MOBILE ROBOT LOCALIZATION UNDER PROCESSING AND COMMUNICATION CONSTRAINTS

A THESIS
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

ESHA NERURKAR

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

STERGIOS ROUMELIOTIS, ADVISOR

March, 2016

Acknowledgements

This work would not have been possible without the guidance and support of my adviser, Professor Stergios Roumeliotis. He epitomizes the ideal teacher; one who teaches by example and passes along not only his knowledge and experience, but also his passion for his work. I am immensely grateful to him for his constant encouragement and invaluable advice, for pushing me beyond my limits, and for providing outstanding research and career opportunities. I am thankful for the time and valuable inputs from my committee members, Professor Saad, Professor Giannakis, Professor Isler, and Professor Kumar. During my time at the MARS lab, I got the opportunity to work with many amazing people, Tassos, Niko, Sam, Faraz, Ke, Joel, Paul, Gian Luca, Agostino, Kejian, Dimitri, Chao, Igor, Luis, Jack, Ahmed, Elliot, Ruipeng, and Ryan. Thank you all for providing such a wonderful work environment filled with stimulating research discussions and laughter in equal measures.

I owe a lot to the vibrant city of Minneapolis, with its mighty Mississippi and numerous lakes, trails, eating joints, and watering holes, for creating a welcoming home away from home. I made many great friends here, DK, Raghav, Ajay, Vinu, Shantanu, Nihar, Neha, Salil (x2), Pu(oo)rva, Aditya, Vinita, Shraddha, Sajal, Anay, Siddharth, Erika, Shameek, Nikhil (x2) (and others, who I am surely forgetting here), with whom I have shared many happy hours full of laughter, food, and play. I am immensely thankful to Aniket, my wonderfully quirky friend, and Manasi, my partner-in-crime since childhood, for single-handedly upholding our friendships through graduate school; their insanity went a long way in keeping me sane.

This work would be incomplete without acknowledging the unwavering support and unconditional love of my family. Even though they are so far away, my parents have been a constant source of inspiration and guidance. They have never failed to believe in

me and have done everything in their power (and then some more) so that I could have this opportunity today. My husband, Rahul, with his seemingly boundless kindness and patience, has been my companion through the everyday ups and downs of graduate school. I am thankful to him for still loving me at the end of this; without him by my side, I would not be writing these words.

Last but not the least, I gratefully acknowledge financial support from the National Science Foundation, the University of Minnesota Department of Computer Science and Engineering, the Digital Technology Center at the University of Minnesota, and the University of Minnesota Graduate School Doctoral Dissertation Fellowship.

Abstract

Mobile robot localization is one of the most fundamental problems in robotics. For robots assisting humans in tasks such as surveillance, search and rescue, and space exploration, accurate localization, that is, precisely estimating the robot's pose (position and orientation), is a prerequisite for autonomous operation. The system resources (processing and communication) for localization, however, are often limited, and their availability varies widely depending upon the application and the operating environment. Therefore, the objective of this work is to develop resource-aware estimators for robot localization, which optimally utilize all available resources in order to maximize estimation accuracy.

In the first part of this thesis, we address the problem of robot localization under *processing* constraints, focusing on the key applications of single-robot Simultaneous Localization and Mapping (SLAM) and multi-robot Cooperative Localization (CL). For SLAM, we propose two resource-aware approaches, the approximate Minimum Mean Squared Error (MMSE) estimator-based Power-SLAM algorithm and the approximate batch Maximum A Posterior (MAP) estimator-based Constrained Keyframe-based Localization and Mapping (C-KLAM). When approximations are inevitable due to processing constraints, both approaches aim to minimize the information loss while generating consistent estimates. For CL, we exploit the sparse structure of the batch MAP estimator to develop a resource-aware, fully-distributed multi-robot localization algorithm, that harnesses the processing, storage, and communication resources of the entire team, to obtain substantial speed-up.

The second part of this thesis focuses on CL under *communication* constraints, in particular, asynchronous communication and bandwidth constraints. Due to limited communication range or the presence of obstacles, robots communicate asynchronously, that is, they can only interact with different sub-teams over time and exchange information intermittently. For this scenario, we develop a family of resource-aware information-exchange rules for the robots, in order to ensure optimal and consistent localization performance. Lastly, this thesis investigates the problem of decentralized estimation under stringent communication-bandwidth constraints. Here, robots can communicate only

a severely quantized version (few or only one bit), of their real-valued sensor measurements, to the team. Existing estimation frameworks, however, are designed to process either real-valued or quantized measurements. To overcome this drawback, we propose a paradigm shift in estimation methodology by focusing on the design and performance evaluation of the first-ever, resource-aware, hybrid estimators. The proposed hybrid estimators are able to process both locally-available real-valued information, along with the quantized information received from the team, in order to maximize localization accuracy.

Finally, we note that mobile robot applications are no longer limited to specialized and expensive robots. Commonly-available hand-held devices such as cell phones, PDAs, and even cars, are equipped with processing, sensing, and networking capabilities. Therefore, when coupled with the proposed innovative, scalable, and resource-aware algorithms, these ubiquitous mobile devices can lead to a proliferation of novel location-based services.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

This thesis focuses on developing estimation algorithms for single and multi-robot localization under time-varying processing and communication constraints. In this section, we motivate the need for accurate robot localization, discuss the challenges arising due to resource constraints, and present an overview of the main research contributions of this work.

1.1 Robot Localization

Autonomous mobile robots have the potential to assist humans in a multitude of tasks ranging from construction [1, 2], transportation [3, 4], surveillance [5, 6], and environment monitoring [7], to aiding the visually-impaired [8, 9], search and rescue [10, 11] and space and underwater exploration [12, 13]. A prerequisite for the successful execution of these tasks is accurate robot *localization*, that is, determining the robot's own position and orientation (pose). As an example, consider robots assisting fire-fighters inside a collapsed building [14]. These robots need to accurately estimate their own pose, so that the information they provide about the locations of victims, hazardous materials, and collapsed structures will be reliable and useful to emergency personnel. Thus, precise robot localization is of critical importance since it enables effective interaction between the autonomous robots and their environment.

In most applications, mobile robots localize themselves by fusing measurements obtained from their onboard proprioceptive sensors (e.g., odometry or inertial measurement units [IMUs]) and exteroceptive sensors (e.g., GPS, cameras, and laser scanners). Localization using only proprioceptive measurements, known as dead reckoning, is inaccurate and suffers from unbounded error accumulation over time. This can be attributed to the integration of bias and noise contaminating these sensor measurements, effectively rendering the estimates unusable. To address this problem, exteroceptive sensors are used to "aid" the proprioceptive sensors and thus bound the estimation error. The presence of GPS signals significantly simplifies the localization problem by providing robot-position estimates of bounded uncertainty. However, in many representative operating environments, GPS measurements are often unreliable (e.g., in urban canyons, indoors, etc.) or unavailable (e.g., in space, underwater, inside caves, etc.). Moreover, the GPS accuracy, often in the order of meters, may be insufficient for high-precision applications, such as self-driving cars navigating in traffic. Therefore, this work focuses on developing accurate robot-localization algorithms in GPS-denied environments for the following two important application scenarios:

- Single-robot Simultaneous Localization and Mapping (SLAM): When a robot navigates within unknown areas (e.g., search and rescue or space exploration missions), it does not have access to an *a priori* map of its environment. In these scenarios, in order to determine its location with respect to its surroundings, the robot builds a map of the environment while simultaneously localizing itself within the same map. In this context, the map often comprises the positions of landmarks observed by the robot using its exteroceptive sensors, such as point features (corners) [15, 16], line segments (edges) [17], and planes (walls) [18], extracted from camera images or laser-scanner data. This process of joint estimation of the robot's pose and the positions of the landmarks observed in the environment, is known as SLAM [19, 20, 21, 22]. SLAM plays a crucial role in enabling long-term, autonomous navigation for the following two reasons: (i) By re-observing landmarks over multiple time steps (either consecutive time steps or when the robot revisits a previously traversed area, known as "loop closure"), the robot's pose estimation

errors remain bounded over long time periods [23], and (ii) under certain conditions[1] , the map generated by the robot can be used for navigation tasks such as path-planning around obstacles to reach goal locations.

- Multi-robot Cooperative Localization (CL): As opposed to a single robot, multi-robot teams provide the distinct advantages of robustness to single-point failures, versatile functionality, and improved speed of operation. However, in order to ensure the successful execution of collaborative tasks [24, 25, 26], along with each robot's individual pose, the *relative* poses of all robots in the team have to be accurately estimated. An approach, particularly suitable for achieving this in GPS-denied environments, is Cooperative Localization (CL) [27, 28], where the robots share information, either directly, or via a Fusion Center (FC). In CL, robots equipped with networking capabilities use all available sensor measurements from the team, including robot-to-robot relative measurements (such as distance, bearing, and orientation), to *jointly* estimate their poses, resulting in increased estimation accuracy for the entire team [29]. Moreover, when navigating in unknown environments, SLAM can be extended to multiple robots, known as multi-robot cooperative SLAM (C-SLAM), in order to generate a map of the environment and attain bounded localization errors for all robots in the team [30].

Over the past two decades, numerous solutions have been proposed for SLAM and CL that differ based on, mainly, the estimation algorithm used, such as the maximum likelihood estimator (MLE) [31, 32], the extended Kalman filter (EKF) [22, 28], the maximum a posteriori (MAP) estimator [33, 34], and the particle filter [35, 36]. Moreover, for CL, in conjunction with these estimators, three main system architectures have been proposed. In most cases, the centralized architecture [37, 32, 38] is used for CL, where each robot communicated its own proprioceptive and exteroceptive measurements to a leader robot, or a FC, for processing. However, since all computations are performed centrally, this approach is susceptible to single-point failures of the FC. Moreover, processing and memory resources are utilized sub-optimally, that is, the FC is responsible for all computations and data storage, while the other robots remain idle. To address

[1] The map is sufficiently dense to represent obstacles in the environment.

these issues, two main decentralized architectures have been employed: (i) The *multi-centralized* CL (MC-CL) [39] in which each robot acts as a FC, hence improving the robustness of the system, and (ii) The *distributed* CL [28] in which the processing load of the FC is reduced by distributing the computations among all robots in the team.

1.2 Resource-Constrained Robot Localization

One of the main challenges in designing an estimation algorithm for robot localization is operating under limited resources. Specifically, in many applications, it is often required that the robot localization algorithm operates efficiently in the background and generates real-time and accurate pose estimates with minimum resource utilization. Moreover, the resources available for localization may not be constant, but vary over time, depending on, for example, the current resource requirements of higher-level tasks and the battery life of the robots. Addressing these time-varying resource constraints poses significant challenges in the design of localization algorithms, as discussed below.

1.2.1 Processing Constraints

One of the main limitations of SLAM is its inherent lack of scalability for operating in large-scale environments. For example, the EKF-based SLAM is one of the most popular estimators for SLAM due to its ease of implementation and excellent numerical properties. Its computational complexity, however, is quadratic in the number of features in the map, limiting its application to small-sized areas. Moreover, state-of-art exteroceptive sensors, such as the Velodyne lidar (3D laser scanner) [40] and the Microsoft Kinect (RGB-D sensor) [41], provide large volumes of visual and depth data at very high frequencies. If these sensors are used for constructing dense area maps, the EKF-based SLAM's computational requirements will soon exceed the robot's processing capabilities.

A similar problem arises in CL, when processing the robot-to-robot relative measurements. Even though CL does not involve map-building, the computational complexity of processing the robot-to-robot relative measurements, in standard centralized EKF-based CL, in the worst case, is quartic in the number of robots in the team. Therefore,

for large teams, or even small teams with limited processing capabilities, these computational requirements can easily overwhelm the team's resources and prohibit real-time operation. One possible solution is to distribute the processing of the estimation algorithm among the robots themselves, resulting in reduced computational cost *per robot*. CL, however, is a *joint-state estimation* problem, that is, the pose estimates of all robots are correlated due to the processing of relative robot-to-robot measurements. Therefore, distributed algorithms have to ensure that these correlations are correctly handled, in order to ensure consistent estimation.[2] Moreover, these algorithms may incur additional communication overhead since intermediate results may also need to be exchanged between the robots. These requirements often lead to a trade-off between localization accuracy and communication constraints that need to be considered when designing a distributed CL algorithm.

Communication Constraints

For mobile robot teams, asynchronous communication and bandwidth limitations often guide the design of estimation algorithms for CL. This is due to the fact that each robot has to share its own information (e.g., in the form of its pose estimate and covariance, or its exteroceptive and proprioceptive measurements) with other robots in the team [28], [43]. Robots participating in tightly-coupled tasks (e.g., transporting a large piece of equipment) can communicate *synchronously* and transmit/receive information to/from the entire team; that is, the communication graph associated with the underlying mobile robot network is connected.[3] In contrast, during exploration tasks and depending on the robots' motion, communication range, and presence of obstacles, each robot may communicate *asynchronously* with a different sub-team at every time step; that is, the underlying communication graph can be both time-varying and not complete. A key challenge for robot localization under asynchronous communication constraints is to keep track of the information flow within the robot network *over time*, mainly for two reasons: (i) If information (e.g., measurements) is unknowingly double-counted (processed more than once by the estimator), it may result into overly

[2] A state estimator is consistent if the estimation errors are zero-mean and have covariance matrix smaller or equal to the one calculated by the filter/smoother [42].

[3] Each robot can communicate with all other robots in the team.

optimistic and inconsistent estimates, and (ii) If unprocessed information is mistakenly discarded, it will result in suboptimal localization performance.

Aggravating this problem further is the issue of bandwidth limitations. Even for robots within communication range, factors such as: (i) inherent communication-bandwidth limitations in underwater applications, resulting from diffraction, attenuation, scattering, and fading, or (ii) constraints imposed on data transmission to conserve battery power and increase the robots' operating time, may prevent robots from communicating all necessary information. Moreover, while in many cases, robots may be able to select and transmit all, or a subset, of their analog[4] measurements, depending upon the bandwidth-availability, applications with severe communication bandwidth constraints (e.g., underwater), restrict each robot to transmiting only a *few bits* per analog measurement. In such cases, every robot is forced to carry out *lossy quantization* of its measurements and commonly-used estimators (e.g., EKF or MAP) have to be modified in order to process quantized, instead of analog, observations.

