600 meters, ensuring that within groups each drone observes the other, but different



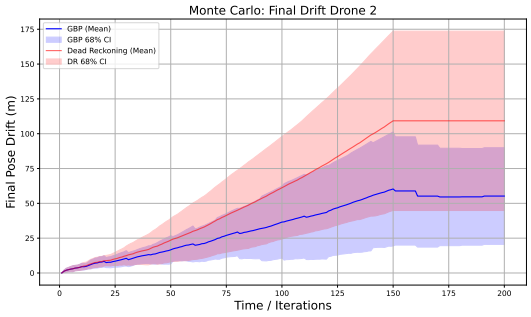
(a) Unary Anchor



(b) ZUPT $\sigma = [0.5^\circ, 0.1, 0.02]$



(c) Persistent Variable



(d) ZUPT $\sigma = [0.005^\circ, 0.001, 0.0002]$

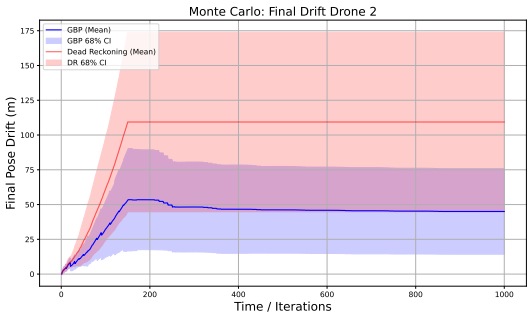
Figure 6.6: FD for landed drone under different models (unary anchor, ZUPT, persistent variable).

groups do not observe each other.

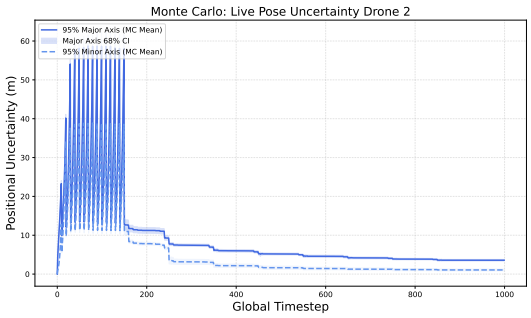
6.4.1. Persistent variable

Figure 6.7 illustrates the initial results for the described experiment, with the persistent variable model. The figure shows how the landed drone can reduce its drift, showing diminishing returns on this with each passing group, at t=220, 420, 620.

The covariance plot on the right shows the estimated 95% confidence interval, that first shows an oscillatory pattern. This pattern comes from the fact that the covariance grows at each timestep with new IMU factors added to the graph. Every 10 timesteps, the drones observe each other through range-bearing measurements, and the covariance shrinks drastically, leading to the sawtooth pattern. After t=150, the drone lands and the covariance only shrinks from that point onwards. Now even



(a) FD landed drone

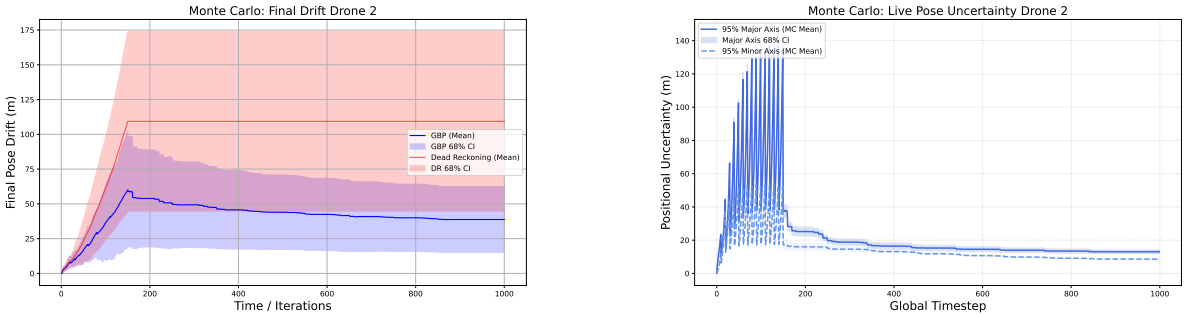


(b) Covariance landed drone half major and half minor axis

Figure 6.7: Final Drift and Covariance estimates for landed drone in persistent variable mode, using unlimited range-bearing measurements while in communication range.

though the error is not reduced by a significant amount, The daisy chain topology allows the landed drone to more accurately estimate its covariance. This in turn allows for the mean of the estimate to

slowly converge towards the ground truth as more groups, i.e. fully independent measurements, pass by. Figure 6.8a shows a reduced final error as compared to Figure 6.7a. In addition, the confidence interval of Figure 6.8a over the 50 monte carlo runs is more narrow by 19% than for Figure 6.7a.
These results are a consequence of the overconfidence that is created in the factor graph as a result of using GBP with range-bearing factors between drones in both directions. This topology results in small loops, that increase the ”rumor mill” effect. One variable sees back its own information via a short loop, believing however that it is independent information. This leads to the variable its precision and information falsely increasing.
Comparing the covariance plots in Figure 6.7b and Figure 6.8b, it is clear that the fully connected range-bearing measurements topology is more overconfident, believing at t=200 to have a 95% confidence major axis of 10m error, whilst it is clear from the final drift plot that the error ϵ at t=200 is $\epsilon \approx 50$m. Figure 6.8b is already estimating the covariance more accurately, believing at t=200 to have a 95% confidence major axis of 25m error. However, it provides no mechanism to scale up the covariance again to account for the overconfidence.



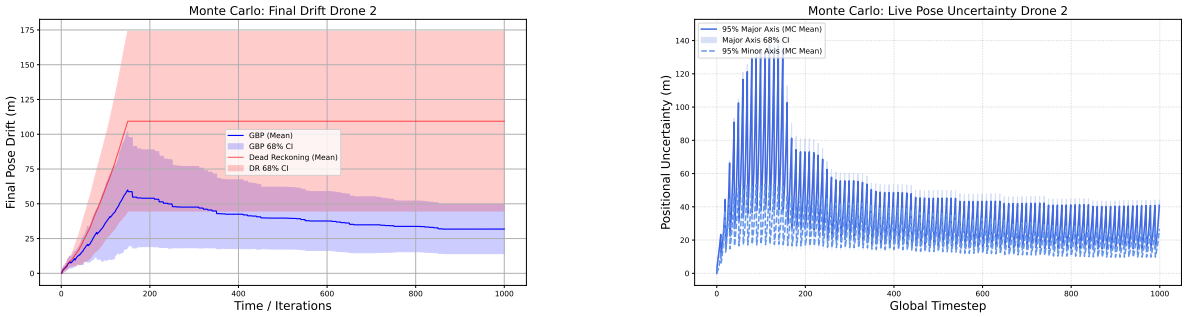
(a) FD landed drone

(b) Covariance landed drone half major and half minor axis

Figure 6.8: Final Drift and Covariance estimates for landed drone in persistent variable mode, using daisy chain limited range-bearing measurements while in communication range.

6.4.2. Zero-update model (ZUPT)

The zero-update model, in contrast with the persistent variable model, adds uncertainty to the landed drone at each timestep, by creating a new variable and factor in the graph, with a given uncertainty.
Although this presents the perceivingly disadvantageous characteristic of growing the covariance over time, as seen in Figure 6.9b this can work to the advantage of the solution in this particular case. Namely, it combats the overconfidence by artificially inflating the covariance. This is also seen in Figure 6.9a, which achieves an estimate for the landed drone that has both a lower mean value and a more narrow confidence interval than the results for the persistent variable model as presented in Figure 6.8.
Figure 6.10 Presents the drift at t=150, when passing over the landed drone, for each consecutive



(a) FD landed drone

(b) Covariance landed drone half major and half minor axis

Figure 6.9: Final Drift and Covariance estimates for landed drone using ZUPT model $\sigma = [0.5°, 0.1, 0.02]$, using daisy chain limited range-bearing measurements while in communication range.

group. This shows the short-term incremental accuracy gain for each group as the landed anchor

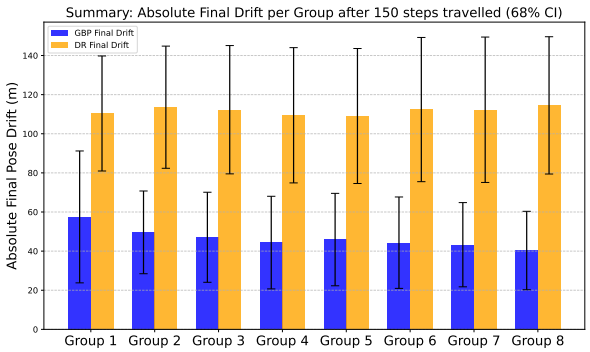


Figure 6.10: Barchart of Final Drift per group at t=150 (when flying over the landed drone). It shows a downward trend, indicating that as more groups fly over, and the landed drone localises better, it functions more and more to the benefit of the next group.

drone improves its own pose estimate under the ZUPT model.

6.5. Concluding remarks

This chapter systematically evaluated the proposed distributed GBP architecture across varying operational scenarios. The results first established that the baseline anchor-free GBP system successfully halves the mean Final Drift (FD) compared to pure dead-reckoning. Subsequently, evaluating single-group landing topologies revealed that purely relative measurements cannot mitigate global drift, as the network becomes highly overconfident. However, the evaluation of multi-group topologies demonstrated that this limitation can be overcome. Although the persistent variable model is only additive in information and thus presents limitations, by employing a ZUPT factor model, a landed drone can successfully combat graph overconfidence, act as a localized absolute reference for subsequent groups passing overhead, and systematically reduce the Final Drift of the overarching MAS. The mathematical phenomena driving these results are analyzed in the following Discussion.