1.3 Research Objectives

While resource-constrained robot localization has been studied over the past few years, existing approaches have focused on developing estimators tailored to a particular application, with fixed processing and communication resources. Consequently, these estimators can neither exploit additional resources to improve localization accuracy, nor can they adapt to reduced resource availability; that is, they lack resource-awareness. Hence, a new estimator has to be designed for every new task, resulting in substantial increase in development time and costs. In addition, some of the existing estimators introduce *ad hoc* approximations to meet resource constraints and, thus, are inherently incapable of providing any performance guarantees. These shortcomings have significantly increased costs and hindered the deployment of robot teams. The key objective of this thesis work is to address these drawbacks by designing application-independent, resource-aware, and consistent estimators for mobile robot localization. By treating resource constraints as time-varying parameters, instead of predefined constants, these

[4] Sensors sample a process and provide a measurement which is often represented in digital form using 32 or 64 bit floating-point number representation. We refer to such measurements as analog.

estimators can adjust online to the availability of processing and communication resources. Additionally, a key advantage of these estimators is their portability; the same resource-aware estimator can now be used in various robots performing diverse tasks, resulting in substantial savings in development time and costs. Moreover, even during a particular task, the estimator can optimally handle fluctuating resource availability and maximize localization accuracy. To achieve this objective, the four main contributions of this work are as follows:

1.3.1 SLAM under processing constraints

The first key contribution of this work is the development of resource-aware and consistent estimators for SLAM. Specifically, we present two approaches: (i) Power SLAM, based on the EKF, and (ii) Constrained Keyframe-based Localization and Mapping (C-KLAM), based on the MAP estimator.

As opposed to the quadratic computational complexity of the standard EKF-based SLAM, the proposed Power SLAM is an approximate and consistent approach with complexity only *linear* in the number of features in the map. While existing approximate approaches discard information at every time step and/or introduce *ad hoc* approximations with no performance guarantees, Power SLAM exploits the structure of the EKF's covariance update to develop a systematic technique for minimizing information loss over multiple time steps. Moreover, Power SLAM is an anytime algorithm that can adjust its processing requirements online based on the availability of time-varying computational resources, and quantify the information loss due to the approximations employed.

The second algorithm that we introduce is the C-KLAM, an approximate but consistent MAP-based SLAM approach. Specifically, although, the MAP estimator can improve the robot-trajectory and map-estimation accuracy by acting as a *smoother* and reducing the linearization errors, its worst-case computational complexity is $O(K^3N^3)$, where K and N denote the number of robot poses and mapped landmarks, respectively. C-KLAM reduces the computational requirements of MAP-based SLAM by only estimating a sparse map, consisting of only the distinctive landmarks necessary for loop-closure detection and the key robot poses that observed them. A key advantage of C-KLAM, as opposed to existing approaches, is that information from non-key poses

and non-key landmarks is not discarded. Instead, it is used (through proper marginalization) to generate consistent constraints between the key poses, hence improving the estimation accuracy without significantly affecting the processing cost.

1.3.2 Distributed CL under processing constraints

The second key contribution of this work is a fully-distributed, resource-aware, and consistent batch MAP-based algorithm for CL. As opposed to centralized MAP-based CL, the proposed algorithm introduces a novel distributed data processing scheme that harnesses the computational and storage resources of all robots in the team in order to speed up the processing, while generating pose estimates identical to centralized CL (that uses all available measurement information). Moreover, we present a resource-aware, approximate extension of this approach, that is capable of trading accuracy for processing/communication bandwidth. Lastly, we carry out detailed computational and communication complexity analysis for every step of the proposed distributed CL algorithm, so as to quantify its processing gain and communication overhead.

1.3.3 Asynchronous Multi-Centralized CL

The third key contribution of this work is a generalized framework of information-transfer schemes for performing consistent MC-CL under asynchronous communication and bandwidth limitations. The proposed information-transfer schemes, which differ based on their bandwidth requirements per communication link, are independent of the estimation algorithm used and enable each robot to generate pose estimates identical to centralized CL, albeit delayed. Moreover, these schemes depend upon a design parameter, q, by varying which, a family of information-transfer schemes can be generated that trade communication-bandwidth requirements per link (increasing with q) for time delays in obtaining the centralized-equivalent estimates (decreasing with increasing q). Moreover, to facilitate the selection of a suitable scheme for an application, we: (i) present analytical results for the expected time-delay in obtaining centralized-equivalent estimates, (ii) derive the necessary and sufficient conditions on the communication-graph connectivity, and (iii) carry out a detailed communication-complexity analysis for each of the proposed information-transfer schemes.

1.3.4 Hybrid Estimation Framework for bandwidth-constrained CL

The last key contribution of this work focuses on applications with stringent communication bandwidth constraints (e.g., underwater), where robots can communicate *only a few bits* per analog measurement. In this scenario, along with the quantized measurements communicated by other robots in the team, each robot also has access to its own local analog measurements. Existing estimators, however, can process either analog measurements (e.g., EKF and MAP) or quantized measurements (e.g., Sign-of-Innovation Kalman filter [44], Batch and Iteratively Quantized Kalman filter [39], and Batch and Iteratively Quantized MAP estimator [45]). For this reason, robots can utilize only a part of the available measurement information, resulting in suboptimal localization performance. To address this limitation, we propose a paradigm shift in estimation methodology by introducing a hybrid estimation framework for processing *both* analog and quantized measurements. Specifically, we develop optimal encoding and decoding rules for MMSE and MAP-based hybrid estimators for CL, resulting in a localization system with substantially improved estimation accuracy. Lastly, we note that the proposed hybrid estimators are resource-aware, and can handle the *general* case of *time-varying* communication-bandwidth availability, that is, when robots can communicate $n \geq 1$ bits per analog measurement.

1.4 Organization of the manuscript

Chapters 2 and 3 present our work on SLAM under processing constraints, for the proposed Power SLAM and C-KLAM algorithms, respectively. The derivations and complexity analysis for these algorithms are described, and their use is demonstrated in real-world experiments. Chapters 4–7 focus on our work on CL under processing and communication constraints. Specifically, details of the distributed MAP-based CL approach, along with experimental results, are presented in Chapter 4. Chapter 5 presents our work on MC-CL under asynchronous communication and bandwidth constraints. Chapters 6 and 7 present the detailed derivations as well as simulation and experimental results for the hybrid MMSE and hybrid MAP estimators, respectively. Finally, Chapter 8 summarizes the key results of this thesis and identifies interesting directions for future research.

Chapter 2

Power-SLAM: A linear-complexity, anytime algorithm for EKF-based SLAM

In this chapter, we present Power-SLAM, an approximate EKF-based estimator for SLAM. As opposed to the standard EKF-based SLAM, which has computational complexity quadratic in the number of features in the map, the proposed Power-SLAM has processing requirements that are only *linear* in the number of features in the map. Moreover, in addition to being conservative as compared to the standard EKF-based SLAM, Power-SLAM has processing requirements that can be adjusted to the availability of computational resources. Parts of this chapter have been published in [46, 47].

2.1 Introduction

One of the most challenging problems faced in autonomous navigation is Simultaneous Localization and Mapping (SLAM), where robots jointly estimate their own pose (i.e., position and orientation) and model the environment. Mobile robot tasks such as search and rescue missions, and space and underwater exploration are classical examples of SLAM, where (i) the robots do not have access to global positioning devices, such as GPS, or information from such sources is unreliable (e.g., in urban environments or

underwater), and (ii) *a priori* information about the environment (e.g., a map) is not available to the robot.

SLAM has been studied extensively in the literature and numerous solutions have been proposed. These solutions differ primarily in the assumptions made for the environment (static or dynamic), the map representation (point/line/plane features, global or robocentric mapping, etc.), the robot's sensors (laser scanners, cameras, etc.) and the estimation framework used (EKF, PF, MAP estimator, etc.).

Amongst these, probably the most commonly used estimator for SLAM is the EKF, due to its ease of implementation. The EKF is the optimal MMSE estimator, up to linearization errors. Moreover, it recursively computes the covariance matrix that not only provides a concrete measure of the *uncertainty* in the state estimates but also maintains the cross-correlations between the robot's and the landmarks' estimates. Thus, the covariance matrix provides crucial information necessary for minimizing the risk of failure while making decisions related to data association and path planning.

Unfortunately, storing and updating this covariance matrix in EKF-based SLAM is a major bottleneck. Even under the assumption that only few map features are detected at each time step, both the memory and computational requirements of EKF-based SLAM are quadratic, $O(N^2)$, in the number of features, N, in the map. While storage requirements can be handled efficiently by the memory devices available today, the computational complexity has prevented the deployment of mobile robots in large-scale environments. Another critical drawback of using the EKF, resulting from the linearization of the non-linear process and measurement models prevalent in SLAM, is its inherent inconsistency over time. Estimator consistency is vital for SLAM because an inconsistent estimator provides no guarantee for the accuracy of the generated state estimates, hence rendering the robot/landmark estimates unreliable.

As detailed in the following section, a number of EKF-based approaches exist that address the computational complexity of SLAM by: (i) delaying the quadratic covariance update step, or (ii) employing an approximate structure for the estimator. The main limitation of the methods under the first category, is that inevitably, at some point, the delayed covariance update will have to be carried out, incurring a computational cost of $O(N^2)$. For large values of N, this can become prohibitive. On the other hand, many of the approximate approaches do not maintain the cross-correlations between the

robot's and the landmarks' estimates, which can lead to inconsistency and divergence of the EKF[1] . Furthermore, amongst the approximate approaches that do maintain these correlations, information is discarded during *every* time step, often based on criteria that do *not* guarantee the best use of the available CPU cycles, thus resulting into suboptimal estimators.

To address this problem, in this work we introduce an EKF-based algorithm for SLAM with *linear*, in the number of features in the map, computational complexity. The proposed conservative[2] approximate estimator *minimizes the information discarded* over *multiple time steps*. This is achieved by: (i) extending the time horizon over which approximations are invoked by using the Global Map Postponement technique [see Section 2.3.2], and (ii) using the Power method to compute and retain, after each approximation, only the *most informative* updates (i.e., Kalman vectors whose outer product minimizes the trace of the covariance matrix) [see Section 2.3.3]. Finally, in order to speed up the rate of convergence of the proposed estimator, rank-2 covariance updates, that minimize the trace of the covariance matrix under the linear computational complexity constraint, are applied at every time step [see Section 2.3.4]. The proposed approach is flexible in the sense that the parameters involved at each stage of the algorithm can be adjusted to meet the availability of computational resources. Before presenting the details of the Power-SLAM algorithm, we briefly review some of the representative EKF-based SLAM approaches.

2.2 Related Work

Although numerous approaches exist for reducing the computational complexity of SLAM (e.g., PF [35], thin junction trees [48], treemaps and multigrids [49, 50], square root SAM [33], etc.), we hereafter limit our discussion of related work to approaches based on the EKF.

In their seminal work, Smith, Self, and Cheeseman [19] introduced the concept of the stochastic map and proved that the features' and robot's estimates are not independent,

[1] Note the difference between the two causes of estimator inconsistency resulting due to: (i) linearization of non-linear models, and (ii) incorrect handling of cross-correlations in the covariance matrix.

[2] An approximate EKF-based SLAM estimator is conservative if its estimated covariance is larger than that of the corresponding standard EKF.

as was previously assumed. By using an EKF-based estimator for solving the SLAM problem, Moutarlier and Chatila [20] showed that the complete covariance matrix for both the robot and the features must be maintained in order to avoid inconsistency of the EKF filter. Reducing the computational burden for real-time application of SLAM has been studied by numerous researchers and the proposed EKF-based solutions can be classified into two main categories:

2.2.1 Standard EKF-based SLAM

By restructuring the standard EKF equations, Davison [51] showed that it is possible to process multiple observations of the *same* landmark (map feature) in constant time, while delaying the complete update of the covariance matrix. Knight *et al.* [52] extended this idea to the case of sub-maps. As in [51], covariance and state updates are limited to the sub-map (i.e., constant-time complexity) until the robot moves outside that particular area. When this happens, the whole map needs to be updated which requires $O(N^2)$ operations. Similarly in [53], new sub-maps, each with p features, are initialized at various locations along the trajectory. As long as $p \ll N$, where N is the total number of features in the global map, each sub-map can be updated in constant time (i.e., quadratic, $O(p^2)$, in the number of features in this particular sub-map). Once all the sub-maps are merged, however, the computational cost again becomes quadratic, $O(N^2)$, in the *total* number of features. Paz *et al.* present a "divide and conquer" EKF-based SLAM approach in [54] that uses sub-maps. Specifically, the authors describe an efficient map-joining algorithm so that the computational complexity of EKF-based SLAM is reduced to $O(N)$ from $O(N^2)$ per time step. While this is true for the exploration phase of SLAM, if the robot traverses the map again, the cost of the proposed approach becomes the same as that of standard EKF-based SLAM.

2.2.2 Approximate EKF-based SLAM

The method described in [55] and [56] retains the standard structure of the EKF-based SLAM algorithm but reduces the number of landmarks considered per update step. This is achieved by selecting and processing only the most informative features (based on their covariance); the remaining features are removed from the state vector. Although

this algorithm correctly maintains correlations between the robot and the landmarks, it introduces an approximation since not all available map features are processed.

Leonard and Feder [57] introduced the concept of multiple overlapping sub-map regions (Decoupled Stochastic Maps), each with its own stochastic map. Their approach scales the EKF-based SLAM algorithm to linear computational complexity. However, there exists no proof for the consistency of this method and it is not possible to estimate the impact of the approximation on the map's uncertainty.

In the relative-map approach presented in [58] the relative, instead of the absolute, positions of the features are estimated. By excluding the vehicle pose estimate from the state vector, the covariance matrix takes on a simple block-diagonal structure. Hence, the resulting computational complexity for processing each observation becomes constant time. A drawback of this method is that it does not ensure relative map consistency[3] . The Geometric Projection filter [59] can be used to impose the consistency constraint, but it increases the computational burden to $O(C^3)$, where C is the number of independent constraints that need to be applied. These constraints have to be imposed every time the robot pose is required. Also, this method lacks a common frame of reference and thus it cannot provide a direct update to the robot pose.

Guivant and Nebot's Compressed EKF (CKF) approach [60] combines the ideas of sub-maps and relative maps. By using sub-maps, this algorithm has complexity $O({N_a}^2)$, where N_a is the number of features in the local map. As in the case of [51, 52], it postpones the global update which can be carried out with the complexity of a full SLAM update. While this algorithm, in its optimal form has $O(N^2)$ complexity, an approximation was introduced that involves relative maps and operates in linear time. In this case, only a subset of the map features are updated.

Julier and Uhlmann introduced the Covariance Intersection (CI) method [61] which does not consider the estimates' correlations. Although this estimator is conservative and its computational requirements scale linearly with the number of features, it has very slow convergence. When partial correlation information is available, the Split CI (SCI) [62] can be employed. This method works better than CI but does not use the complete correlation information and is still as conservative as CI for the robot

[3] A relative map is consistent if all possible transformations to an absolute map yield unique and unambiguous absolute landmark locations [59].

estimates.