7

Discussion

Before analyzing the effects of stationary anchors, the baseline performance of the system needs to be put into context. The results conclusively demonstrate that the implemented distributed Gaussian Belief Propagation (GBP) framework succesfully maintains relative swarm geometry and halves the mean Absolute Trajectory Error (ATE) compared to pure dead-reckoning. This confirms that the foundational message-passing architecture and the analytical linearizations of the factors are mathematically sound. By proving that the baseline system effectively leverages relative measurements to bound local uncertainty, we can be confident that the behaviors observed during the landing maneuvers are genuine algorithmic phenomena, rather than underlying software or integration errors.

In the results, it was extensively observed that the estimates for all drones grew overconfident with time, and especially so for the landed drone. Now it is known that for the distributed algorithm of GBP, overconfidence is an issue. Davison *et al.* [23] point out that overconfidence is generally greater for more inter-connected nodes, suggesting that the persistent variable node would become overconfident more quickly than other nodes, explaining the diminishing improvements on error with new groups flying over, which were even more pronounced than expected. However, Murai *et al.* [44] also claim that since GBP converges to the exact marginal means upon convergence, this implies that relative precisions of local messages are convergently correct. This explains the fact that the overconfidence did not pose concerns at first in the single-group scenario.
Nonetheless, short loops also lead to overconfidence. The information of one node comes back to the same node via a short loop, as visualised in Figure 7.1. While the node assumes it receives independent new information, it is actually receiving information from itself unknowingly via the loop, increasing its precision and shrinking its covariance while it should not. Different mitigations were proposed to this

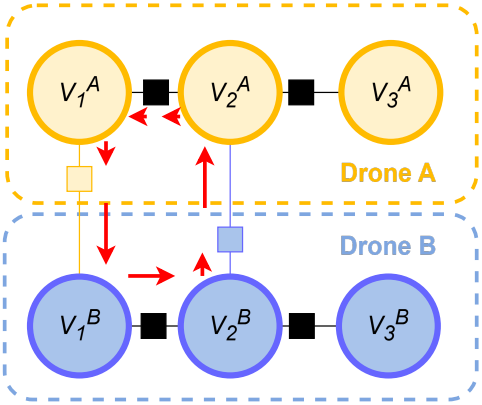


Figure 7.1: Illustration of the rumor mill effect. Information flows back to a variable via a short loop, while it is assumed to be independent, leading to overconfidence.

overconfidence:

- Reducing the frequency of range-bearing factors, in order to create longer temporal loops so the effect diminishes.
- Reducing the number of range-bearing factors created at a single timestep.
- Controlling the directionality of range-bearing measurements. It is assumed that a range-bearing factor constrains the heading of the observing drone, but not of the observed drone, rendering the directionality significant.
- Restricting the range-bearing factors to only connect in a daisy chain fashion between drones. 1 connects to 2, 2 connects to 3, 3 to 4, but 4 not back to 1. This prevents the creation of loops.
- Increasing the sliding time-window.

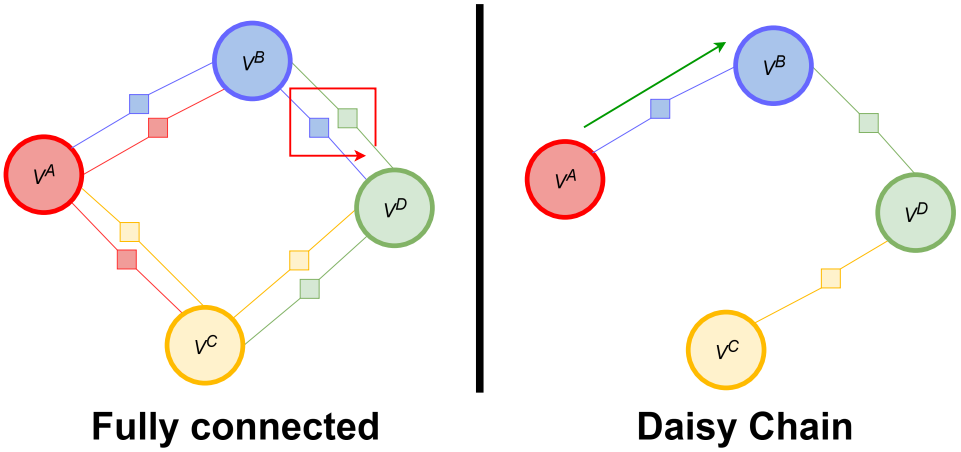


Figure 7.2: Comparison of factor graph network topologies. [Left] a fully connected graph, which creates short loops, exacerbating the overconfidence effect. [Right] The daisy chain restricts range-bearing measurements to a directional flow A<-B<-C<-D, not creating any factor loops.

Reducing the frequency of range-bearing factors, in conjunction with daisy-chaining the factors, diluted the overconfidence and made the solution viable. Since daisy-chaining means only 1 range-bearing factor is created between two variables at a timestep, it becomes significant to ensure that in the next iteration a range-bearing factor is created in the opposite direction. Otherwise, only the heading of one of the two drones is constrained by the rest of the factor graph. The use of a distributed algorithm was a constraint from the operational scenario. Having a single point of failure is not an acceptable configuration for UAV MAS under adversarial conditions. However, the use of a distributed solution inherently complicates consistent estimation, thereby exacerbating this issue of overconfidence.
For example the low-cost/ limited hardware assumption, makes it infeasible to track the entire factor graph. However, marginalizing out old states forgets inter-drone factors, not properly accounting for cross-correlations between the drone's state estimates.
Next to that, flying for extended time without any absolute reference information induces non-linear drift, that is only approximated when representing it as a Gaussian distribution. While the real covariance would resemble a "banana" shape, due to the lever-arm effect of the heading error, the assumption of Gaussianity will try to fit it into an ellipse. This is another cause of overconfidence in GBP.

This tension between physical reality and mathematical stability also dictates the limits of the sensor models used in this evaluation. In physical UWB sensors, far-field angular resolution degrades as distance increases due to dropping Signal-to-Noise Ratios. However, to prevent the GBP precision matrices from collapsing at long distances, bearing noise was modeled as a constant. While this compromise aids algorithmic stability, it means the simulated system maintains artificially high geometric rigidity at longer distance. A critical consequence of this is that it causes a disbalance in the GBP solver at long range. The bearing uncertainty is artificially small, causing the bearing measurement to mathematically dominate and skew the heading disproportionately. As dead-reckoning relies on heading

integration, this causes long-term position drift via the lever-arm effect. The inherent vulnerability of the GBP solver to precision disbalance remains a fundamental limitation of simplified noise models.

The inherent overconfidence explains why the Persistent Variable model, despite its theoretical appeal as an information hub, underperformed in mitigating global drift. By fusing correlated messages into a single static variable, it became highly overconfident. Conversely, the ZUPT model proved to be an effective, albeit artificial, mitigation strategy. By continuously injecting a small amount of uncertainty at each timestep through the zero-velocity constraint, it acted as a covariance inflation mechanism, counteracting the rumor-mill effect and allowing the landed drone to more accurately converge towards the ground truth during multi-group scenarios.

Now the results imply that landing a drone out of a group that is already performing cooperative localisation yields little to no improvement in Final Drift performance. Consequently, the operational recommendation would be to fly the mission with the entire group, without performing landing maneuvers.

However, in the multi-group scenario, it is observed that the landed drone can slowly improve its estimate towards ground truth. In large multi-batch scenarios it could therefore be operationally beneficial to land a drone out of the first group, make it converge to a small error as subsequent groups fly over, and in the end function as a stationary anchor. As the results show in Figure 6.5, once the landed drone has a near ground-truth estimate, it can successfully reset the drift of other drones to near zero, rendering them a lower Final Drift by the end of the flight. Ultimately, the findings discussed in this chapter highlight the inherent tension in distributed cooperative localisation. While factor graphs with distributed algorithms provide a scalable architecture, accounting for cross-correlations is challenging in these distributed algorithms, and the overconfidence that occurs as a result fundamentally limits single-group deployments. It is only by recognizing this limitation and shifting to multi-group deployment that the overconfidence is mitigated. Having established these operational realities, the following chapter synthesizes these discoveries to formally answer the core research questions of this thesis.

8

Conclusion

This thesis has explored whether landing a UAV out of a Multi-agent Systems (MAS) can improve the location estimate of the rest of the MAS, in order to improve mission success. A simulation framework was proposed in order to explore distributed cooperative localization for drones under adversarial conditions, under varying parameters and operational scenarios. The framework combines Gaussian Belief Propagation algorithm on factor graphs in Python with monte carlo simulations to evaluate proposed scenarios. The following sections describe the main findings and discuss directions for future work.

8.1. Research Questions

1. What representation and algorithm can be used to model the MAS dynamics and measurements, and why is this choice appropriate for the problem and sensors considered?

To address this question, an elaborate study into the state-of-the-art literature on cooperative localization was performed.
This thesis utilizes factor graphs as a representation of the state and measurements in the system. Factor graphs provide a way to keep track of the state and measurement histories, and optimize over them jointly. Factor graphs provide the best fit for the scenario, as they allow one to back-correct the state history when a signal of opportunity, i.e. new and improved information on position, presents itself. Additionally, factor graphs have been split up to be used in a distributed setting by others before as well.
The algorithm of choice was Gaussian Belief Propagation (GBP). The algorithm is naturally distributed, performing inference via local message passing between the nodes of the factor graph. Furthermore, GBP has shown promise in GNSS-denied navigation, handling asynchronicity and communication failures well. In addition, the parametrised representation of all nodes as Gaussian distributions keeps computation and communication streamlined.
The distributed algorithm presents challenges, complicating accounting for cross-covariances, leading to overconfidence in the estimates. This in turn limits the performance of landing a drone in single-group scenarios. Nevertheless, it presents an accurate representation of the system, satisfies the operational constraint of distributedness and can lead to improvements in multi-group scenarios.

2. What is the improvement in absolute localization accuracy as the number of landed drones (N) increases, and at what point do these performance gains diminish?

In a single-group scenario, where all drones fly in formation within communication range of each other, the landed drone drone does not lead to any performance gains when compared to cooperative localization without any landing/stationary anchor. As the group is connected through relative measurements, they reach a consensus and the group drifts as a rigid body. The landed drone in turn cannot improve its estimate beyond the average error of the group, and does not add any performance gains to the group. Rather, by landing a drone in this single-group scenario, it leaves communication range with the rest of the group, which with a reduced number of drones in the group, will drift more quickly.

On the other hand, in the multi-group scenario, the landed drone can asymptotically improve its location estimate. In turn, it can limit the drift of subsequent groups flying over. However, this error reduction takes effect slowly over multiple groups, rendering it only useful in a large batch operational scenario.

3. Given specific mission constraints (MAS size, sensor noise levels, and graph connectivity), what is the minimum number of landed drones required to guarantee a maximum allowable Final Drift threshold?

The results indicate that the minimum required number of landed drones is fundamentally dictated by the deployment topology rather than the sheer size of the MAS. In a single-group, continuous flight topology, no feasible number of landed drones (N) can guarantee a maximum allowable Final Drift, because the entire graph becomes overconfident and drifts as a rigid body. However, when adopting a multi-group deployment topology, the required number of landed drones reduces. Utilizing the Zero-velocity update (ZUPT) anchor model to counteract graph overconfidence, a single landed drone ($N = 1$) can act as a localized absolute reference point for subsequent groups passing overhead. However, this requires it first to converge its estimate to 0 as subsequent groups fly over. Therefore, to guarantee a specific maximum Final Drift threshold (T) over a long-range mission, the minimum number of landed drones required is one per distance interval where the baseline cooperative dead-reckoning drift approaches T. By staging single ZUPT anchors at these specific distance intervals, a mission of any length or MAS size can theoretically bound its global drift while sacrificing a minimal number of agents.

All the answers to the subquestions above contribute to addressing the main research question, that was outlined in the introduction of this thesis:

> *To what extent does landing drones in a MAS improve absolute localization performance when performing cooperative localization, under low-cost IMU and relative range and bearing measurements?*

The synthesis of the preceding sub-questions gives the definitive answer to this thesis's primary research question: Deploying a landed drone does not universally improve absolute localization. Rather, its effectiveness is strictly dictated by the interplay between the MAS its operational flight topology and the algorithmic formulation of the anchor.

In single-group, continuous flight scenarios, landing a drone provides no measurable reduction in Final Drift (FD) for the overarching MAS. Because purely relative range and bearing measurements lack absolute observability, the swarm merely reaches a highly rigid consensus while continuing to drift globally. Furthermore, the inherent "rumor mill" effect of distributed Gaussian Belief Propagation (GBP) causes the network to become overconfident, preventing the stationary drone from actively improving its own drifted estimate.

However, absolute localization is successfully improved when the landing strategy is applied to multi-group missions. By landing a drone from an initial group, it serves as a stationary anchor for subsequent, independent batches of UAVs. Crucially, this thesis proved that to leverage this anchor effectively, the system must utilize a ZUPT factor model. By injecting a controlled amount of uncertainty at each timestep, the ZUPT model acts as an algorithmic covariance inflation mechanism. This successfully combats the overconfidence of the GBP algorithm, allowing the landed anchor to absorb independent geometric constraints from passing groups and asymptotically converge toward the ground truth. Once converged, this anchor limits the accumulated drift of all subsequent UAVs.

Ultimately, under realistic low-cost sensor constraints, deliberately landing drones is a tactical trade-off. It is only effective provided it is utilized to incrementally build stationary infrastructure across multi-wave missions rather than to correct a continuous, single-wave swarm.

8.2. Future work

Challenges remain that should be addressed in further research, to enhance the effectiveness of the proposed method. The following directions are recommended for future work:

- **Address the overconfidence of GBP in more detail**
  Overconfidence in the state estimates is one of the major reasons that the multi-group strategies discussed in this thesis do not yield the intended performance. Overconfidence is a known shortcoming of loopy BP [23], but it is interesting to see what other choices, next to the ones already explored, could still be made to mitigate this issue as best as possible.
- **Dynamic and range-dependent sensor models** This thesis used a conservative, constant noise model for the bearing, to guarantee mathematical stability of GBP. However, real UWB sensing suffers from signal-to-noise ratio degradation in the far-field, causing angular uncertainty to increase. Future research should focus on integrating these true physical noise characteristics into the factor graph without inducing algorithmic divergence.
- **Implementing GBP in existing libraries**
  Implementing Gaussian Belief Propagation in a way that is compatible with factor graph libraries such as GTSAM [65], would greatly facilitate comparing to state-of-the-art research. Furthermore, it would be a key enabler to test Gaussian Belief Propagation with more advanced navigation representations (quaternions, lie groups or general curved manifolds).
- **Estimate biases in GBP**
  Explicitly estimating biases as part of the state in GBP, as is common in kalman filtering, would enable GBP to better localise the landed drone. The measurements of the IMU when the drone is stationary, in a real-life scenario, could be used to re-calibrate the biases, and backtracking this new knowledge over the whole trajectory in the factor graph could lead to a more pronounced improvement on location estimation.
- **Use different message passing schemes in GBP**
  In this thesis, the parallel message passing scheme was used. However, in the scenario this thesis concerns, with each drone's graph being quite sparse, different message passing schemes could provide a much higher efficiency.

References

[1] J. Li, G. Yang, and Q. Cai, "Autonomous Aerial–Ground Cooperative Navigation Based on Information-Seeking in GNSS-Denied Environments," *IEEE Internet of Things Journal*, vol. 10, no. 19, pp. 17 058–17 069, Oct. 2023, ISSN: 2327-4662. DOI: 10.1109/JIOT.2023.3271643. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/10112646 (visited on 10/08/2025).

[2] I. Chandran and K. Vipin, "Multi-UAV networks for disaster monitoring: Challenges and opportunities from a network perspective," *Drone Systems and Applications*, vol. 12, pp. 1–28, Jan. 2024. DOI: 10.1139/dsa-2023-0079. [Online]. Available: https://cdnsciencepub.com/doi/full/10.1139/dsa-2023-0079 (visited on 10/21/2025).

[3] S. Boers, "Distributed Multi-Robot Exploration Missions: Unified approach using Gaussian Belief Propagation," en, Ph.D. dissertation, 2024. [Online]. Available: https://resolver.tudelft.nl/uuid:c2afc343-813a-4dbb-908d-8952d2eb5686.

[4] E. T. Alotaibi, S. S. Alqefari, and A. Koubaa, "LSAR: Multi-UAV Collaboration for Search and Rescue Missions," *IEEE Access*, vol. 7, pp. 55 817–55 832, 2019, ISSN: 2169-3536. DOI: 10.1109/ACCESS.2019.2912306. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8695011 (visited on 10/21/2025).

[5] Y. Tian, Y. Chang, F. Herrera Arias, C. Nieto-Granda, J. P. How, and L. Carlone, "Kimera-Multi: Robust, Distributed, Dense Metric-Semantic SLAM for Multi-Robot Systems," *IEEE Transactions on Robotics*, vol. 38, no. 4, pp. 2022–2038, Aug. 2022, ISSN: 1941-0468. DOI: 10.1109/TRO.2021.3137751. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9686955 (visited on 09/17/2025).

[6] L. Ruan, G. Li, W. Dai, S. Tian, G. Fan, J. Wang, and X. Dai, "Cooperative Relative Localization for UAV Swarm in GNSS-Denied Environment: A Coalition Formation Game Approach," *IEEE Internet of Things Journal*, vol. 9, no. 13, pp. 11 560–11 577, Jul. 2022, ISSN: 2327-4662. DOI: 10.1109/JIOT.2021.3130000. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9624939 (visited on 10/08/2025).

[7] S. Vakulina, *Russia's hybrid war on Europe: Jamming and spoofing in the 'grey zone'*, en, Section: news_news. [Online]. Available: https://www.euronews.com/2025/09/05/tackling-russias-hybrid-war-on-europe-jamming-and-spoofing-in-the-grey-zone (visited on 03/26/2026).

[8] H. Yao, X. Liang, R. Chen, X. Wang, H. Qi, L. Chen, and Y. Wang, "A Benchmark of Absolute and Relative Positioning Solutions in GNSS Denied Environments," *IEEE Internet of Things Journal*, vol. 11, no. 3, pp. 4243–4273, Feb. 2024, ISSN: 2327-4662. DOI: 10.1109/JIOT.2023.3300018. [Online]. Available: https://ieeexplore.ieee.org/document/10197620/ (visited on 09/08/2025).

[9] T. Qin, P. Li, and S. Shen, "VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, Aug. 2018, ISSN: 1941-0468. DOI: 10.1109/TRO.2018.2853729. [Online]. Available: https://ieeexplore.ieee.org/document/8421746 (visited on 11/11/2025).