The Sparse Weight Kalman Filter (SWKF) [63] approach proposed by Julier relies on the sparsification of the Kalman gain matrix. Based on the observation that most of its elements are significantly smaller as compared to the ones corresponding to the robot pose and the observed landmark, these are set to zero. The resulting approximate algorithm has linear computational complexity but generates very conservative estimates.

In our approach, we first present the Global Map Postponement (GMP) technique that reformulates and extends the postponement method [51, 52] to the case of the *global* map. We show that by using the GMP, the computational cost of the *exact* EKF-based SLAM remains *linear* in the number of states, N, as long as the number of delayed updates (or equivalently the number of stored Kalman vectors), m, is significantly smaller than N. However, as the robot moves around in the environment and re-observes landmarks, the number of delayed updates, m, increases. In order to ensure that $m \ll N$ (hence maintain linear computational complexity), we employ a low-rank *approximation* that uses the Power method [64] for computing the largest eigenvalues and the corresponding eigenvectors in linear time. This technique retains the most informative of the Kalman vectors and allows us to extend the postponement horizon indefinitely. Finally, in order to speed up the convergence of our proposed estimator, linear-cost, rank-2 updates, that minimize the trace of the covariance matrix, are imposed at every time step.

2.3 Algorithm Description

2.3.1 Standard EKF-based SLAM

This section introduces the notation used in this chapter and briefly describes the EKF-based SLAM equations in 2D. Note that our proposed approach can be easily extended to 3D. The state vector, $\mathbf{x}_k$, consists of:

$$\mathbf{x}_k = \left[\mathbf{x}_{r_k}^T \ , \mathbf{p}_{1_k}^T \ , \mathbf{p}_{2_k}^T, \ldots, \mathbf{p}_{N_k}^T\right]^T . \tag{2.1}$$

Here, $\mathbf{x}_{r_k} = [x_{r_k}, y_{r_k}, \phi_{r_k}]^T$, denotes the position and orientation of the robot and $\mathbf{p}_{i_k} = [x_{i_k}, y_{i_k}]^T$ denotes the position of the i^{th} landmark, $i = 1, \ldots, N$, at time-step k. All

the above quantities are expressed with respect to a global frame of reference.

The robot is equipped with proprioceptive (odometry) sensors that provide linear, v_{m_k}, and rotational, ω_{m_k}, velocity measurements. The robot's motion model is given by:

$$\mathbf{x}_{r_{k+1}} = \mathbf{f}(\mathbf{x}_{r_k}, \mathbf{u}_{m_k} - \mathbf{w}_k), \tag{2.2}$$

where $\mathbf{f}$ is in general a non-linear function and $\mathbf{u}_{m_k} = [v_{m_k}, \omega_{m_k}]^T$ is the measured control input. The vector $\mathbf{w}_k = [w_{v_k}, w_{\omega_k}]^T$, with covariance $\mathbf{Q}_k$, represents the zero-mean, white Gaussian noise in the linear and angular velocity measurements, respectively. Additionally, the robot is equipped with exteroceptive sensors that allow it to measure the distance and bearing to landmark i. The robot's measurement model is given by:

$$\mathbf{z}_{k+1} = \mathbf{h}(\mathbf{x}_{r_{k+1}}, \mathbf{p}_{i_{k+1}}) + \mathbf{n}_{k+1} \tag{2.3}$$

with $\mathbf{h} = [d_{k+1}, \theta_{k+1}]^T$, where d_{k+1} and θ_{k+1} are the true distance and bearing from the robot to landmark i and $\mathbf{n}_{k+1} = [n_{d_{k+1}}, n_{\theta_{k+1}}]^T$, is the additive zero-mean white Gaussian measurement noise with covariance $\mathbf{R}_{k+1}$.

Propagation

The propagation equations for the robot and landmarks' state estimates are given by:

$$\hat{\mathbf{x}}_{r_{k+1|k}} = \mathbf{f}(\hat{\mathbf{x}}_{r_{k|k}}, \mathbf{u}_{m_k}) \tag{2.4}$$

$$\hat{\mathbf{p}}_{i_{k+1|k}} = \hat{\mathbf{p}}_{i_{k|k}}, \quad i = 1, \ldots, N \tag{2.5}$$

where $\hat{\mathbf{m}}_{j_{l|p}}$ denotes the estimates of the random vector $\mathbf{m}_j$ at time-step l, given all the measurements up to time-step p. Furthermore, (2.5) results from the assumption that the landmarks are stationary. The covariance propagation equation is given by:

$$\mathbf{P}_{k+1|k} = \mathbf{\Phi}_k \mathbf{P}_{k|k} \mathbf{\Phi}_k^T + \mathbf{G}_k \mathbf{Q}_k \mathbf{G}_k^T, \tag{2.6}$$

where $\mathbf{P}$ is the symmetric state covariance matrix with the following structure[4] :

$$\mathbf{P} = \begin{bmatrix} \mathbf{P}_{\mathbf{x}_r\mathbf{x}_r} & \mathbf{P}_{\mathbf{x}_r\mathbf{p}_1} & . & . & . & \mathbf{P}_{\mathbf{x}_r\mathbf{p}_N} \\ \mathbf{P}_{\mathbf{p}_1\mathbf{x}_r} & \mathbf{P}_{\mathbf{p}_1\mathbf{p}_1} & . & . & . & \mathbf{P}_{\mathbf{p}_1\mathbf{p}_N} \\ . & . & . & & & . \\ . & . & & . & & . \\ . & . & & & . & . \\ \mathbf{P}_{\mathbf{p}_N\mathbf{x}_r} & \mathbf{P}_{\mathbf{p}_N\mathbf{p}_1} & . & . & . & \mathbf{P}_{\mathbf{p}_N\mathbf{p}_N} \end{bmatrix} \tag{2.7}$$

and

$$\mathbf{\Phi}_k = \begin{bmatrix} \mathbf{\Phi}_{rk} & \mathbf{0}_{3\times 2N} \\ \mathbf{0}^T_{3\times 2N} & \mathbf{I}_{2N} \end{bmatrix}, \mathbf{G}_k = \begin{bmatrix} \mathbf{G}_{rk} \\ \mathbf{0}_{2N\times 2} \end{bmatrix}. \tag{2.8}$$

Here,

$$\begin{aligned} \mathbf{\Phi}_{rk} =& \nabla_{\mathbf{x}_r}(\mathbf{f}(\hat{\mathbf{x}}_{r_{k|k}}, \mathbf{u}_{m_k})) \\ \mathbf{G}_{rk} =& \nabla_{\mathbf{w}}(\mathbf{f}(\hat{\mathbf{x}}_{r_{k|k}}, \mathbf{u}_{m_k})) \end{aligned}$$

and $\mathbf{I}_{2N}$ is the $2N \times 2N$ identity matrix.

Update

The estimates for the robot's distance and bearing measurements to landmark i, at time-step $k+1$, are given by:

$$\hat{\mathbf{z}}_{k+1|k} = \mathbf{h}(\hat{\mathbf{x}}_{r_{k+1|k}}, \hat{\mathbf{p}}_{i_{k+1|k}}) = \mathbf{h}(\hat{\mathbf{x}}_{k+1|k}). \tag{2.9}$$

Once the actual landmark measurement, $\mathbf{z}_{k+1}$, is obtained, the state and covariance are updated as follows:

$$\mathbf{r}_{k+1} = \mathbf{z}_{k+1} - \hat{\mathbf{z}}_{k+1|k} \tag{2.10}$$

$$\mathbf{S}_{k+1} = \mathbf{H}_{k+1}\mathbf{P}_{k+1|k}\mathbf{H}^T_{k+1} + \mathbf{R}_{k+1} \tag{2.11}$$

$$\mathbf{K}_{k+1} = \mathbf{P}_{k+1|k}\mathbf{H}^T_{k+1}\mathbf{S}^{-1}_{k+1} \tag{2.12}$$

$$\hat{\mathbf{x}}_{k+1|k+1} = \hat{\mathbf{x}}_{k+1|k} + \mathbf{K}_{k+1}\mathbf{r}_{k+1} \tag{2.13}$$

$$\mathbf{P}_{k+1|k+1} = \mathbf{P}_{k+1|k} - \mathbf{K}_{k+1}\mathbf{S}_{k+1}\mathbf{K}^T_{k+1} \tag{2.14}$$

[4] The time subscripts are omitted here to simplify the presentation.

where the measurement matrix, $\mathbf{H}_{k+1}$, is given by:

$$\mathbf{H}_{k+1} = [\mathbf{H}_r \ \mathbf{0}_{2\times 2(i-1)} \ \mathbf{H}_i \ \mathbf{0}_{2\times 2(N-i)}] \tag{2.15}$$

and $\mathbf{H}_r = \nabla_{\mathbf{x}_r}(\mathbf{h}(\hat{\mathbf{x}}_{k+1|k}))$, $\mathbf{H}_i = \nabla_{\mathbf{p}_i}(\mathbf{h}(\hat{\mathbf{x}}_{k+1|k}))$. Here, the quantities $\mathbf{r}_{k+1}$, $\mathbf{S}_{k+1}$, $\mathbf{K}_{k+1}$, $\hat{\mathbf{x}}_{k+1|k+1}$, and $\mathbf{P}_{k+1|k+1}$ denote the measurement residual vector, the residual covariance matrix, the Kalman gain matrix, the updated state vector, and the updated covariance matrix respectively, at time-step $k+1$. Table 2.1 lists the dimensions of the various quantities that appear in the EKF-based SLAM formulation.

Table 2.1: Dimensions of terms appearing in EKF-based SLAM

Terms in EKF-based SLAM		Dimension
State vector	$\mathbf{x}_k$	$(2N+3)\times 1$
State Covariance	$\mathbf{P}_k$	$(2N+3)\times(2N+3)$
Jacobian of $\mathbf{f}$ w.r.t. $\mathbf{x}_r$	$\mathbf{\Phi}_{rk}$	3×3
Jacobian of $\mathbf{f}$ w.r.t. $\mathbf{w}$	$\mathbf{G}_{rk}$	3×2
State Transition matrix	$\mathbf{\Phi}_k$	$(2N+3)\times(2N+3)$
Jacobian of $\mathbf{h}$ w.r.t. $\mathbf{x}_r$	$\mathbf{H}_r$	2×3
Jacobian of $\mathbf{h}$ w.r.t. $\mathbf{p}_i$	$\mathbf{H}_i$	2×2
Measurement matrix	$\mathbf{H}_{k+1}$	$2\times(2N+3)$
Measurement Residual	$\mathbf{r}_{k+1}$	2×1
Residual Covariance	$\mathbf{S}_{k+1}$	2×2
Kalman Gain	$\mathbf{K}_{k+1}$	$(2N+3)\times 2$

The $O(N^2)$ computational complexity of EKF-based SLAM arises due to the covariance update step [see (2.14)], which involves the multiplication of the Kalman gain matrix (dimensions $(2N+3)\times 2$) with its transpose [see (2.12)]. For robots involved in mapping of dense environments, N, i.e., the number of landmarks, continuously increases, and hence the cost of updating the covariance matrix can prohibit real-time performance. In order to overcome this computational bottleneck, we now present the details of our proposed algorithm in the next three sections.

2.3.2 Global Map Postponement SLAM

The Global Map Postponement (GMP) approach forms the basis of our proposed algorithm. In this section, we demonstrate that under the GMP approach, the computational complexity of EKF-based SLAM is $O(mN)$ per time step, where m is the number of delayed state updates. Therefore, while $m \ll N$, the computational complexity of EKF-based SLAM remains linear in N. This is an important result that will be exploited later in this chapter for developing the Power-SLAM algorithm.

Contrary to the approaches of [51, 52] that employ postponement only when the robot operates within small areas (sub-maps), we hereafter present our GMP method which poses no restrictions on the motion of the robot. Consider the case where at time-step $k+1$, a new landmark observation is processed to update the covariance. In GMP, (2.14) is reformulated as:

$$\mathbf{P}_{k+1|k+1} = \mathbf{P}_{k+1|k} - (\mathbf{K}_{k+1}\mathbf{S}_{k+1}^{1/2})(\mathbf{K}_{k+1}\mathbf{S}_{k+1}^{1/2})^T, \tag{2.16}$$

where $\mathbf{S}^{1/2}$ is a lower-triangular matrix obtained from the Cholesky factorization of $\mathbf{S}$. Since $\mathbf{S}$ is a 2×2 matrix, this Cholesky factorization is carried out in constant time. The dimensions of the resulting term, $(\mathbf{K}_{k+1}\mathbf{S}_{k+1}^{1/2})$, are $(2N+3) \times 2$ and this matrix is split into two vectors, $\mathbf{k}_1$, $\mathbf{k}_2$, each of dimensions $(2N+3) \times 1$. This gives us:

$$\mathbf{P}_{k+1|k+1} = \mathbf{P}_{k+1|k} - \sum_{i=1}^{2} \mathbf{k}_i \mathbf{k}_i^T, \tag{2.17}$$

where $\mathbf{k}_i$ is the i^{th} column of $\mathbf{K}_{k+1}\mathbf{S}_{k+1}^{1/2}$ and furthermore $\mathbf{k}_i = \sqrt{\lambda_i}\mathbf{v}_i$, where $\mathbf{v}_i$ is the eigenvector corresponding to the eigenvalue, λ_i, of $\mathbf{K}_{k+1}\mathbf{S}_{k+1}\mathbf{K}_{k+1}^T$. Here, it is important to note that in the GMP approach, the vector outer-product sum, $\sum_{i=1}^{2} \mathbf{k}_i\mathbf{k}_i^T$, is *never* computed. Instead the Kalman vectors, $\mathbf{k}_i$, are stored for later processing[5] .

Maintaining this sum of vector outer-products forms the framework of GMP and allows us to introduce approximations that reduce the computational complexity of EKF-based SLAM. Therefore for clarity, we divide the discussion on GMP into two parts: (i) first we demonstrate how this structure can be maintained through subsequent

[5] Throughout this chapter, we refer to these vectors as "Kalman" vectors. While this is true for the ones appearing during updates (though they are scaled and rotated), we also extend this definition to describe vectors that result later on from the low-rank approximation and/or after sparsifications.

propagation and update steps, and then (ii) we present the computational complexity of GMP, $O(mN)$, for propagation, update, and landmark initialization.