[10] H. Matsuki, R. Murai, P. H. J. Kelly, and A. J. Davison, "Gaussian Splatting SLAM," en, in *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Seattle, WA, USA: IEEE, Jun. 2024, pp. 18 039–18 048, ISBN: 979-8-3503-5300-6. DOI: 10.1109/CVPR52733.2024.01708. [Online]. Available: https://ieeexplore.ieee.org/document/10657715/ (visited on 09/24/2025).

[11] J. Li, G. Yang, Q. Cai, H. Niu, and J. Li, "Cooperative navigation for UAVs in GNSS-denied area based on optimized belief propagation," en, *Measurement*, vol. 192, p. 110 797, Mar. 2022, ISSN: 02632241. DOI: 10.1016/j.measurement.2022.110797. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0263224122000975 (visited on 10/14/2025).

[12] S. Wang, Y. Wang, D. Li, and Q. Zhao, “Distributed Relative Localization Algorithms for Multi-Robot Networks: A Survey,” en, *Sensors*, vol. 23, no. 5, p. 2399, Jan. 2023, ISSN: 1424-8220. DOI: 10.3390/s23052399. [Online]. Available: https://www.mdpi.com/1424-8220/23/5/2399 (visited on 09/12/2025).

[13] C. Tu, X. Cui, G. Liu, and M. Lu, *Weighted Covariance Intersection for Range-based Distributed Cooperative Localization of Multi-Agent Systems*, arXiv:2508.12207 [eess], Aug. 2025. DOI: 10.48550/arXiv.2508.12207. [Online]. Available: http://arxiv.org/abs/2508.12207 (visited on 10/13/2025).

[14] R. Kurazume, S. Nagata, and S. Hirose, “Cooperative positioning with multiple robots,” in *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, May 1994, 1250–1257 vol.2. DOI: 10.1109/ROBOT.1994.351315. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/351315 (visited on 10/20/2025).

[15] L. Chen, C. Liang, S. Yuan, M. Cao, and L. Xie, “Relative Localizability and Localization for Multirobot Systems,” *IEEE Transactions on Robotics*, vol. 41, pp. 2931–2949, 2025, ISSN: 1941-0468. DOI: 10.1109/TRO.2025.3544103. [Online]. Available: https://ieeexplore.ieee.org/document/10896856/ (visited on 09/30/2025).

[16] T. Li, X. Yu, Q. Lin, Y. Lv, G. Wen, and C. Shi, “Distributed Cooperative Localization for Unmanned Systems Using UWB/INS Integration in GNSS-Denied Environments,” *IEEE Transactions on Instrumentation and Measurement*, vol. 74, pp. 1–13, 2025, ISSN: 1557-9662. DOI: 10.1109/TIM.2025.3559161. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/10960434 (visited on 10/08/2025).

[17] B. Zhou, M. Tang, C. Liu, X. Zhong, H. He, X. Chen, J. Song, Y. Wang, X. Zhang, and Q. Li, “Cooperative Indoor Localization Using Mobile Robot Anchors via Factor Graph Optimization,” *IEEE Internet of Things Journal*, vol. 12, no. 15, pp. 31 420–31 431, Aug. 2025, ISSN: 2327-4662. DOI: 10.1109/JIOT.2025.3574883. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/11017671 (visited on 10/06/2025).

[18] Y. Qu and Y. Zhang, “Cooperative Localization of Low-cost UAV Using Relative Range Measurements in Multi-UAV Flight,” en, in *AIAA Guidance, Navigation, and Control Conference*, Toronto, Ontario, Canada: American Institute of Aeronautics and Astronautics, Aug. 2010, ISBN: 978-1-60086-962-4. DOI: 10.2514/6.2010-8187. [Online]. Available: https://arc.aiaa.org/doi/10.2514/6.2010-8187 (visited on 11/17/2025).

[19] D. Akhihiero, U. Olawoye, S. Das, and J. Gross, “Cooperative Localization for GNSS-Denied Subterranean Navigation: A UAV–UGV Team Approach,” en, *NAVIGATION: Journal of the Institute of Navigation*, vol. 71, no. 4, navi.677, 2024, ISSN: 0028-1522, 2161-4296. DOI: 10.33012/navi.677. [Online]. Available: http://navi.ion.org/lookup/doi/10.33012/navi.677 (visited on 10/25/2025).

[20] X. Cheng, W. Shi, W. Cai, W. Zhu, T. Shen, F. Shu, and J. Wang, “Communication-Efficient Coordinated RSS-Based Distributed Passive Localization via Drone Cluster,” *IEEE Transactions on Vehicular Technology*, vol. 71, no. 1, pp. 1072–1076, Jan. 2022, ISSN: 1939-9359. DOI: 10.1109/TVT.2021.3125361. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9606574 (visited on 09/15/2025).

[21] S. Wang, Y. Wang, X. Bai, and D. Li, “Communication Efficient, Distributed Relative State Estimation in UAV Networks,” *IEEE Journal on Selected Areas in Communications*, vol. 41, no. 4, pp. 1151–1166, Apr. 2023, ISSN: 1558-0008. DOI: 10.1109/JSAC.2023.3242708. [Online]. Available: https://ieeexplore.ieee.org/document/10038530/ (visited on 09/17/2025).

[22] I. Rekleitis, G. Dudek, and E. Milios, “Multi-robot collaboration for robust exploration,” en, in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, vol. 4, San Francisco, CA, USA: IEEE, 2000, pp. 3164–3169, ISBN: 978-0-7803-5886-7. DOI: 10.1109/ROBOT.2000.845150. [Online]. Available: http://ieeexplore.ieee.org/document/845150/ (visited on 10/22/2025).

[23] A. J. Davison and J. Ortiz, *FutureMapping 2: Gaussian Belief Propagation for Spatial AI*, arXiv:1910.14139 [cs], Nov. 2022. DOI: 10.48550/arXiv.1910.14139. [Online]. Available: http://arxiv.org/abs/1910.14139 (visited on 09/11/2025).

[24] S. Roumeliotis and G. Bekey, “Distributed multirobot localization,” *IEEE Transactions on Robotics and Automation*, vol. 18, no. 5, pp. 781–795, Oct. 2002, ISSN: 2374-958X. DOI: 10.1109/TRA.2002.803461. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/1067998 (visited on 10/22/2025).

[25] 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, ISSN: 0921-8890. DOI: 10.1016/j.robot.2018.09.010. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0921889018301969 (visited on 10/09/2025).

[26] B. Chenchana, O. Labbani-Igbida, S. Renault, and S. Boria, “Range-Based Collaborative MSCKF Localization,” in *2018 25th International Conference on Mechatronics and Machine Vision in Practice (M2VIP)*, Nov. 2018, pp. 1–6. DOI: 10.1109/M2VIP.2018.8600900. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8600900 (visited on 09/15/2025).

[27] Y. Li, Y. Wang, W. Yu, and X. Guan, “Multiple Autonomous Underwater Vehicle Cooperative Localization in Anchor-Free Environments,” *IEEE Journal of Oceanic Engineering*, vol. 44, no. 4, pp. 895–911, Oct. 2019, ISSN: 1558-1691. DOI: 10.1109/JOE.2019.2935516. [Online]. Available: https://ieeexplore.ieee.org/document/8826251/ (visited on 10/14/2025).

[28] C. Liang, L. Chen, B. Cui, and J. Mei, “3-D relative localization for multi-robot systems with angle and self-displacement measurements,” EN, *The International Journal of Robotics Research*, p. 02 783 649 251 363 276, Sep. 2025, ISSN: 0278-3649. DOI: 10.1177/02783649251363276. [Online]. Available: https://doi.org/10.1177/02783649251363276 (visited on 09/30/2025).

[29] M. Todescato, A. Carron, R. Carli, and L. Schenato, “Distributed localization from relative noisy measurements: A robust gradient based approach,” en, in *2015 European Control Conference (ECC)*, Linz, Austria: IEEE, Jul. 2015, pp. 1914–1919, ISBN: 978-3-9524269-3-7. DOI: 10.1109/ECC.2015.7330818. [Online]. Available: https://ieeexplore.ieee.org/document/7330818/ (visited on 09/11/2025).

[30] S. Choudhary, L. Carlone, C. Nieto, J. Rogers, H. I. Christensen, and F. Dellaert, “Distributed trajectory estimation with privacy and communication constraints: A two-stage distributed Gauss-Seidel approach,” in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, May 2016, pp. 5261–5268. DOI: 10.1109/ICRA.2016.7487736. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7487736 (visited on 09/22/2025).

[31] S. Choudhary, L. Carlone, C. Nieto, J. Rogers, H. I. Christensen, and F. Dellaert, “Distributed mapping with privacy and communication constraints: Lightweight algorithms and object-based models,” EN, *The International Journal of Robotics Research*, vol. 36, no. 12, pp. 1286–1311, Oct. 2017, ISSN: 0278-3649. DOI: 10.1177/0278364917732640. [Online]. Available: https://doi.org/10.1177/0278364917732640 (visited on 09/12/2025).

[32] Y. Tian, A. Koppel, A. S. Bedi, and J. P. How, “Asynchronous and Parallel Distributed Pose Graph Optimization,” *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5819–5826, Oct. 2020, ISSN: 2377-3766. DOI: 10.1109/LRA.2020.3010216. [Online]. Available: https://ieeexplore.ieee.org/document/9143442 (visited on 09/08/2025).

[33] Y. Tian, K. Khosoussi, D. M. Rosen, and J. P. How, “Distributed Certifiably Correct Pose-Graph Optimization,” *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 2137–2156, Dec. 2021, ISSN: 1941-0468. DOI: 10.1109/TRO.2021.3072346. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9425438 (visited on 09/12/2025).

[34] T. Halsted, O. Shorinwa, J. Yu, and M. Schwager, *A Survey of Distributed Optimization Methods for Multi-Robot Systems*, arXiv:2103.12840 [cs], Mar. 2021. DOI: 10.48550/arXiv.2103.12840. [Online]. Available: http://arxiv.org/abs/2103.12840 (visited on 09/10/2025).

[35] A. Cunningham, M. Paluri, and F. Dellaert, “DDF-SAM: Fully distributed SLAM using Constrained Factor Graphs,” en, in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Taipei: IEEE, Oct. 2010, pp. 3025–3030, ISBN: 978-1-4244-6674-0. DOI: 10.1109/IROS.2010.5652875. [Online]. Available: http://ieeexplore.ieee.org/document/5652875/ (visited on 09/15/2025).

[36] A. Cunningham, V. Indelman, and F. Dellaert, "DDF-SAM 2.0: Consistent distributed smoothing and mapping," in *2013 IEEE International Conference on Robotics and Automation*, May 2013, pp. 5220–5227. DOI: 10.1109/ICRA.2013.6631323. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/6631323 (visited on 09/12/2025).

[37] R. Murai, I. Alzugaray, P. H. Kelly, and A. J. Davison, "Distributed Simultaneous Localisation and Auto-Calibration Using Gaussian Belief Propagation," *IEEE Robotics and Automation Letters*, vol. 9, no. 3, pp. 2136–2143, Mar. 2024, ISSN: 2377-3766. DOI: 10.1109/LRA.2024.3352361. [Online]. Available: https://ieeexplore.ieee.org/document/10387679 (visited on 09/10/2025).

[38] X. Dong, G. Hu, B. Gao, Y. Zhong, and W. Ruan, "Windowing-Based Factor Graph Optimization With Anomaly Detection Using Mahalanobis Distance for Underwater INS/DVL/USBL Integration," *IEEE Transactions on Instrumentation and Measurement*, vol. 73, pp. 1–13, 2024, ISSN: 1557-9662. DOI: 10.1109/TIM.2024.3353286. [Online]. Available: https://ieeexplore.ieee.org/document/10398248/ (visited on 02/09/2026).

[39] H. Tang, T. Zhang, X. Niu, J. Fan, and J. Liu, "Impact of the Earth Rotation Compensation on MEMS-IMU Preintegration of Factor Graph Optimization," *IEEE Sensors Journal*, vol. 22, no. 17, pp. 17 194–17 204, Sep. 2022, ISSN: 1558-1748. DOI: 10.1109/JSEN.2022.3192552. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9841461 (visited on 11/12/2025).

[40] F. Dellaert, "Factor Graphs: Exploiting Structure in Robotics," en, *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 4, no. Volume 4, 2021, pp. 141–166, May 2021, ISSN: 2573-5144. DOI: 10.1146/annurev-control-061520-010504. [Online]. Available: https://www.annualreviews.org/content/journals/10.1146/annurev-control-061520-010504 (visited on 10/07/2025).

[41] A. T. Ihler, J. W. Fisher, R. L. Moses, and A. S. Willsky, "Nonparametric belief propagation for self-calibration in sensor networks," in *Proceedings of the 3rd international symposium on Information processing in sensor networks*, ser. IPSN '04, New York, NY, USA: Association for Computing Machinery, Apr. 2004, pp. 225–233, ISBN: 978-1-58113-846-7. DOI: 10.1145/984622.984656. [Online]. Available: https://dl.acm.org/doi/10.1145/984622.984656 (visited on 10/23/2025).

[42] F. Meyer, O. Hlinka, and F. Hlawatsch, "Sigma Point Belief Propagation," *IEEE Signal Processing Letters*, vol. 21, no. 2, pp. 145–149, Feb. 2014, ISSN: 1558-2361. DOI: 10.1109/LSP.2013.2290192. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/6661389 (visited on 10/23/2025).

[43] Á. F. García-Fernández, L. Svensson, and S. Särkkä, "Cooperative Localization Using Posterior Linearization Belief Propagation," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 1, pp. 832–836, Jan. 2018, ISSN: 1939-9359. DOI: 10.1109/TVT.2017.2734683. [Online]. Available: https://ieeexplore.ieee.org/document/7999230/ (visited on 10/14/2025).

[44] R. Murai, J. Ortiz, S. Saeedi, P. H. J. Kelly, and A. J. Davison, "A Robot Web for Distributed Many-Device Localization," en, *IEEE Transactions on Robotics*, vol. 40, pp. 121–138, 2024, ISSN: 1552-3098, 1941-0468. DOI: 10.1109/TRO.2023.3324127. [Online]. Available: https://ieeexplore.ieee.org/document/10286058/ (visited on 09/02/2025).

[45] A. Patwardhan, R. Murai, and A. J. Davison, "Distributing Collaborative Multi-Robot Planning With Gaussian Belief Propagation," en, *IEEE Robotics and Automation Letters*, vol. 8, no. 2, pp. 552–559, Feb. 2023, ISSN: 2377-3766, 2377-3774. DOI: 10.1109/LRA.2022.3227858. [Online]. Available: https://ieeexplore.ieee.org/document/9976221/ (visited on 09/02/2025).

[46] A. Patwardhan and A. J. Davison, "A Distributed Multi-Robot Framework for Exploration, Information Acquisition and Consensus," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, May 2024, pp. 12 062–12 068. DOI: 10.1109/ICRA57147.2024.10610185. [Online]. Available: https://ieeexplore-ieee-org.tudelft.idm.oclc.org/document/10610185 (visited on 09/10/2025).

[47] Y. Fan, K. Ding, X. Qi, and L. Liu, “Cooperative Localization of 3D Mobile Networks via Relative Distance and Velocity Measurement,” *IEEE Communications Letters*, vol. 25, no. 9, pp. 2899–2903, Sep. 2021, ISSN: 1558-2558. DOI: 10.1109/LCOMM.2021.3087498. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9448229 (visited on 11/11/2025).

[48] O. Pohjavirta, X. Liang, Y. Wang, A. Kukko, J. Pyörälä, E. Hyyppä, X. Yu, H. Kaartinen, and J. Hyyppä, “Automated registration of wide-baseline point clouds in forests using discrete overlap search,” *Forest Ecosystems*, vol. 9, p. 100 080, Jan. 2022, ISSN: 2197-5620. DOI: 10.1016/j.fecs.2022.100080. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S219756202200080X (visited on 03/17/2026).

[49] K. Guo, X. Li, and L. Xie, “Ultra-Wideband and Odometry-Based Cooperative Relative Localization With Application to Multi-UAV Formation Control,” *IEEE Transactions on Cybernetics*, vol. 50, no. 6, pp. 2590–2603, Jun. 2020, ISSN: 2168-2275. DOI: 10.1109/TCYB.2019.2905570. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8680745 (visited on 10/22/2025).

[50] S. O. Dulman, A. Baggio, P. J. Havinga, and K. G. Langendoen, “A geometrical perspective on localization,” en, in *Proceedings of the first ACM international workshop on Mobile entity localization and tracking in GPS-less environments*, San Francisco California USA: ACM, Sep. 2008, pp. 85–90, ISBN: 978-1-60558-189-7. DOI: 10.1145/1410012.1410032. [Online]. Available: https://dl.acm.org/doi/10.1145/1410012.1410032 (visited on 10/06/2025).

[51] T. Liang, T. Zhang, J. Yang, D. Feng, and Q. Zhang, “UAV-Aided Positioning Systems for Ground Devices: Fundamental Limits and Algorithms,” *IEEE Internet of Things Journal*, vol. 9, no. 15, pp. 13 470–13 485, Aug. 2022, ISSN: 2327-4662. DOI: 10.1109/JIOT.2022.3144234. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9684492 (visited on 10/08/2025).

[52] T. R. Wanasinghe, G. K. I. Mann, and R. G. Gosine, “Distributed Leader-Assistive Localization Method for a Heterogeneous Multirobotic System,” *IEEE Transactions on Automation Science and Engineering*, vol. 12, no. 3, pp. 795–809, Jul. 2015, ISSN: 1558-3783. DOI: 10.1109/TASE.2015.2433014. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7114338 (visited on 10/13/2025).

[53] S.-I. Sou, F.-J. Wu, and W.-C. Wu, “JoLo: Multi-Device Joint Localization Based on Wireless Data Fusion,” *IEEE Transactions on Mobile Computing*, vol. 22, no. 7, pp. 4016–4031, Jul. 2023, ISSN: 1558-0660. DOI: 10.1109/TMC.2022.3150758. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9712352 (visited on 10/07/2025).

[54] J. Ortiz, T. Evans, and A. J. Davison, *A visual introduction to Gaussian Belief Propagation*, en, arXiv:2107.02308 [cs], Jul. 2021. DOI: 10.48550/arXiv.2107.02308. [Online]. Available: http://arxiv.org/abs/2107.02308 (visited on 09/02/2025).

[55] J. Bongard, “Probabilistic Robotics. Sebastian Thrun, Wolfram Burgard, and Dieter Fox. (2005, MIT Press.) 647 pages,” *Artificial Life*, vol. 14, no. 2, pp. 227–229, Apr. 2008, ISSN: 1064-5462. DOI: 10.1162/artl.2008.14.2.227. [Online]. Available: https://ieeexplore.ieee.org/document/6792214 (visited on 05/24/2026).

[56] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, en. Elsevier, Jun. 2014, Google-Books-ID: mn2jBQAAQBAJ, ISBN: 978-0-08-051489-5.