1. *Propagation*: The covariance propagation equation at the next time step, i.e., time-step $k+2$, is given by [see (2.6), (2.17)]:

$$\begin{aligned}\mathbf{P}_{k+2|k+1} &= \mathbf{\Phi}_{k+1}\mathbf{P}_{k+1|k}\mathbf{\Phi}_{k+1}^T + \mathbf{G}_{k+1}\mathbf{Q}_{k+1}\mathbf{G}_{k+1}^T \\ &\quad - \sum_{i=1}^{2}(\mathbf{\Phi}_{k+1}\mathbf{k}_i)(\mathbf{\Phi}_{k+1}\mathbf{k}_i)^T \\ &= \mathbf{P}_{k+2|k} - \sum_{i=1}^{2}\mathbf{k}_i^*\mathbf{k}_i^{*T}, \end{aligned} \tag{2.18}$$

   where $\mathbf{k}_i^* = \mathbf{\Phi}_{k+1}\mathbf{k}_i$. At this step, in the GMP approach the quantities $\mathbf{P}_{k+2|k}$ and $\mathbf{k}_i^*$ are evaluated but the vector product, $\mathbf{k}_i^*\mathbf{k}_i^{*T}$, is not computed. Instead the vectors, $\mathbf{k}_i^*$, are stored for later processing.

2. *Update*: The residual covariance is given by [see (2.11), (2.18)]:

$$\begin{aligned}\mathbf{S}_{k+2} &= \mathbf{H}_{k+2}\mathbf{P}_{k+2|k}\mathbf{H}_{k+2}^T + \mathbf{R}_{k+2} \\ &\quad - \sum_{i=1}^{2}(\mathbf{H}_{k+2}\mathbf{k}_i^*)(\mathbf{H}_{k+2}\mathbf{k}_i^*)^T. \end{aligned}$$

   Here, $\mathbf{S}_{k+2}$ is evaluated, i.e., the vector outer-product sum in the above equation is explicitly calculated. Next, the Kalman gain is expressed as [see (2.12), (2.18)]:

$$\mathbf{K}_{k+2} = \mathbf{P}_{k+2|k}\mathbf{H}_{k+2}^T\mathbf{S}_{k+2}^{-1} - \sum_{i=1}^{2}\mathbf{k}_i^*(\mathbf{H}_{k+2}\mathbf{k}_i^*)^T\mathbf{S}_{k+2}^{-1}.$$

   Here again, $\mathbf{K}_{k+2}$ is evaluated completely. Once the Kalman gain matrix is obtained, the state update [see (2.13)] is carried out. Finally, the covariance update is expressed as [see (2.14), (2.17), (2.18)]:

$$\begin{aligned}\mathbf{P}_{k+2|k+2} &= \mathbf{P}_{k+2|k} - \sum_{i=1}^{2}\mathbf{k}_i^*\mathbf{k}_i^{*T} - \mathbf{K}_{k+2}\mathbf{S}_{k+2}\mathbf{K}_{k+2}^T \\ &= \mathbf{P}_{k+2|k} - \sum_{i=1}^{2}\mathbf{k}_i^*\mathbf{k}_i^{*T} - \sum_{i=1}^{2}\mathbf{k}_i\mathbf{k}_i{}^T \\ &= \mathbf{P}_{k+2|k} - \sum_{i=1}^{4}\mathbf{k}_i\mathbf{k}_i{}^T. \end{aligned} \tag{2.19}$$

Note that, in (2.19), for simplifying the notation, we have set $\mathbf{k}_3 = \mathbf{k}_1$, $\mathbf{k}_4 = \mathbf{k}_2$ and $\mathbf{k}_1 = \mathbf{k}_1^*$, $\mathbf{k}_2 = \mathbf{k}_2^*$.

As evident from (2.18), (2.19), the vector outer-product sum structure in the GMP is preserved for subsequent propagation and update steps. Specifically, from (2.19), we see that after each update step, two additional Kalman vectors are generated. Therefore, repeating this process for all subsequent propagation and updates, at time-step $k+m$, the covariance update equation has the form:[6]

$$\mathbf{P}_{k+m|k+m} = \mathbf{P}_{k+m|k} - \sum_{i=1}^{2m} \mathbf{k}_i \mathbf{k}_i^T. \tag{2.20}$$

Maintaining this structure, we hereafter present the complexity analysis for GMP SLAM and show that propagation, update, and landmark initialization can be performed with computational cost $O(mN)$.

Propagation

The covariance propagation equation at time-step $k+m+1$ is given by [see (2.6), (2.20)]:

$$\begin{aligned}\mathbf{P}_{k+m+1|k+m} &= \mathbf{\Phi}_{k+m}\mathbf{P}_{k+m|k}\mathbf{\Phi}_{k+m}^T + \mathbf{G}_{k+m}\mathbf{Q}_{k+m}\mathbf{G}_{k+m}^T \\ &\quad - \sum_{i=1}^{2m}(\mathbf{\Phi}_{k+m}\mathbf{k}_i)(\mathbf{\Phi}_{k+m}\mathbf{k}_i)^T \\ &= \mathbf{P}_{k+m+1|k} - \sum_{i=1}^{2m}\mathbf{k}_i^*\mathbf{k}_i^{*T},\end{aligned} \tag{2.21}$$

where $\mathbf{P}_{k+m+1|k}$ is the propagated covariance at time-step $k+m+1$ given measurements up to time-step k. As in standard EKF-based SLAM, $\mathbf{P}_{k+m+1|k}$ is computed in linear time. Due to the special block-diagonal structure of the $\mathbf{\Phi}_{k+m}$ matrix [see (2.8)], each $\mathbf{k}_i^*$ can be calculated in constant time. Since $2m$ such Kalman vectors have to be calculated, the overall computational complexity of this step is also constant time, i.e., $O(m)$.

[6] For simplicity, we assume one measurement per time step, i.e. one update step at every time step.

Update

The residual covariance is given by [see (2.11), (2.21)]:

$$\mathbf{S}_{k+m+1} = \mathbf{H}_{k+m+1}\mathbf{P}_{k+m+1|k}\mathbf{H}_{k+m+1}^T + \mathbf{R}_{k+m+1} - \sum_{i=1}^{2m}(\mathbf{H}_{k+m+1}\mathbf{k}_i^*)(\mathbf{H}_{k+m+1}\mathbf{k}_i^*)^T. \tag{2.22}$$

As in standard EKF-SLAM, the term $(\mathbf{H}_{k+m+1}\mathbf{P}_{k+m+1|k}\mathbf{H}_{k+m+1}^T + \mathbf{R}_{k+m+1})$ is calculated in constant time. Since $\mathbf{H}_{k+m+1}$ contains only two non-zero blocks [see (2.15)], each term, $\mathbf{H}_{k+m+1}\mathbf{k}_i^*$, is calculated in constant time resulting in a total additional cost of $O(m)$ for calculating $2m$ such terms. Finally, since the dimensions of $\mathbf{H}_{k+m+1}\mathbf{k}_i^*$ are 2×1, each vector outer-product, $(\mathbf{H}_{k+m+1}\mathbf{k}_i^*)(\mathbf{H}_{k+m+1}\mathbf{k}_i^*)^T$, can be evaluated in constant time and the computational complexity for calculating the sum of $2m$ such matrices is $O(m)$. Thus the overall computational complexity for calculating the residual covariance matrix is also $O(m)$.

The Kalman gain is expressed as [see (2.12), (2.21)]:

$$\mathbf{K}_{k+m+1} = \mathbf{P}_{k+m+1|k}\mathbf{H}_{k+m+1}^T\mathbf{S}_{k+m+1}^{-1} - \sum_{i=1}^{2m}\mathbf{k}_i^*(\mathbf{H}_{k+m+1}\mathbf{k}_i^*)^T\mathbf{S}_{k+m+1}^{-1}. \tag{2.23}$$

Similar to the standard EKF-based SLAM, the first term, $\mathbf{P}_{k+m+1|k}\mathbf{H}_{k+m+1}^T\mathbf{S}_{k+m+1}^{-1}$, is calculated in $O(N)$ (note that since $\mathbf{S}$ is of dimensions 2×2, $\mathbf{S}^{-1}$ is calculated in constant time). Furthermore, since the terms $\mathbf{H}_{k+m+1}\mathbf{k}_i^*$ have already been calculated [see (2.22)], the cost of computing $\mathbf{k}_i^*(\mathbf{H}_{k+m+1}\mathbf{k}_i^*)^T\mathbf{S}_{k+m+1}^{-1}$ is $O(N)$. Thus the summation term in (2.23) is evaluated in $O(mN)$, leading to an overall computational complexity of $O(mN)$ for this step. Also, once the Kalman gain $\mathbf{K}_{k+m+1}$ is available, the state update [see (2.13)] is carried out with $O(N)$ computational cost.

Finally, the covariance update is expressed as [see (2.14), (2.21)]:

$$\begin{aligned}\mathbf{P}_{k+m+1|k+m+1} &= \mathbf{P}_{k+m+1|k} - \sum_{i=1}^{2m}\mathbf{k}_i^*\mathbf{k}_i^{*T} \\ &\quad - \mathbf{K}_{k+m+1}\mathbf{S}_{k+m+1}\mathbf{K}_{k+m+1}^T \\ &= \mathbf{P}_{k+m+1|k} - \sum_{i=1}^{2m}\mathbf{k}_i^*\mathbf{k}_i^{*T} - \sum_{i=1}^{2}\mathbf{k}_i{\mathbf{k}_i}^T,\end{aligned} \tag{2.24}$$

where $\mathbf{k}_i$ is the i^{th} column of $(\mathbf{K}_{k+m+1}\mathbf{S}_{k+m+1}^{1/2})$. Again, to simplify the notation, we denote $\mathbf{k}_{2m+1} = \mathbf{k}_1$, $\mathbf{k}_{2m+2} = \mathbf{k}_2$ and $\mathbf{k}_j = \mathbf{k}_j^*$, $j = 1, \ldots, 2m$, to obtain:

$$\mathbf{P}_{k+m+1|k+m+1} = \mathbf{P}_{k+m+1|k} - \sum_{i=1}^{2(m+1)} \mathbf{k}_i \mathbf{k}_i{}^T. \tag{2.25}$$

At this step, we do not actually evaluate the sum of the outer-product of the Kalman vectors and hence we only consider the computations required for generating the new Kalman vectors $\mathbf{k}_1$ and $\mathbf{k}_2$. Since the generation of new Kalman vectors only involves the Cholesky factorization of the 2×2 matrix $\mathbf{S}$ followed by the computation of $\mathbf{k}_i$, the covariance update step in GMP has $O(N)$ computational cost.

Landmark Initialization

Next we describe how landmark initialization can be efficiently carried out in the GMP framework. Every time a new landmark, $N+1$, is detected, an estimate for this landmark, $\hat{\mathbf{p}}_{N+1}$, has to be appended to the state vector. Also, the covariance matrix $\mathbf{P}_{k+m|k+m}$ [see (2.20)] needs to be appropriately augmented. While the new landmark's initial estimate can be generated as in the standard EKF-based SLAM, the following steps have to be carried out for updating the covariance:

1. Firstly, zeros are appended as the last two additional elements of each $\mathbf{k}_i$ vector.

2. Matrix $\mathbf{P}_{k+m|k}$ [see (2.7)] is augmented to include the block matrices that correspond to[7] :

   (a) $\mathbf{P}_{\mathbf{p}_{N+1}\mathbf{p}_{N+1}} = \mathbf{H}_{N+1}^T \left(\mathbf{H}_r \mathbf{P}_{\mathbf{x}_r\mathbf{x}_r} \mathbf{H}_r^T + \mathbf{R}\right) \mathbf{H}_{N+1}$, i.e., the new landmark's covariance.

   (b) $\mathbf{P}_{\mathbf{x}_r\mathbf{p}_{N+1}} = -\mathbf{P}_{\mathbf{x}_r\mathbf{x}_r} \mathbf{H}_r^T \mathbf{H}_{N+1}$, i.e., the new landmark's cross-correlation with the robot.

   (c) $\mathbf{P}_{\mathbf{p}_i\mathbf{p}_{N+1}} = -\mathbf{P}_{\mathbf{p}_i\mathbf{x}_r} \mathbf{H}_r^T \mathbf{H}_{N+1}$, $i = 1 \ldots N$, i.e., the new landmark's cross-correlation terms with each of the N existing landmarks.

   where $\mathbf{H}_r$ and $\mathbf{H}_{N+1}$ are the non-zero blocks of the measurement matrix [see (2.15)] corresponding to the observation of landmark $N+1$.

[7] Time indices have been omitted to simplify the discussion.

Although $N+2$ terms, as seen above, need to be determined to update the covariance matrix, the cost of calculating each of them is constant once $\mathbf{P}_{\mathbf{x}_r\mathbf{x}_r}$ and $\mathbf{P}_{\mathbf{p}_i\mathbf{x}_r}$ are retrieved. As shown below, we can obtain $\mathbf{P}_{\mathbf{x}_r\mathbf{x}_r}$ and $\mathbf{P}_{\mathbf{p}_i\mathbf{x}_r}$ [see (2.20)] at a cost of $O(m)$ each (note that $\mathbf{P}_{\mathbf{x}_r\mathbf{x}_r}$ and $\mathbf{P}_{\mathbf{p}_i\mathbf{x}_r}$ are also needed for data association and can be determined by the same process). We obtain the 3×3 sub-matrix $\mathbf{P}_{\mathbf{x}_r\mathbf{x}_r}$ as follows:

$$\mathbf{P}_{\mathbf{x}_r\mathbf{x}_{r\,k+m|k+m}} = \mathbf{P}_{\mathbf{x}_r\mathbf{x}_{r\,k+m|k}} - \sum_{i=1}^{2m} \mathbf{k}_{ri}\mathbf{k}_{ri}^T, \tag{2.26}$$

where $\mathbf{k}_{ri}$ denotes the first 3 elements of the vector $\mathbf{k}_i$ that correspond to the robot. Since each $\mathbf{k}_{ri}$ vector has dimensions 3×1, $\mathbf{P}_{\mathbf{x}_r\mathbf{x}_{r\,k+m|k+m}}$ can be evaluated in constant time, i.e., $O(m)$. Similarly, the 2×3 sub-matrix $\mathbf{P}_{\mathbf{p}_j\mathbf{x}_r}$ is evaluated in $O(m)$ as follows:

$$\mathbf{P}_{\mathbf{p}_j\mathbf{x}_{r\,k+m|k+m}} = \mathbf{P}_{\mathbf{p}_j\mathbf{x}_{r\,k+m|k}} - \sum_{i=1}^{2m} \mathbf{k}_{p_ji}\mathbf{k}_{ri}^T, \tag{2.27}$$

where $\mathbf{k}_{p_ji}$ denotes the 2 elements of the vector $\mathbf{k}_i$ that correspond to landmark $\mathbf{p}_j$. Subsequently, each new term of the covariance matrix, corresponding to landmark $N+1$, can be evaluated at a cost of $O(m)$. Since $N+2$ such terms need to be calculated, the overall cost for inserting a new landmark in the map is $O(mN)$.

From the preceding presentation, it is evident that by using the GMP technique, the computational complexity of standard EKF-based SLAM is $O(mN)$. Table 2.2 summarizes the computational complexity at each step of GMP. Here it is important

Table 2.2: Computational Complexity of the GMP EKF-based SLAM Algorithm. N: number of landmarks in the map, m: number of Kalman vectors in the vector outer-product sum.