[57] K. Murphy, Y. Weiss, and M. I. Jordan, *Loopy Belief Propagation for Approximate Inference: An Empirical Study*, arXiv:1301.6725 [cs], Jan. 2013. DOI: 10.48550/arXiv.1301.6725. [Online]. Available: http://arxiv.org/abs/1301.6725 (visited on 10/22/2025).

[58] M. J. Wainwright and M. I. Jordan, “Graphical Models, Exponential Families, and Variational Inference,” English, *Foundations and Trends® in Machine Learning*, vol. 1, no. 1–2, pp. 1–305, Nov. 2008, ISSN: 1935-8237, 1935-8245. DOI: 10.1561/2200000001. [Online]. Available: https://www.nowpublishers.com/article/Details/MAL-001 (visited on 10/22/2025).

[59] Y. Weiss and W. T. Freeman, “Correctness of Belief Propagation in Gaussian Graphical Models of Arbitrary Topology,” *Neural Computation*, vol. 13, no. 10, pp. 2173–2200, Oct. 2001, ISSN: 0899-7667. DOI: 10.1162/089976601750541769. [Online]. Available: https://doi.org/10.1162/089976601750541769 (visited on 10/31/2025).

[60] M. Alberi, M. Baldoncini, C. Bottardi, E. Chiarelli, G. Fiorentini, K. G. C. Raptis, E. Realini, M. Reguzzoni, L. Rossi, D. Sampietro, V. Strati, and F. Mantovani, "Accuracy of Flight Altitude Measured with Low-Cost GNSS, Radar and Barometer Sensors: Implications for Airborne Radiometric Surveys," *Sensors*, vol. 17, no. 8, p. 1889, Aug. 2017, arXiv:1802.00327 [physics], ISSN: 1424-8220. DOI: 10.3390/s17081889. [Online]. Available: http://arxiv.org/abs/1802.00327 (visited on 04/23/2026).

[61] B. Etzlinger, F. Meyer, F. Hlawatsch, A. Springer, and H. Wymeersch, "Cooperative Simultaneous Localization and Synchronization in Mobile Agent Networks," *IEEE Transactions on Signal Processing*, vol. 65, no. 14, pp. 3587–3602, Jul. 2017, ISSN: 1941-0476. DOI: 10.1109/TSP.2017.2691665. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7893772 (visited on 10/21/2025).

[62] J. Wahlström and I. Skog, "Fifteen Years of Progress at Zero Velocity: A Review," *IEEE Sensors Journal*, vol. 21, no. 2, pp. 1139–1151, Jan. 2021, ISSN: 1558-1748. DOI: 10.1109/JSEN.2020.3018880. [Online]. Available: https://ieeexplore.ieee.org/document/9174869 (visited on 04/23/2026).

[63] C. Kilic, S. Das, E. Gutierrez, R. Watson, and J. Gross, "ZUPT Aided GNSS Factor Graph with Inertial Navigation Integration for Wheeled Robots," arXiv:2112.07176 [cs], Oct. 2021, pp. 3285–3293. DOI: 10.33012/2021.18064. [Online]. Available: http://arxiv.org/abs/2112.07176 (visited on 04/23/2026).

[64] *The UTIAS multi-robot cooperative localization and mapping dataset*, en. DOI: 10.1177/0278364911398404. [Online]. Available: https://journals.sagepub.com/doi/epdf/10.1177/0278364911398404 (visited on 09/24/2025).

[65] F. Dellaert and M. Kaess, "Factor Graphs for Robot Perception," en, *Foundations and Trends in Robotics*, vol. 6, no. 1-2, pp. 1–139, 2017, ISSN: 1935-8253, 1935-8261. DOI: 10.1561/2300000043. [Online]. Available: http://www.nowpublishers.com/article/Details/ROB-043 (visited on 05/20/2026).

A

Mathematics

A.1. Analytical Jacobian for IMU factor

$$\mathbf{z} = \begin{bmatrix} \Delta\theta \\ \Delta x \\ \Delta y \end{bmatrix} \quad , \quad \mathbf{x} = (\theta_a, x_a, y_a, \theta_b, x_b, y_b) \tag{A.1}$$

$$\Delta x_w = x_b - x_a, \quad \Delta y_w = y_b - y_a \tag{A.2}$$

$$h(\mathbf{x}) = \begin{bmatrix} \theta_b - \theta_a \\ cos\,\theta_a \Delta x_w + sin\,\theta_a \Delta y_w \\ -sin\,\theta_a \Delta x_w + cos\,\theta_a \Delta y_w \end{bmatrix} \tag{A.3}$$

$$\frac{\delta\Delta\theta}{\delta\theta_a} = -1, \quad \frac{\delta\Delta\theta}{\delta x_a} = 0, \quad \frac{\delta\Delta\theta}{\delta y_a} = 0, \quad \frac{\delta\Delta\theta}{\delta\theta_b} = 1, \quad \frac{\delta\Delta\theta}{\delta x_b} = 0, \quad \frac{\delta\Delta\theta}{\delta y_b} = 0, \tag{A.4}$$

$$\frac{\delta\Delta x}{\delta\theta_a} = -sin\,\theta_a \delta\Delta x_w + cos\,\theta_a \Delta y_w, \quad \frac{\delta\Delta x}{\delta x_a} = -cos\,\theta_a, \quad \frac{\delta\Delta x}{\delta y_a} = -sin\,\theta_a \tag{A.5}$$

$$\frac{\delta\Delta x}{\delta\theta_b} = 0, \quad \frac{\delta\Delta x}{\delta x_b} = cos\,\theta_a, \quad \frac{\delta\Delta x}{\delta y_b} = sin\,\theta_b \tag{A.6}$$

$$\frac{\delta\Delta y}{\delta\theta_a} = -cos\,\theta_a \Delta x_w - sin\,\theta_a \Delta y_w, \quad \frac{\delta\Delta y}{\delta x_a} = sin\,\theta_a, \quad \frac{\delta\Delta y}{\delta y_a} = -cos\,\theta_a \tag{A.7}$$

$$\frac{\delta\Delta y}{\delta\theta_b} = 0, \quad \frac{\delta\Delta y}{\delta x_b} = -sin\,\theta_a, \quad \frac{\delta\Delta y}{\delta y_b} = cos\,\theta_a \tag{A.8}$$

A.2. Analytical Jacobian for range-bearing factor

$$\mathbf{z} = \begin{bmatrix} \phi \\ r \end{bmatrix} \quad , \quad \mathbf{x} = (\theta_a, x_a, y_a, \theta_b, x_b, y_b) \tag{A.9}$$

$$h(\mathbf{x}) = \begin{bmatrix} \phi \\ r \end{bmatrix} = \begin{bmatrix} atan2(y_b - y_a, x_b - x_a) - \theta_a \\ \sqrt{(x_b - x_a)^2 + (y_b - y_a)^2} \end{bmatrix} \tag{A.10}$$

$$\Delta x = x_b - x_a, \quad \Delta y = y_b - y_a, \quad q = \Delta x^2 + \Delta y^2, \quad r = \sqrt{q} \tag{A.11}$$

$$\frac{\delta r}{\delta x_a} = \frac{\delta r}{\delta q} \cdot \frac{\delta q}{\delta x_a} = \frac{1}{2\sqrt{q}} \cdot 2\Delta x \cdot -1 = \frac{-\Delta x}{\sqrt{q}} (= \frac{x_a - x_b}{\sqrt{(x_b - x_a)^2 + (y_b - y_a)^2}}) \tag{A.12}$$

$$\frac{\delta r}{\delta x_b} = \frac{\delta r}{\delta q} \cdot \frac{\delta q}{\delta x_b} = \frac{1}{2\sqrt{q}} \cdot 2\Delta x \cdot\ 1 = \frac{\Delta x}{\sqrt{q}} (= \frac{x_b - x_a}{\sqrt{(x_b - x_a)^2 + (y_b - y_a)^2}}) \tag{A.13}$$

$$\frac{\delta r}{y_a} = \frac{\delta r}{\delta q} \cdot \frac{\delta q}{\delta y_a} = \frac{1}{2\sqrt{q}} \cdot 2\delta y \cdot -1 = \frac{-\Delta y}{\sqrt{q}} (= \frac{y_a - y_b}{\sqrt{(x_b - x_a)^2 + (y_b - y_a)^2}}) \tag{A.14}$$

$$\frac{\delta r}{\delta y_b} = \frac{\delta r}{\delta q} \cdot \frac{\delta q}{\delta y_b} = \frac{1}{2\sqrt{q}} \cdot 2\Delta y \cdot 1 = \frac{\Delta y}{\sqrt{q}} (= \frac{y_b - y_a}{\sqrt{(x_b - x_a)^2 + (y_b - y_a)^2}}) \tag{A.15}$$

$$\frac{\delta r}{\delta \theta a} = 0, \quad \frac{\delta r}{\delta \theta b} = 0 \tag{A.16}$$

$$\frac{\delta \phi}{\delta x_a} = \frac{\delta \phi}{\delta \Delta x} \cdot \frac{\delta \Delta x}{\delta x_a} = \frac{-\Delta y}{q} \cdot -1 = \frac{\Delta y}{q} \tag{A.17}$$

$$\frac{\delta \phi}{\delta x_b} = \frac{\delta \phi}{\delta \Delta y} \cdot \frac{\delta \Delta y}{\delta x_a} = \frac{\Delta x}{q} \cdot -1 = \frac{-\Delta y}{q} \tag{A.18}$$

$$\frac{\delta \phi}{\delta y_a} = \frac{\delta \phi}{\delta \Delta y} \cdot \frac{\delta \Delta y}{\delta y_a} = \frac{\Delta x}{q} \cdot -1 = \frac{-\Delta x}{q} \tag{A.19}$$

$$\frac{\delta \phi}{\delta y_b} = \frac{\delta \phi}{\delta \Delta y} \cdot \frac{\delta \Delta y}{\delta y_a} = \frac{\Delta x}{q} \cdot 1 = \frac{\Delta x}{q} \tag{A.20}$$

$$\frac{\delta \phi}{\delta \theta a} = -1, \quad \frac{\delta \phi}{\delta \theta b} = 0 \tag{A.21}$$

$$J_{ij} = \frac{\partial h_i}{\partial x_j} \quad , \quad \mathbf{J} = \begin{bmatrix} -1 & \frac{\Delta y}{q} & \frac{-\Delta x}{q} & 0 & \frac{-\Delta y}{q} & \frac{\Delta x}{q} \\ 0 & \frac{-\Delta x}{q} & \frac{\Delta x}{q} & 0 & \frac{\Delta x}{q} & \frac{\Delta y}{q} \end{bmatrix} \tag{A.22}$$