Steps in GMP EKF-SLAM	Computational Complexity
State Propagation	$O(1)$
Covariance Propagation	$O(N)$
Residual Covariance	$O(m)$
Kalman Gain	$O(mN)$
State Update	$O(N)$
Covariance Update	$O(N)$
Landmark Initialization	$O(mN)$

to note that since *no approximation* has been made up to this point and as long as the number of delayed updates, m, is significantly smaller than N, the GMP SLAM will produce exactly the *same* estimates as the standard EKF-based SLAM, but in *linear* time. Inevitably, however, as the robot navigates and makes new observations, m will continuously increase. Therefore, in order to maintain the structure of the covariance matrix [see (2.20)] while allowing for linear-time updates, it is necessary to devise a technique whereby the number, m, of $\mathbf{k}_i$ vectors in the vector outer-product sum $\sum_{i=1}^{2m} \mathbf{k}_i \mathbf{k}_i^T$ [right-hand side of (2.20)] remains upper-bounded by a quantity $M_{max} \ll N$. In the following section, we describe how this is achieved by employing a low-rank approximation of $\sum_{i=1}^{2m} \mathbf{k}_i \mathbf{k}_i^T$, based on the Power method.

2.3.3 Low-Rank Approximation

Let $\mathbf{D} = \sum_{i=1}^{2m} \mathbf{k}_i \mathbf{k}_i^T$. Once the increasing number of Kalman vectors, m, in $\mathbf{D}$ reaches M_{max}, some of these vectors are discarded and $\mathbf{D}$ is approximated by $\mathbf{D}^*$ as the sum of the vector outer product of only M_{min} vectors, where $M_{min} < M_{max}$. The motivation for this low-rank approximation is to ensure that only m, where $M_{min} \leq m \leq M_{max} \ll N$, vectors will be involved in further computations [see (2.20)] and hence the computational cost of GMP will remain linear.

Since the objective of the Power-SLAM estimator is to minimize the information loss, i.e., *minimize* the trace of the covariance matrix $\mathbf{P}_{k+m|k+m}$ [see (2.20)], it is necessary to *maximize* the trace of the approximated Kalman vector outer-product sum $\mathbf{D}^*$. Given the above optimality criterion and the constraint that only M_{min} out of the M_{max} Kalman vectors in $\mathbf{D}$ can be retained, the *optimal* solution to this problem is obtained by determining the eigenvectors, $\mathbf{v}_i$, that correspond to the M_{min} largest eigenvalues of $\mathbf{D}$ and using them to construct $\mathbf{D}^*$. Specifically, consider the following eigen-decomposition of $\mathbf{D}$:

$$\mathbf{D} = \sum_{i=1}^{M_{max}} \mathbf{k}_i \mathbf{k}_i^T = \sum_{i=1}^{M_{max}} \lambda_i \mathbf{v}_i \mathbf{v}_i^T, \tag{2.28}$$

where $\lambda_1 > \lambda_2 \geq \lambda_3 ... \geq \lambda_{M_{max}}$ are the eigenvalues of $\mathbf{D}$ and $\mathbf{v}_i$, $i = 1, \ldots, M_{max}$, are the corresponding eigenvectors. The proposed low-rank approximation of $\mathbf{D}$ retains its

M_{min} largest eigenvalue-eigenvector pairs:

$$\mathbf{D} \simeq \sum_{i=1}^{M_{min}} \lambda_i \mathbf{v}_i \mathbf{v}_i^T = \sum_{i=1}^{M_{min}} \mathbf{k}_i^* \mathbf{k}_i^{*T} = \mathbf{D}^*, \tag{2.29}$$

where $M_{min} < M_{max}$ and, $\mathbf{k}_i^* = \sqrt{\lambda_i}\mathbf{v}_i$, are the new Kalman vectors. Note that this approximation is *optimal* since it retains the most informative vectors, i.e., the scaled eigenvectors that correspond to the largest eigenvalues of $\mathbf{D}$.

Furthermore, in order to maintain the linear-time processing cost of our proposed approach, it is necessary to use an algorithm that calculates these eigenvalue-eigenvectors pairs in linear time. We next show how this can be accomplished by employing the Power method [64] (Algorithm 1).

Algorithm 1 Power Method

Require: Matrix $\mathbf{D}$, scalars n_p, M_{min}

1: **for** $j = 1$ to M_{min} **do**
2: Generate random vector[8] $\mathbf{s}_0$
3: **for** $k = 0$ to $(n_p - 1)$ **do**
4: Compute $\mathbf{s}_{k+1} \leftarrow \mathbf{D}\mathbf{s}_k$
5: Find $\alpha \leftarrow ||\mathbf{s}_{k+1}||_\infty$
6: $\mathbf{s}_{k+1} \leftarrow \mathbf{s}_{k+1}/\alpha$
7: **end for**
8: $\lambda_j \leftarrow \alpha$ {λ_j is the dominant eigenvalue of $\mathbf{D}$}
9: $\mathbf{v}_j \leftarrow \mathbf{s}_{n_p}/||\mathbf{s}_{n_p}||$ {$\mathbf{v}_j$ is the eigenvector of $\mathbf{D}$, corresponding to λ_j}
10: $\mathbf{D} \leftarrow \mathbf{D} - \lambda_j \mathbf{v}_j \mathbf{v}_j^T$
11: **end for**
12: **return** $\lambda_j, \mathbf{v}_j, j = 1, \ldots, M_{min}$

In order to evaluate the computational complexity of the Power method, when applied to this problem, consider the first iteration when $j = 1$ and $k = 0$. Given $\mathbf{s}_0$, Step 4 of Algorithm 1 calculates:

$$\mathbf{s}_1 = \mathbf{D}\mathbf{s}_0 = \left(\sum_{i=1}^{M_{max}} \mathbf{k}_i {\mathbf{k}_i}^T \right) \mathbf{s}_0 = \sum_{i=1}^{M_{max}} \mathbf{k}_i (\mathbf{k}_i^T \mathbf{s}_0) \tag{2.30}$$

and the computational cost for this step is $O(M_{max}N)$. Next, the costs for obtaining the ∞-norm in Step 5 and dividing $\mathbf{s}_{k+1}$ by α in Step 6 are $O(N)$ each. Thus the total

[8] In this particular problem, the convergence speed of the Power Method increases significantly by selecting $\mathbf{s}_0 = \mathbf{k}_j$ where $||\mathbf{k}_j|| > ||\mathbf{k}_i||$, $\forall i \in \{1 \ldots M_{max}\} \backslash \{j\}$.

cost for Steps 4-6 remains $O(M_{max}N)$. Since Steps 4-6 are repeated n_p times[9] , the total cost to acquire α (dominant eigenvalue λ_1) and $\mathbf{s}_{n_p}$ (dominant eigenvector $\mathbf{v}_1$), becomes $O(n_p M_{max}N)$. Once the dominant eigenvalue/eigenvector is obtained, $\mathbf{D}$ is modified in Step 10 to include the additional vector outer-product, $\lambda_j \mathbf{v}_j \mathbf{v}_j^T$. As a result, $\mathbf{D}$ will now contain $M_{max}+1$ vector outer-products.

Similarly, by repeating the above process, the second largest eigenvalue/eigenvector pair can be acquired at a cost of $O(n_p(M_{max}+1)N)$ and the new $\mathbf{D}$ will contain $M_{max}+2$ vector outer-products. Thus, we can see that the cost of obtaining the i^{th} largest eigenvalue/eigenvector pair, where $i = 1, \ldots, M_{min}$, is $O(n_p(M_{max}+(i-1))N)$. The total cost for obtaining all M_{min} such pairs becomes:

$$\begin{aligned} & O(n_p(M_{min}M_{max} + (1 + \ldots + (M_{min}-1)))N) \\ & = O(n_p M_{min}(M_{max} + \frac{M_{min}-1}{2})N) \\ & \approx O(n_p M_{min} M_{max} N) \end{aligned}$$

since $M_{min} \ll M_{max}$. Hence, as long as $n_p M_{min} M_{max} \ll N$, the computational cost of the Power method remains linear in N.

Remark 1 (Speeding up the Power Method)**.** *A simplistic and very fast solution to the low-rank approximation described in* (2.29) *would be to select and retain the largest M_{min} out of the M_{max} available $\mathbf{k}_i$ vectors based on their 2-norm, $||\mathbf{k}_i||$. Although this is often a reasonable approximation and, as explained later, guarantees that the resulting estimator remains conservative, it is not optimal unless:*

1. *$\mathbf{k}_i = \sqrt{\lambda_i}\mathbf{v}_i$ for $i = 1 \ldots M_{min}$, where $\mathbf{v}_i$ are the eigenvectors corresponding to the M_{min} largest eigenvalues of $\mathbf{D}$, or*

2. *$||\mathbf{k}_i|| \simeq 0$, for $i = (M_{min}+1) \ldots M_{max}$.*

While condition (1) is rarely satisfied in practice, condition (2), from extensive simulation studies, is seen to be usually true for $i = (M_{mid}+1) \ldots M_{max}$, where $M_{mid} \gg M_{min}$.

In order to expand the time horizon over which this low-rank approximation is delayed (i.e., intuitively large values of M_{max} allow us to retain the most informative $\mathbf{k}_i$

[9] In most cases, n_p = 7-10 steps are necessary for this iterative process to converge. Based on two successive estimates for the eigenvector, convergence is detected when $|1 - \mathbf{s}_{n_p}^T \mathbf{s}_{n_p-1}/(||\mathbf{s}_{n_p}|| \times ||\mathbf{s}_{n_p-1}||)| < 10^{-6}$, i.e, the angle between these two vectors is smaller than $\sim 10^{-6}$ rad.

vectors), a further approximation can be employed based on the condition (2) mentioned earlier. Based on this observation, the matrix $\mathbf{D}$ *is first approximated as:*

$$\mathbf{D} = \sum_{i=1}^{M_{max}} \mathbf{k}_i \mathbf{k}_i^T \simeq \sum_{i=1}^{M_{mid}} \mathbf{k}_i \mathbf{k}_i^T = \tilde{\mathbf{D}} \tag{2.31}$$

and Algorithm 1 is applied to $\tilde{\mathbf{D}}$ *instead of* $\mathbf{D}$ *to determine its* M_{min} *largest eigenvectors and eigenvalues, i.e.,*

$$\tilde{\mathbf{D}} = \sum_{i=1}^{M_{mid}} \tilde{\lambda}_i \tilde{\mathbf{v}}_i \tilde{\mathbf{v}}_i^T \simeq \sum_{i=1}^{M_{min}} \tilde{\lambda}_i \tilde{\mathbf{v}}_i \tilde{\mathbf{v}}_i^T = \sum_{i=1}^{M_{min}} \tilde{\mathbf{k}}_i^* \tilde{\mathbf{k}}_i^{*T} = \tilde{\mathbf{D}}^*, \tag{2.32}$$

where $\tilde{\lambda}_i$, $\tilde{\mathbf{v}}_i$, *and* $\tilde{\mathbf{k}}_i^*$ *are defined as in* (2.29).

Selecting the M_{mid} *largest (in the* 2*-norm sense)* $\mathbf{k}_i$ *vectors incurs a cost of* $O(M_{max}N)$ *for determining their magnitude and a cost of* $O(M_{max}\log(M_{max}))$ *for sorting the vectors in descending order based on their* 2*-norms. After this process is complete, the cost of the Power Method reduces to* $O(n_p M_{min} M_{mid} N)$. *Typical values of these parameters used in our tests are: (i)* $M_{max} = (2-to-10)\%$ *of* N, *(ii)* $M_{mid} = \max\{2, (5-to-10)\%$ *of* $M_{max}\}$ *(i.e., the Power Method is not used when* $M_{mid} = 2$ *which corresponds to* $N < 250$, *i.e,* ~ 100 *- landmark maps), (iii)* $M_{min} = 1-2$, *and (iv)* $n_p = 7-10$ *(i.e.,* $n_p M_{min} M_{mid}$ *is 1-2 orders of magnitude smaller than* N*). Note that these are only representative values and they can be adjusted on-line to meet the availability of computational resources. Subject to the linear computational complexity constraint, choosing a higher value of* M_{max} *will allow the algorithm to extend the time horizon over which the approximations are carried out, hence allowing the algorithm to capture more information. On the other hand when selecting* M_{min}, *it has been seen (from simulation studies) that most of the information is captured in the first* $1-2$ *eigenvalue/eigenvector pairs. Therefore, choosing a higher value of* M_{min} *will not lead to a substantial increase in the information gain. Finally, the value of* M_{mid} *depends on the particular realization of SLAM and should be determined after studying the problem at hand.*

Lastly, we would like to note that the above approximation is well justified for the following two reasons:

1. Most of the elements of the vectors $\mathbf{k}_i$ have very small values, except (i) the

elements that correspond to the robot, and (ii) elements corresponding to landmarks observed over the last $(M_{max} - M_{min})$ time steps and landmarks strongly correlated with them.

2. The rank-2 covariance update process (described in Section 2.3.4), sparsifies the $\mathbf{k}_i$'s by replacing the largest elements, in the absolute value sense, of the $\mathbf{k}_i$'s with zeros. Hence, only few directions, $\mathbf{v}_i$, of $\mathbf{D}$ contain substantial information (typically $M_{min} = 1$ or 2). The remaining ones can be discarded without significant loss of accuracy.

At this point, we should note that the SWKF approach in [63] is also based on the first observation mentioned above. In that case, however, all elements of $\mathbf{k}_i$, except those corresponding to the robot and the observed landmark, are discarded at *every* time step. Since there exist strong correlations between neighborhoods of landmarks in dense maps, this crude approximation generates very conservative updates in the SWKF. Furthermore, and in stark contrast to the one-step approximations involved in [63] and [60], by employing the GMP framework, we delay the time when an approximation becomes necessary. This, in effect, allows us to retain the most *informative* among all the $\mathbf{k}_i$ vectors accumulated over an extended period of time, thus significantly reducing the information loss.

Remark 2 (Conservative Estimator)**.** *A key advantage of the presented low-rank approximation is that the covariance matrix remains conservative. Since* $\mathbf{D} \succeq \mathbf{D}^*$ (2.20) *yields:*

$$\mathbf{P}_{k+m|k+m} = \mathbf{P}_{k+m|k} - \mathbf{D} \;\succeq\; \mathbf{P}_{k+m|k} - \mathbf{D}^* = \mathbf{P}^*_{k+m|k+m} \tag{2.33}$$

Here $\mathbf{P}^*_{k+m|k+m} = \mathbf{P}_{k+m|k} - \sum_{i=1}^{M_{min}} \mathbf{k}^*_i \mathbf{k}^{*T}_i$ *is the new approximated covariance. The estimator also remains conservative for* $\tilde{\mathbf{D}}^*$*, since [see* (2.31), (2.32)*]* $\mathbf{D} \succeq \tilde{\mathbf{D}} \succeq \tilde{\mathbf{D}}^*$.

Remark 3 (Quantifying the Information Loss)**.** *Importantly, the Power-SLAM approach provides a concrete measure of the information loss incurred due to the low-rank approximation. Quantifying the approximation involved is necessary in order to adjust the parameters* M_{min} *and* M_{max} *on-line, so as to meet performance requirements. This can be achieved by computing the ratio* $(\mathrm{tr}(\mathbf{D}) - \mathrm{tr}(\mathbf{D}^*))/\mathrm{tr}(\mathbf{D})$ *with complexity* $O((M_{max} + M_{min})N)$*, where* $\mathrm{tr}(\mathbf{D}) = \sum_{i=1}^{M_{max}} \mathbf{k}_i{}^T \mathbf{k}_i$ *and* $\mathrm{tr}(\mathbf{D}^*) = \sum_{i=1}^{M_{min}} \mathbf{k}^{*T}_i \mathbf{k}^*_i$.

2.3.4 Linear-Time, Rank-2 Covariance Updates

The main drawback of any low-rank approximation of $\mathbf{D}$ [see (2.29)] is that it does not guarantee loss of rank of the covariance matrix, $\mathbf{P}_{k+m|k+m}$ [see (2.33)], after "infinite" time, as is expected when the system reaches steady-state [65, 23]. This is due to the fact that, in general, the rank of matrix $\mathbf{P}_{k+m|k}$ is $(2N+3)$, i.e., it is full-rank, while the rank of $\mathbf{D}$ is at most M_{max}. Hence the rank of $\mathbf{P}_{k+m|k+m} = \mathbf{P}_{k+m|k} - \mathbf{D}$ will be at least $(2N+3-M_{max})$ with $M_{max} \ll N$.

Furthermore, due to the propagation steps [see (2.21)], the covariance $\mathbf{P}_{k+m|k}$, in general, will increase continuously. Moreover, $\mathbf{D}$ will also become larger when the same landmarks are re-observed in a given period of time. This is due to the fact that re-observing landmarks provides additional information, resulting in larger Kalman gain. But since $\mathbf{D}$ is never explicitly computed, this information is never incorporated in $\mathbf{P}_{k+m|k}$ to reduce the covariance.

In order to ensure that the trace of $\mathbf{P}_{k+m|k}$ decreases monotonically, certain elements of $\mathbf{D}$ (as permitted by the linear computational complexity constraint) have to be subtracted from $\mathbf{P}_{k+m|k}$ at every time step. Note, however, that any modification of $\mathbf{D}$ requires that the positive semi-definite property of $\mathbf{D}$ be maintained, else the low-rank approximation described in the previous section will not guarantee consistency. To achieve this, we propose the following rank-2 covariance updates:

1. From $\mathbf{D}$, choose the vector $\mathbf{k}_j$ with the maximum element, $k_{\xi j}$, in the absolute value sense, amongst all the $\mathbf{k}_i$'s. Let this be the ξ^{th} element of $\mathbf{k}_j$.

2. Express $\mathbf{k}_j$ as follows:

$$\mathbf{k}_j = \mathbf{k}_j^+ + \mathbf{k}_j^-. \tag{2.34}$$

   Here, $\mathbf{k}_j^+$ is identical to $\mathbf{k}_j$, except with its ξ^{th} element set to zero. Therefore, $\mathbf{k}_j^-$ is a vector of all zeros except the ξ^{th} element, which is set to $k_{\xi j}$. Thus,

$$\mathbf{k}_j\mathbf{k}_j^T = (\mathbf{k}_j^+ + \mathbf{k}_j^-)(\mathbf{k}_j^+ + \mathbf{k}_j^-)^T \tag{2.35}$$

$$= \mathbf{k}_j^+\mathbf{k}_j^{+T} + \mathbf{k}_j^+\mathbf{k}_j^{-T} + \mathbf{k}_j^-\mathbf{k}_j^{+T} + \mathbf{k}_j^-\mathbf{k}_j^{-T} \tag{2.36}$$

   Now the rank-2 update step comprises of subtracting the matrix $\delta\mathbf{P}_j = \mathbf{k}_j^+\mathbf{k}_j^{-T} +$

$\mathbf{k}_j^- \mathbf{k}_j^{+T} + \mathbf{k}_j^- \mathbf{k}_j^{-T}$ from $\mathbf{P}_{k+m|k}$ while $\mathbf{k}_j^+ \mathbf{k}_j^{+T}$ is included back in the vector outer-product sum as follows:

$$\mathbf{P}_{k+m|k+m} = \left(\mathbf{P}_{k+m|k} - \delta\mathbf{P}_j\right) - \sum_{i=1}^{2m} \mathbf{k}_i^+ \mathbf{k}_i^{+T} \tag{2.37}$$

where $\mathbf{k}_i^+ = \mathbf{k}_i$, $\forall i \neq j$. Since $\delta\mathbf{P}_j$ has only $O(N)$ non-zero elements, $(\mathbf{P}_{k+m|k} - \delta\mathbf{P}_j)$ can be calculated in linear time. Furthermore as required, after the rank-2 covariance update, the new matrix $\mathbf{D}^* = \sum_{i=1}^{2m} \mathbf{k}_i^+ \mathbf{k}_i^{+T}$ remains positive semi-definite, since it is still expressed as the sum of vector outer-products.

We now prove that the proposed rank-2 update is optimal (i.e., it minimizes the trace of the updated covariance matrix) under the *linear computational complexity requirement.*

Let us begin by assuming that $\mathbf{k}_j^+ = (\mathbf{I} - \mathbf{A}_j)\mathbf{k}_j$, where $\mathbf{A}_j$ is a selector matrix, $\mathbf{k}_j$ is the vector used in the update (to be determined), and $\mathbf{P}_{k+m|k+1} = \mathbf{P}_{k+m|k} - \delta\mathbf{P}_j$ denotes the updated covariance matrix after incorporating a single rank-2 covariance update. Therefore, [see (2.34)] $\mathbf{k}_j^- = \mathbf{A}_j\mathbf{k}_j$ and

$$\delta\mathbf{P}_j = (\mathbf{A}_j\mathbf{k}_j)(\mathbf{A}_j\mathbf{k}_j)^T + \mathbf{k}_j^+(\mathbf{A}_j\mathbf{k}_j)^T + (\mathbf{A}_j\mathbf{k}_j)\mathbf{k}_j^{+T}. \tag{2.38}$$

In order to carry out this rank-2 covariance update, $\mathbf{k}_j$ and $\mathbf{A}_j$ need to be determined such that the following two conditions are satisfied:

(C1) The cost of computing $\delta\mathbf{P}_j$ [see (2.38)] is minimized, allowing at most $O(N)$ operations to maintain the linear computational complexity of the algorithm.

(C2) The trace of $\mathbf{P}_{k+m|k+1} = \mathbf{P}_{k+m|k} - \delta\mathbf{P}_j$ is minimized. Note that the minimization of $\mathrm{tr}(\mathbf{P}_{k+m|k+1})$ ensures minimization of $\mathrm{tr}(\mathbf{P}_{k+m|k+m})$ when the vectors $\mathbf{k}_i$ are discarded during the low-rank approximation.

Since the vector $\mathbf{k}_j$ can, in general be dense, while computing $\mathbf{A}_j\mathbf{k}_j$, (C1) requires that the matrix $\mathbf{A}_j$ has at most $n \ll N^2$ non-zero elements[10] . If these non-zero elements are distributed among $1 \leq p \leq N$ rows of $\mathbf{A}_j$, then the cost for computing $\mathbf{A}_j\mathbf{k}_j$ is $O(n)$ and the resulting vector will have p non-zero elements. Since $(\mathbf{A}_j\mathbf{k}_j)(\mathbf{A}_j\mathbf{k}_j)^T$ is a symmetric matrix, the cost for computing it will be $\frac{p(p+1)}{2}$.

[10] For clarity in the following derivations, we set the state vector size to N.

When computing $\mathbf{k}_j^+ = (\mathbf{I} - \mathbf{A}_j)\mathbf{k}_j = \mathbf{k}_j - \mathbf{A}_j\mathbf{k}_j$, $d \in \{0, 1, \ldots, p\}$ subtractions are necessary, depending on the number of elements of $\mathbf{k}_j^+$ that can be directly set to zero (i.e., by appropriately selecting the elements of $p-d$ rows of $\mathbf{A}_j$, it can be ensured that elements of $\mathbf{A}_j\mathbf{k}_j$ in these rows are the same as those of $\mathbf{k}_j$). Thus, if $\mathbf{k}_j^+$ contains $p-d$ zeros, computing $\mathbf{k}_j^+(\mathbf{A}_j\mathbf{k}_j)^T$ requires $(N-(p-d))p$ operations. Hence the total cost for calculating $\delta\mathbf{P}_j$ can be expressed as a function of p, d, n, and N as follows:

$$c(p, d, n, N) = \frac{1}{2}\left(-p^2 + (2N + 2d + 1)p + 2(n + d)\right). \tag{2.39}$$

Note that (2.39) is a concave function of p with the maximum achieved at $p = \frac{2N+2d+1}{2} > N$. Thus, it is a monotonically increasing function within the interval of interest, i.e., $[1 \ldots N]$, with the minimum occurring at $p = 1$ (i.e., since $\mathbf{A}_j$ cannot be a matrix of all zeros, at least one row of $\mathbf{A}_j$ will have non-zero elements). Substituting $p = 1$ in (2.39) the total cost becomes:

$$c(1, d, n, N) = N + n + 2d. \tag{2.40}$$

Now since $p = 1$, the number of subtractions, d, can either be 0 or 1. Also, the structure of matrix $\mathbf{A}_j$, that contains only one non-zero row (e.g., the ξ^{th} row), is given by:

$$\mathbf{A}_j^T = [\mathbf{0} \ldots \mathbf{a}_\xi \ldots \mathbf{0}], \tag{2.41}$$

where $\mathbf{a}_\xi$ denotes the ξ^{th} row of $\mathbf{A}_j$, with $n \leq N$ non-zero elements.

We now turn our attention to (C2). Minimizing the trace of $\mathbf{P}_{k+m|k+1}$ is equivalent to maximizing the trace of $\delta\mathbf{P}_j$. Substituting (2.41) in (2.38), we obtain:

$$\begin{aligned} \mathrm{tr}(\delta\mathbf{P}_j) &= \mathbf{k}_j^T\left(\mathbf{A}_j + \mathbf{A}_j^T - \mathbf{A}_j^T\mathbf{A}_j\right)\mathbf{k}_j \\ &= -(\mathbf{a}_\xi^T\mathbf{k}_j)^2 + 2k_{\xi j}(\mathbf{a}_\xi^T\mathbf{k}_j) \end{aligned} \tag{2.42}$$

where $k_{\xi j}$ is the ξ^{th} scalar element of vector $\mathbf{k}_j$ and (2.42) is a concave function of $\mathbf{a}_\xi$. Computing its derivative with respect to the elements of $\mathbf{a}_\xi$, the maximum of $\mathrm{tr}(\delta\mathbf{P}_j)$ is reached when:

$$(\mathbf{a}_\xi^T\mathbf{k}_j)\mathbf{k}_j = k_{\xi j}\mathbf{k}_j. \tag{2.43}$$

This is trivially achieved by setting $\mathbf{a}_\xi = \mathbf{e}_\xi$, where $\mathbf{e}_\xi$ is the ξ^{th} canonical unit vector. Therefore, $\mathbf{A}_j$ [see (2.41)] becomes a matrix of zeros, except the ξ^{th} diagonal element

which is equal to one. As a result of this, the vector $\mathbf{k}_j^- = \mathbf{A}_j\mathbf{k}_j$ has only one non-zero element, i.e., $k_{\xi j}$, in the ξ^{th} location; the rest of its elements are zero. Also, the vector $\mathbf{k}_j^+ = (\mathbf{I} - \mathbf{A}_j)\mathbf{k}_j$ has the same elements as $\mathbf{k}_j$, except the ξ^{th} element, which is zero.

Finally from (2.38), we can see that $\delta\mathbf{P}_j$ will have non-zero elements only in its ξ^{th} row and column. Hence, the total cost for computing $\delta\mathbf{P}_j$ becomes $c(1, 0, 0, N) = N$. Subtracting $\delta\mathbf{P}_j$ from $\mathbf{P}_{k+m|k+m}$ will also have cost N. Moreover, due to this special structure of the resulting $\delta\mathbf{P}_j$ matrix (i.e., non-zero elements only in its ξ^{th} row and column), the rank of this matrix is 2 (hence the name rank-2 updates).

What remains to be determined are the indices j and ξ that satisfy (C2). Substituting $\mathbf{a}_\xi^T\mathbf{k}_j = k_{\xi j}$ in (2.42), we have $\max(\mathrm{tr}(\delta\mathbf{P}_j)) = k_{\xi j}^2$. Hence maximizing the $\mathrm{tr}(\delta\mathbf{P}_j)$ is guaranteed by selecting among the $\mathbf{k}_j$ vectors, the one which has the maximum element $k_{\xi j}$, in the absolute value sense. This maximum element, among $2m$ Kalman vectors (each of dimension $N \times 1$), can be determined at a cost of $O(mN)$.

Thus, we demonstrated that the overall computational complexity of a single rank-2 covariance update is $O(mN)$. Furthermore, this rank-2 covariance update process can be repeated multiple times during each time step, depending on the availability of computational resources, to further decrease the trace of $\mathbf{P}_{k+m|k+1}$ and speed up convergence.

Table 2.3: Computational Complexity of Power-SLAM. N: number of landmarks in the map, m: number of Kalman Vectors in the vector outer-product sum, $M_{min} \leq m \leq M_{max} \ll N$, n_p: number of iterations of the Power Method.

Steps in Power-SLAM	Computational Complexity
Global Map Postponement	$O(mN)$
Low Rank Approximation (Power Method)	$O(n_p M_{min} M_{max} N)$
Rank-2 updates	$O(mN)$

Before presenting the simulation and experimental results, we summarize the three key algorithmic components of our proposed approach along with their computational complexity (see Table 2.3). Firstly, we showed that by using the GMP technique, approximations necessary for ensuring linear computational complexity of EKF-based SLAM can be delayed over multiple time steps. Secondly, we presented a linear-cost low-rank approximation technique that retains the most informative Kalman vectors

from the postponement phase using the Power Method. Lastly, in order to speed up the convergence of our proposed estimator, linear-complexity rank-2 covariance updates were introduced. Depending on the availability of computational resources at each time step, multiple rank-2 updates can be carried out to further speed up convergence.