A.3. Uncertainty propagation

non-linear kinematics:

$$\begin{aligned} x_k &= x_{k-1} + v_k \Delta t \cos(\theta_{k-1}) \\ y_k &= y_{k-1} + v_k \Delta t \sin(\theta_{k-1}) \\ \theta_k &= \theta_{k-1} + \omega_k \Delta t \end{aligned} \tag{A.23}$$

Jacobian:

$$F_k = \begin{bmatrix} \frac{\partial x_k}{\partial x_{k-1}} & \frac{\partial x_k}{\partial y_{k-1}} & \frac{\partial x_k}{\partial \theta_{k-1}} \\ \frac{\partial y_k}{\partial x_{k-1}} & \frac{\partial y_k}{\partial y_{k-1}} & \frac{\partial y_k}{\partial \theta_{k-1}} \\ \frac{\partial \theta_k}{\partial x_{k-1}} & \frac{\partial \theta_k}{\partial y_{k-1}} & \frac{\partial \theta_k}{\partial \theta_{k-1}} \end{bmatrix} \tag{A.24}$$

jacobian for the non-linear kinematics:

$$F_k = \begin{bmatrix} 1 & 0 & -v_k \Delta t \sin(\theta_{k-1}) \\ 0 & 1 & v_k \Delta t \cos(\theta_{k-1}) \\ 0 & 0 & 1 \end{bmatrix} \tag{A.25}$$

substitute:

$$J_x = -v_k \Delta t \sin(\theta_{k-1}), \qquad J_y = v_k \Delta t \cos(\theta_{k-1}) \tag{A.26}$$

$$F_k = \begin{bmatrix} 1 & 0 & J_x \\ 0 & 1 & J_y \\ 0 & 0 & 1 \end{bmatrix} \tag{A.27}$$

Covariance:

$$P_{k-1} = \begin{bmatrix} P_{xx} & P_{xy} & P_{x\theta} \\ P_{yx} & P_{yy} & P_{y\theta} \\ P_{\theta x} & P_{\theta y} & P_{\theta\theta} \end{bmatrix} \tag{A.28}$$

law of propagation:

$$P_k = F_k \; P_{k-1} \; F_k^t + Q \tag{A.29}$$

$$F_k P_{k-1} = \begin{bmatrix} 1 & 0 & J_x \\ 0 & 1 & J_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} P_{xx} & P_{xy} & P_{x\theta} \\ P_{yx} & P_{yy} & P_{y\theta} \\ P_{\theta x} & P_{\theta y} & P_{\theta\theta} \end{bmatrix} \tag{A.30}$$

$$F_k P_{k-1} = \begin{bmatrix} P_{xx} + J_x P_{\theta x} & P_{xy} + J_x P_{\theta y} & P_{x\theta} + J_x P_{\theta\theta} \\ P_{yx} + J_y P_{\theta x} & P_{yy} + J_y P_{\theta y} & P_{y\theta} + J_y P_{\theta\theta} \\ P_{\theta x} & P_{\theta y} & P_{\theta\theta} \end{bmatrix} \tag{A.31}$$

Right-multiplying by F_k^T:

$$F_k^T = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ J_x & J_y & 1 \end{bmatrix} \tag{A.32}$$

$$\begin{aligned} P_{\theta\theta,k} &= P_{\theta\theta,k-1} + Q_{\theta\theta} \\ P_{xx,k} &= P_{xx,k-1} + 2J_x P_{\theta x} + J_x^2 P_{\theta\theta} + Q_{xx} \\ P_{yy,k} &= P_{yy,k-1} + 2J_y P_{\theta y} + J_y^2 P_{\theta\theta} + Q_{yy} \end{aligned} \tag{A.33}$$

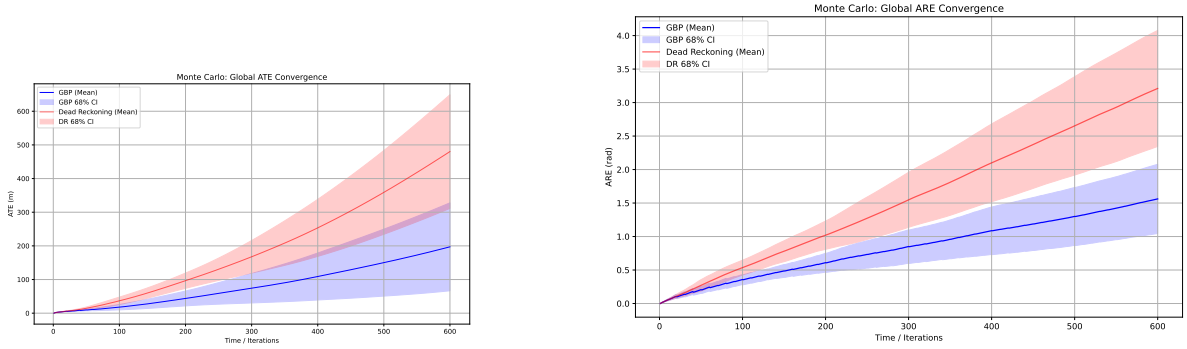
$$\begin{aligned} P_{\theta\theta,k} &= P_{\theta\theta,k-1} + Q_{\theta\theta} \\ P_{xx,k} &= P_{xx,k-1} - 2(v_k \Delta t \sin(\theta_{k-1}))P_{\theta x} + (-v_k \Delta t \sin(\theta_{k-1}))^2 P_{\theta\theta} + Q_{xx} \\ P_{yy,k} &= P_{yy,k-1} + 2(v_k \Delta t \cos(\theta_{k-1}))P_{\theta y} + (v_k \Delta t \cos(\theta_{k-1}))^2 P_{\theta\theta} + Q_{yy} \end{aligned} \tag{A.34}$$

B

Evaluation & Results

B.1. dead-reckoning versus cooperative localisation baseline

Subsection 6.2.1 elaborated on the performance of the cooperative localisation solution using GBP in the context of the Final Drift (FD). This appendix plots in Figure B.1 the performance in terms of ATE and ARE, that evaluate the translation error and rotation error over the full trajectory respectively, reflecting the same trend as the FD. Figure B.2 shows that the error reduction is not local to one drone, but goes for all drones in the system.



(a) trend of mean ATE

(b) trend of mean ARE

Figure B.1: Mean ATE and ARE for the 4 drone diamond formation scenario flight.

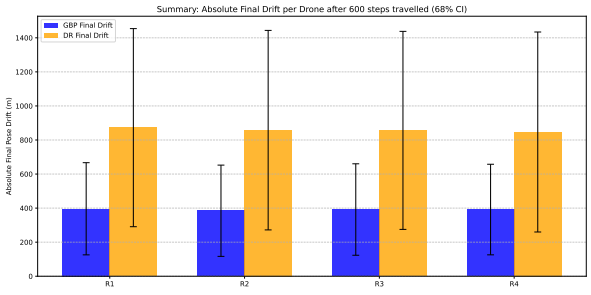
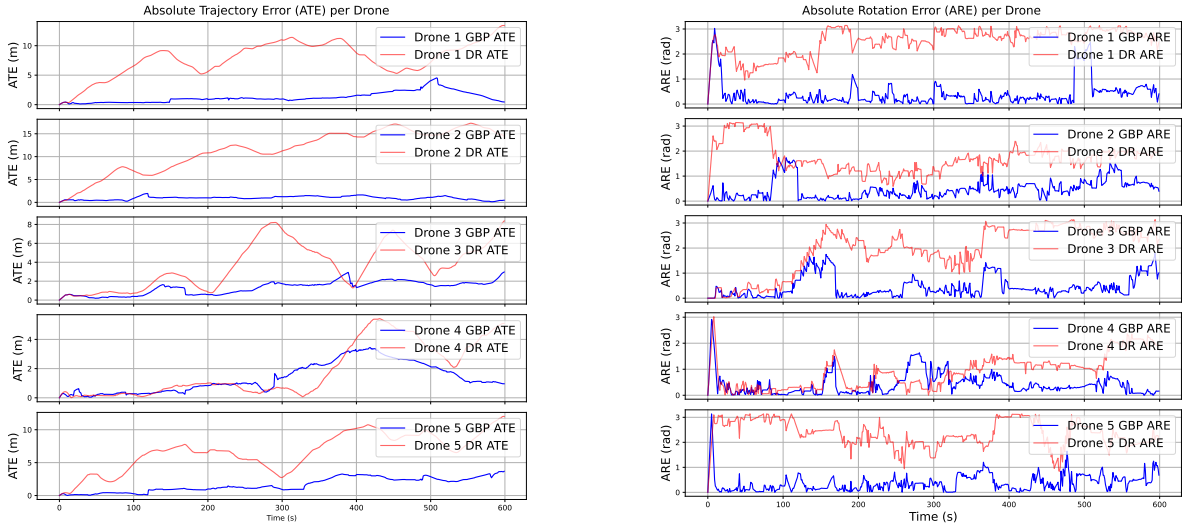


Figure B.2: Barchart of Final Drift for each drone in the 4 drone diamond formation scenario.

B.1.1. UTIAS MR.CLAM datastet

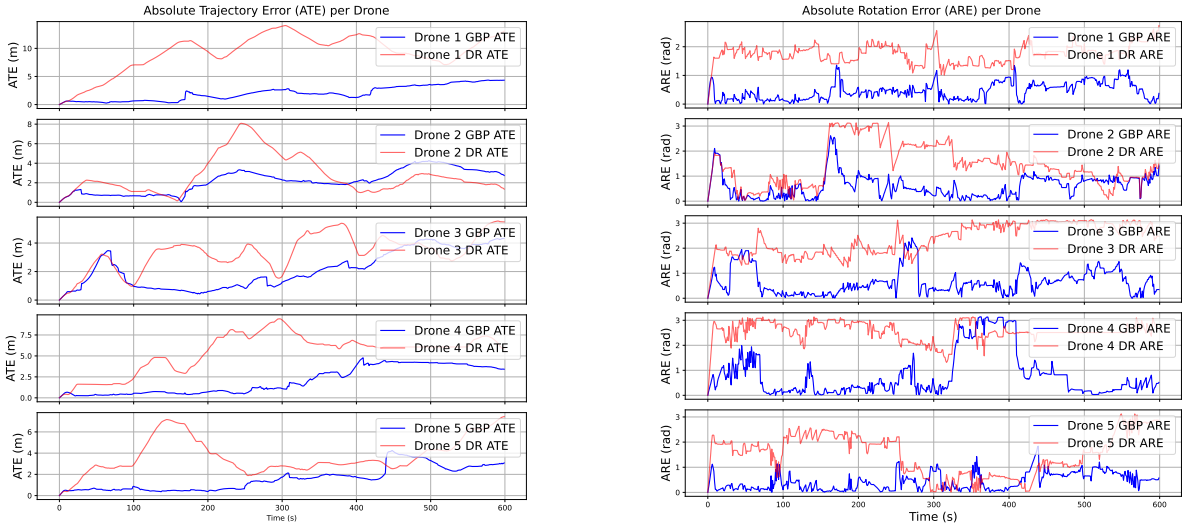
The baseline GBP cooperative localization solution, without any landing manoevres, was evaluated on the UTIAS.MR CLAM [64] datasets 1-4. The performance for dataset 2-4 is visualised in Figure B.3,

Figure B.4 and Figure B.5 respectively.



(a) ATE **(b)** ARE

Figure B.3: A comparison of the ATE and ARE of UTIAS MR.CLAM dataset 2 under DR vs GBP CL without any landmark usage.



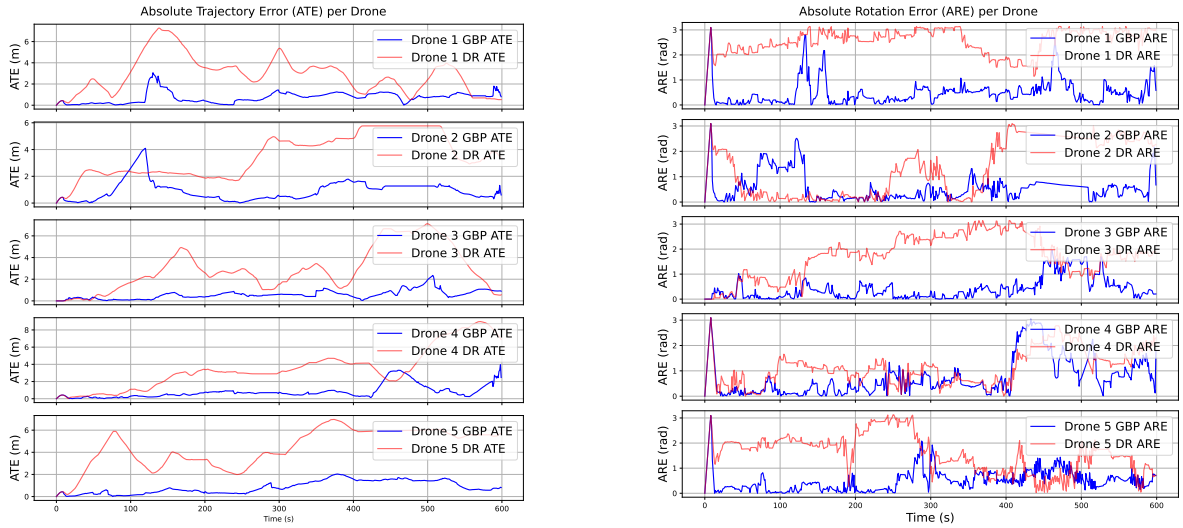
(a) ATE **(b)** ARE

Figure B.4: A comparison of the ATE and ARE of UTIAS MR.CLAM dataset 3 under DR vs GBP CL without any landmark usage.

B.2. Evaluation of landed anchor models

B.2.1. unary anchor factor

The unary anchor factor, in the ideal scenario, receives absolute information. This leads to a large update step in the factor graph, that cannot be accommodated with too few GBP iterations or too small a time-window. Figure B.6 shows that the effect of only increaseing the time-window, without increasing the number of message passing iterations for the GBP algorithm per step, is insufficient. The time-window and number of iterations should therefore always be jointly updated.



(a) ATE **(b)** ARE

Figure B.5: A comparison of the ATE and ARE of UTIAS MR.CLAM dataset 4 under DR vs GBP CL without any landmark usage.

B.3. Multi-group topologies

Figure B.7 shows how the performance of the landed drone with a very certain ZUPT evolves. Although the uncertainty oscillates, increasing for 10 timesteps before a range-bearing measurement appears, the solution is still overconfident in its covariance estimate. This yields worse performance than the ZUPT with similar certainty to the IMU factor.

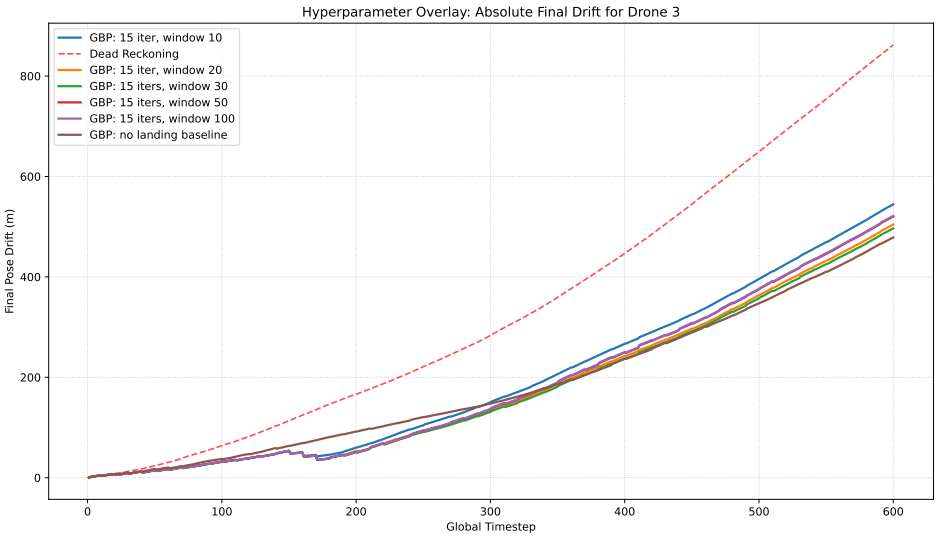
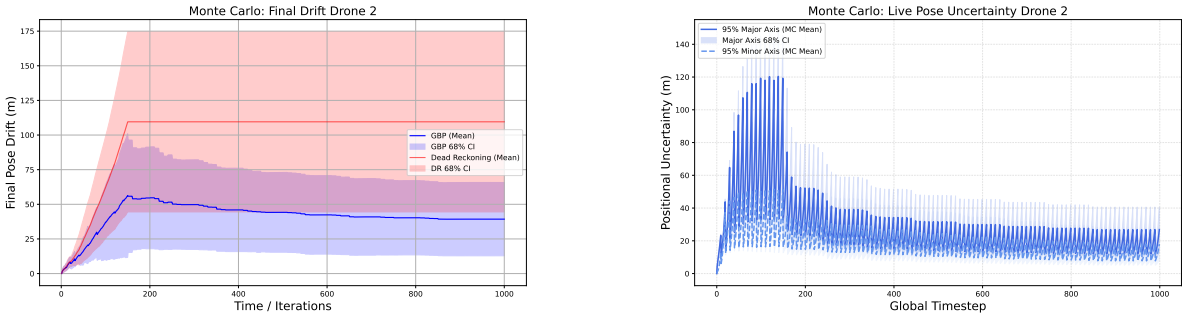


Figure B.6: Overlay of FD characteristics for different time-window sizes under equal amount of message passing iterations.



(a) FD landed drone

(b) Covariance landed drone half major and half minor axis

Figure B.7: Final Drift and Covariance estimates for landed drone using ZUPT model $\sigma = [0.005°, 0.001, 0.0002]$, using daisy chain limited range-bearing measurements while in communication range.