2.4 Simulations

2.4.1 Simulation Setup

The simulations used to validate the performance of the Power-SLAM algorithm have been implemented in MATLAB. The robot starts at a known position and follows an 8-shaped trajectory shown in Fig. 4.7, where the radius of each circle is 150 m. The maximum sensing range of the robot is set to 8 m and it has a 360 degrees field of view for range and bearing measurements. The noise in the measurements is modeled as zero-mean, white Gaussian. Every 0.2 seconds, the robot receives the following measurements: (i) odometry (linear, v, and rotational, ω, velocity) with noise standard deviation $\sigma_v = 3\%v$, and $\sigma_\omega = 3\%\omega$, (ii) range d, with $\sigma_d = 8$ cm, and (iii) bearing θ, with $\sigma_\theta = 1$ degree.

In this simulation, the robot observes approximately 500 landmarks (i.e., the size of the state vector increases from 3 to 1000) over 2000 time steps with an average of 1.6 landmark observations per time step. The robot closes loops approximately every 310 time steps. The maximum number of Kalman vectors, M_{max}, and the number of rank-2 updates at each time step, are both set to 10% of the size of the state vector at that time step. The number of Kalman vectors, $\mathbf{k}_i$, considered for the low-rank approximation, are set to $M_{mid} = \max(2, 0.05M_{max})$. The Power Method extracts the dominant eigenvalue and eigenvector ($M_{min} = 1$).

2.4.2 Simulation Results

The objective of our simulation studies is to demonstrate the accuracy of the Power-SLAM algorithm, verify its consistency, and compare its performance to that of (i) EKF-based SLAM, (ii) SWKF SLAM [63], and (iii) CKF SLAM [60]. We use the First Estimates Jacobian (FEJ) variants of the EKF and the Power-SLAM algorithm in our

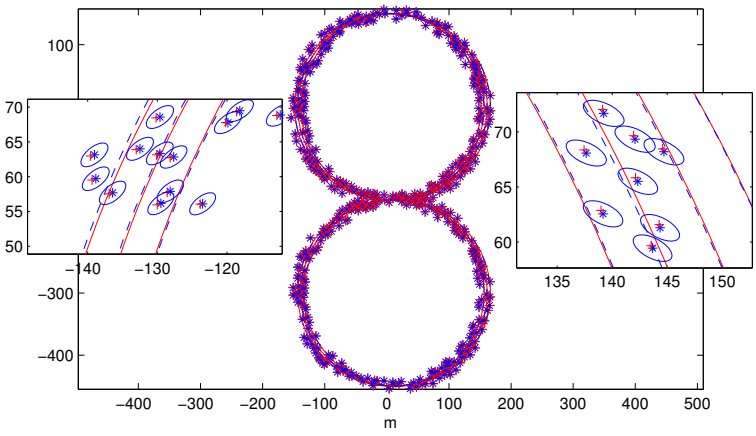


Figure 2.1: True robot trajectory (solid red line), true landmark positions (+), Power-SLAM estimated robot trajectory (dashed blue line), Power-SLAM estimated landmark positions (*), and their 3σ uncertainty ellipses. Insets are zoomed sections for better viewing of the uncertainty ellipses.

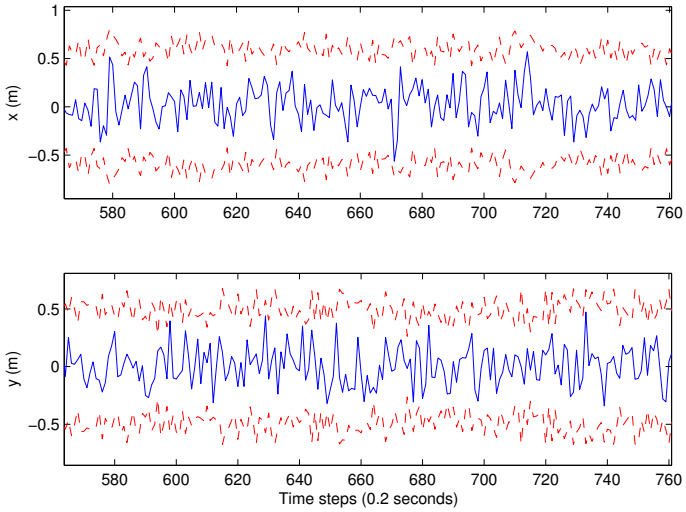


Figure 2.2: Measurement residuals (solid) and corresponding 3σ bounds (dashed).

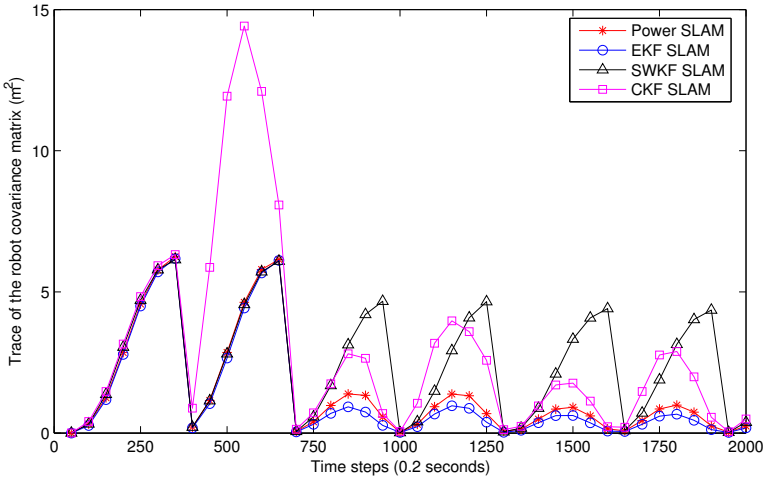


Figure 2.3: Trace of the robot's covariance matrix.

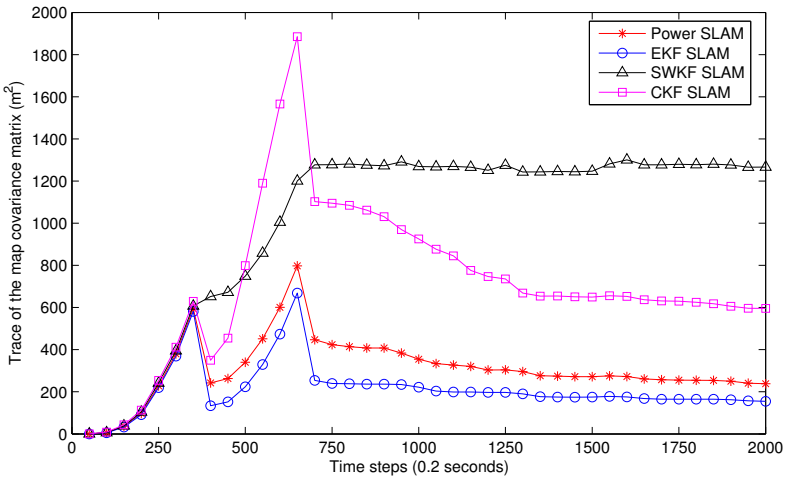


Figure 2.4: Trace of the map's covariance matrix.

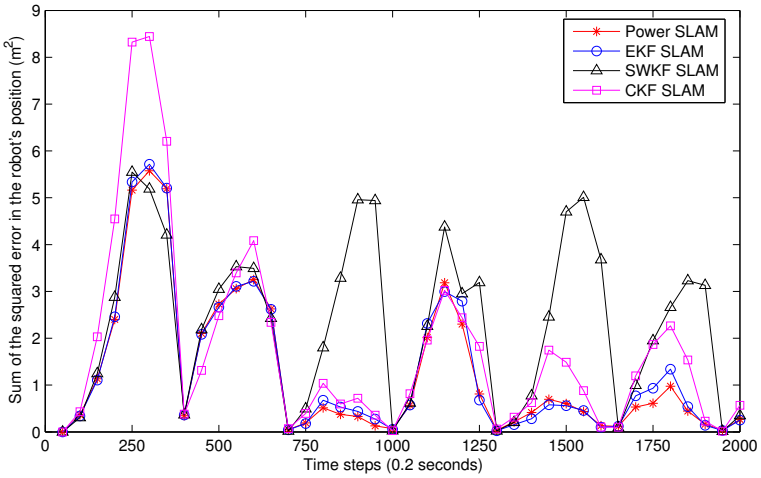


Figure 2.5: Sum of the squared error in the robot's position estimates.

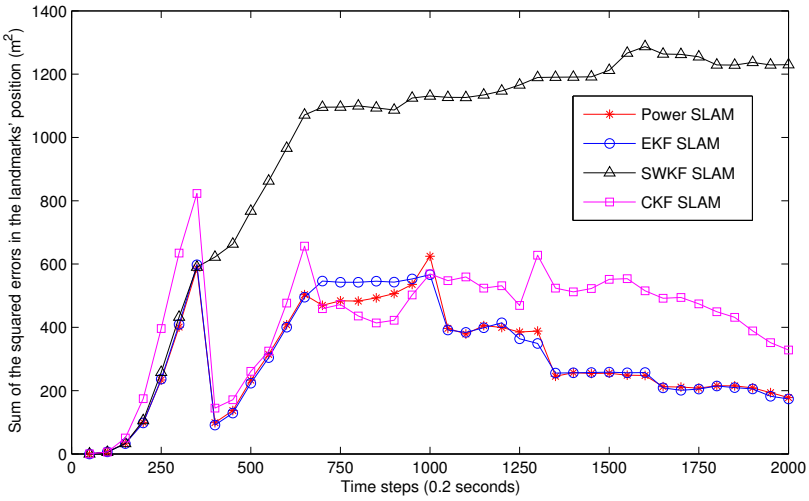


Figure 2.6: Sum of the squared error in the landmarks' position estimates.

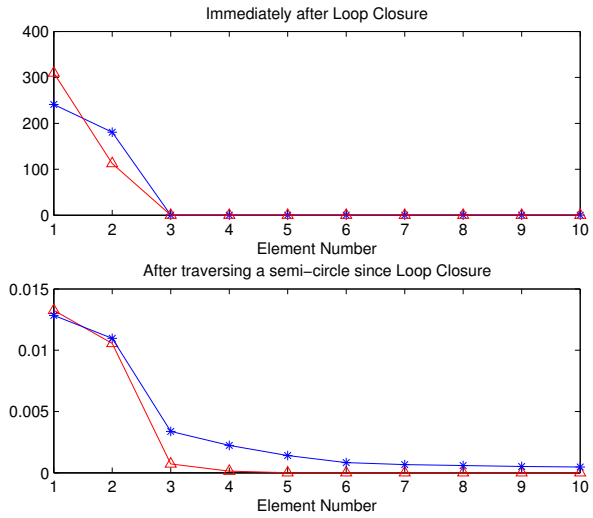


Figure 2.7: The 10 largest squared 2-norms (△) of the Kalman vectors, and the 10 largest eigenvalues (*) of the Kalman vector outer-product sum at 2 time instances: immediately after loop closure (top figure) and traversing a semi-circle after loop closure (bottom figure).

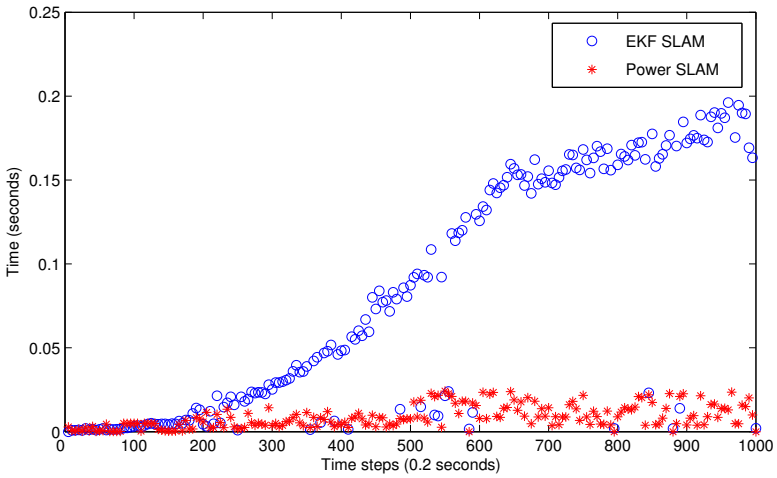


Figure 2.8: Comparison of the execution times for EKF-SLAM and Power-SLAM.

simulation studies. The FEJ estimator, proposed in [66], provides a solution to the inconsistency problem based on the observability analysis of the linearized system model in EKF-SLAM.

Note that the standard EKF-SLAM has computational complexity $O(N^2)$, while all other algorithms evaluated hereafter have processing requirements linear, $O(N)$, in the number of features. However, there are certain differences in the actual processing cost of each of the linear estimators. Although the SWKF estimator has fixed processing requirements, the CKF computational cost can be adjusted. To ensure a fair comparison, the CKF covariance updates are set so as to have the same cost as the rank-2 updates of the Power-SLAM algorithm.

We start with a qualitative evaluation of the Power-SLAM algorithm. As shown in Fig. 2.1, the Power-SLAM estimates for both the robot trajectory and the landmark positions are very close to the real ones. Also note that the 3σ ellipses of uncertainty for the estimated landmark positions contain the true positions, indicating consistency. Fig. 2.2 depicts the measurement residuals along with their corresponding 3σ bounds for the Power-SLAM method (only 200 time steps are shown to ensure clarity). This figure verifies that the Power-SLAM estimator is consistent.

We now turn our attention to the quantitative results presented in Figs. 2.3-2.6. Although all 3 linear-complexity estimators are conservative as compared to the standard EKF-SLAM, the SWKF is the most conservative one, followed by the CKF. The Power-SLAM estimator is the least conservative, which is evident when comparing the trace of the robot-position covariance matrix to the corresponding one for the EKF-SLAM (see Fig. 2.3). The same conclusion can be reached by comparing the traces of the landmarks' covariance matrices for each of these estimators (see Fig. 2.4). For the case of the landmarks, in particular, the SWKF covariance does not decrease with time as the robot revisits the same areas. While this is not true for the CKF, the rate of decrease of the covariance matrix trace is very slow when compared to that of the Power-SLAM estimator. This behavior is due to the fact that both the SWKF and the CKF are based on crude approximations that take place during each time step and result in large information loss. In contrast, the Power-SLAM algorithm is able to minimize the information loss by (i) delaying approximations over large time horizons, and (ii) extracting and retaining the most informative Kalman vectors during each approximation.

The level of "conservatism" of each algorithm, when compared to the EKF-SLAM estimator, also affects the accuracy of the estimates. Specifically, both the robot's and landmarks' position errors for the SWKF and CKF are significantly larger when compared to the ones for the Power-SLAM algorithm (Figs. 2.5 and 2.6), which achieves accuracy almost indistinguishable to that of EKF-SLAM.

The average squared error in the position estimates for each landmark, when compared to the standard EKF, is 75% higher for the CKF and 490% for the SWKF, whereas it is only 14% higher for Power-SLAM. Similarly, for the robot position estimates, the average squared error, when compared to the standard EKF, is 18% higher for the CKF and 96% for the SWKF while it is only 3% higher for Power-SLAM. This is due to the fact that the Power-SLAM algorithm is based on optimal approximations within the linear-complexity processing constraints.

Fig. 2.7 shows the 10 largest eigenvalues and 10 largest values (in the squared 2-norm sense) of the 100 Kalman vectors in $\mathbf{D}$ at two time instances: (i) just after loop closure, and (ii) when the robot has traveled a semi-circle after loop closure. As expected, the Kalman vectors carry substantially more information after loop closure than at other time steps. Moreover, in both cases, 2% to 10% of the Kalman vectors carry the bulk of the information and hence the others can be discarded in order to speed up the Power Method as discussed in Remark 1.

Finally, Fig. 2.8 shows the execution times per time step for EKF-SLAM and Power-SLAM for the first 1000 time steps of the simulation. The results presented in this figure corroborate our analytical studies. In particular, the execution time for Power-SLAM is an order of magnitude lower than that of EKF-SLAM. Thus, Power-SLAM can be successfully used to improve the real-time performance of SLAM without significant loss of accuracy.

2.5 Experiments

2.5.1 Experimental Setup

An iRobot Packbot robot, equipped with a Pointgrey Firefly stereo rig and a PC104 computer was used for the experiments (see Fig. 2.9). The stereo rig has been calibrated using the calibration technique by Zhang [67] and Heikkila *et al.* [68] to obtain its

intrinsic and extrinsic parameters.

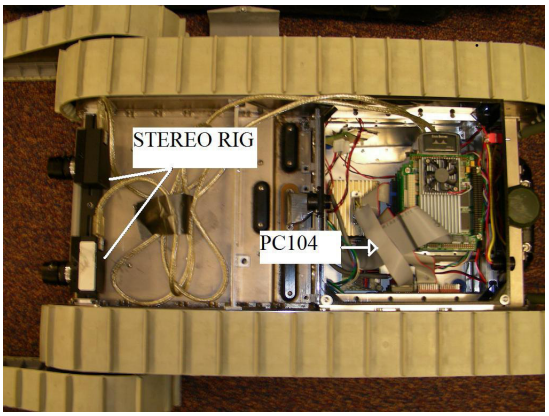


Figure 2.9: The Packbot robot equipped with a Pointgrey Firefly stereo rig.

During the experiments, the Packbot explored an indoor office environment and captured stereo images of its surroundings while moving in a plane. The robot received proprioceptive measurements, i.e., linear, v_m, and angular, ω_m, velocity, at 10 Hz and exteroceptive measurements, i.e., the stereo images, at approximately $0.5 - 1$ Hz. The noise in the proprioceptive measurements is assumed to be zero-mean, white Gaussian with standard deviation $\sigma_v = 3\%\ max(v_m)$ and $\sigma_\omega = 3\%\ max(\omega_m)$. The image resolution is 640×480 pixels and an additive white Gaussian noise of 2 pixels is assumed for the camera measurements.

For the duration of the experiment (approximately 2.5 mins), a total of 103 images were captured by each camera. SIFT keypoints [16], matched in the corresponding stereo images, were used to determine the 3D position of the point features based on stereo triangulation. Examples of detected and matched keypoints are shown in Figs. 2.10 and 2.11, respectively. On average, 1247 keypoints/image were detected, while 14.36 keypoints were matched for each set of stereo images. A total of 1088 point features were added to the state vector.

2.5.2 Experimental Results

Fig. 2.12 compares the trace of the robot's position covariance matrix for the SWKF, CKF, EKF and Power-SLAM estimators. Note that the trace of the robot's covariance

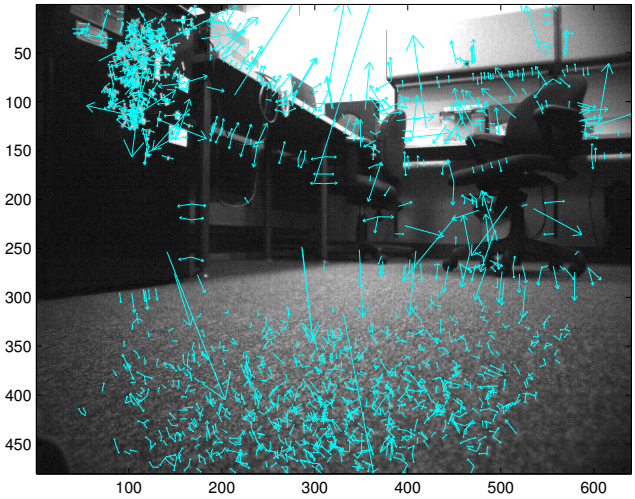


Figure 2.10: Example image with detected SIFT keypoints.

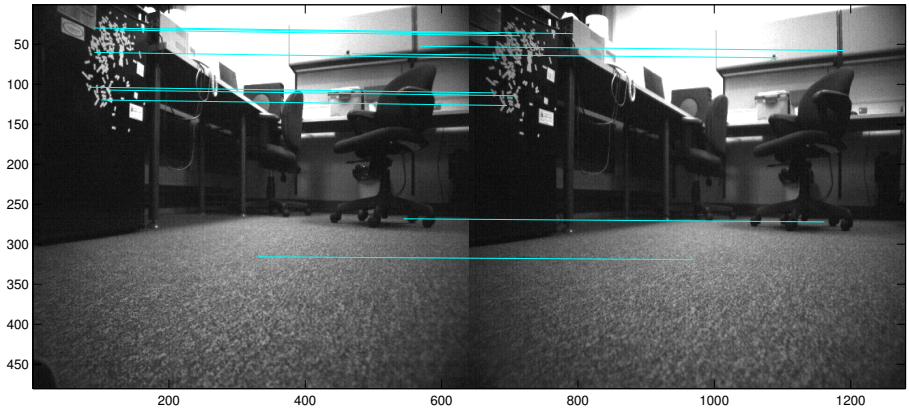


Figure 2.11: Example of matched SIFT keypoints in the left and right images.

matrix for the Power-SLAM estimator is closest to that of the EKF as compared to the SWKF and CKF. Thus, we can conclude that the Power-SLAM estimator is the least conservative approximate estimator, followed by the CKF and finally the SWKF. Fig. 2.13 compares the trace of the features' covariance matrix for the aforementioned four estimators. Here, only the covariance for features that have been re-observed has been included for comparison. From this figure, we see that the performance of the Power-SLAM estimator is almost indistinguishable from that of the EKF. Furthermore, with respect to the uncertainty in the features' position estimates, the Power-SLAM estimator is the least conservative as compared to the SWKF and the CKF. In summary, we note that by employing the Global-Map Postponement technique and the Power Method, the Power-SLAM estimator minimizes the information loss, while satisfying the linear-complexity constraint, and outperforms competing linear-processing-cost alternatives.

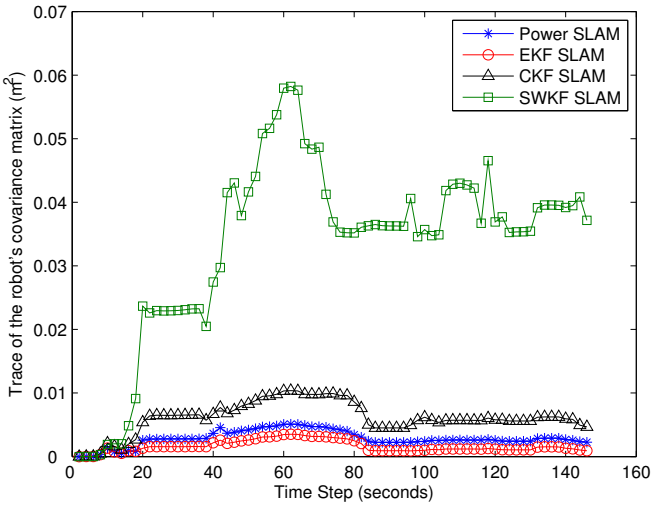


Figure 2.12: Trace of the robot's position covariance matrix.

2.6 Summary

The Power-SLAM algorithm, introduced in this chapter, provides a real-time consistent estimator for simultaneous localization and mapping that has computational complexity

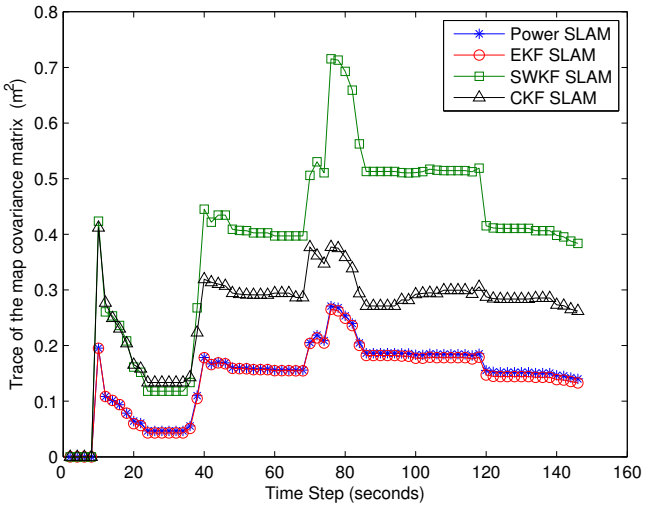


Figure 2.13: Trace of the map's covariance matrix.

linear in the number of features in the map. The Global-Map Postponement approach followed by the Power Method and linear-time rank-2 updates form the crux of the Power-SLAM algorithm. The Global-Map Postponement technique delays the approximations over multiple time steps. The Power Method extracts and retains the dominant information from the Kalman vectors generated during the postponement phase. By working in tandem, these two techniques *minimize* the information loss over *multiple* time steps. Finally, in order to increase the convergence rate of this estimator, linear-time rank-2 updates, which minimize the trace of the covariance matrix, are applied at every time step.

Moreover, a key advantage of the Power-SLAM estimator is its ability to adjust its processing requirements on-line to meet the availability of computational resources. By adaptively trading CPU cycles for estimation accuracy, Power-SLAM bridges the gap between linear-complexity estimators (based on coarse approximations, such as the SWKF and the CKF) and the quadratic-complexity EKF-based SLAM. Furthermore, by minimizing the information loss induced during the necessary approximations, the Power-SLAM algorithm is able to maximize estimation accuracy for a given number of operations.

Chapter 3

C-KLAM: Constrained Keyframe-based Localization and Mapping

In this chapter, we present C-KLAM, a Maximum A Posteriori (MAP) estimator-based keyframe approach for SLAM. As opposed to many existing keyframe-based SLAM approaches that discard information from non-keyframes for reducing the computational complexity the proposed C-KLAM presents a novel, elegant, and computationally-efficient technique for incorporating most of this information, using marginalization, resulting in improved estimation accuracy.

3.1 Introduction and Related Work

For mobile robots navigating in large environments over long time periods, one of the main challenges in designing an estimation algorithm for Simultaneous Localization and Mapping (SLAM) is its inherently high computational complexity. For example, the computational complexity of the Minimum Mean Squared Error (MMSE) estimator for SLAM, i.e., the Extended Kalman Filter (EKF) [19], is $O(N^2)$ for each update step, where N is the number of landmarks in the map. Similarly, for the batch MAP estimator-based SLAM (smoothing and mapping) [33], the worst-case computational

complexity is $O([K+N]^3)$, where K is the number of robot poses in the trajectory. While existing batch MAP-based SLAM approaches such as the $\sqrt{SAM}$ [33], g^2o [69], and Sparse Pose Adjustment (SPA) [70] generate efficient solutions by exploiting the sparsity of the information matrix, for large-scale SLAM with frequent loop closures, this cost eventually prohibits real-time operation.

The approximate solutions developed to reduce MAP-based SLAM's computational complexity can be classified into three main categories. The first one comprises of approaches such as iSAM [71] and iSAM2 [72] that *incrementally* optimize over all robot poses and landmarks, using *all* available measurement information. However, for trajectories with frequent loop closures, (i) fill-ins are generated between periodic batch updates for iSAM, when the number of constraints is greater than five times the number of robot poses [71], and (ii) many nodes in the Bayes tree used by iSAM2 have to be relinearized, hence degrading the efficiency of these approaches. The graphical SLAM approach of [73] provides efficient solutions by employing block coordinate descent-based minimization and by postponing relinearization. Besides the approximation used for minimizing the cost function, this method's accuracy also suffers due to the accumulation of linearization errors when frequent loop closures occur.

The second category includes fixed-lag smoothing approaches such as [74, 75] that consider a constant-size, sliding-window of recent robot poses and landmarks, along with measurements only in that time window. Here, old robot poses and landmarks are *marginalized*[1] and the corresponding measurements are discarded. However, marginalization destroys the sparsity of the information matrix, and the cost of this approach becomes $O(R^3)$, hence limiting the number of poses, R, in the sliding window. Moreover, this approach is unable to close loops for long trajectories.

The third category consists of *keyframe*-based approaches, such as PTAM [76]. PTAM processes measurement information from only a *subset* of all available views, hence information from non-keyframes is *discarded* (as opposed to marginalized) in order to retain the sparsity of the information matrix. Keyframe-based pose-graph approaches [77, 78, 79, 80, 81, 82], on the other hand, make use of all information from both key and non-key frames, but measurements from each frame are used multiple

[1] Given a joint probability distribution function $p(x,y)$ over random variables x and y, the marginal probability distribution function for x is given by $p(x) = \int_{-\infty}^{+\infty} p(x,y)dy$.

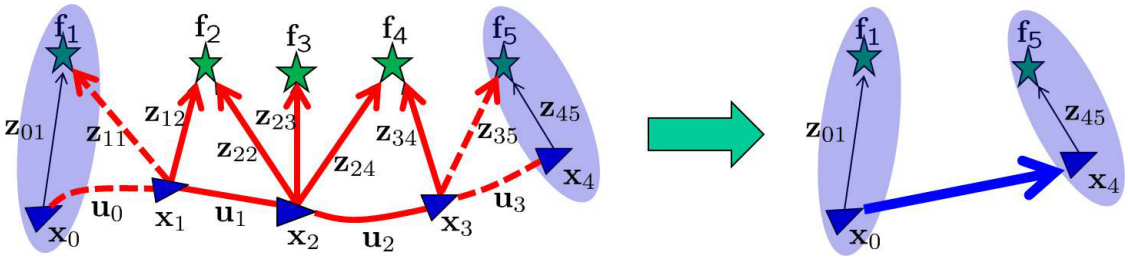


Figure 3.1: An example of the exploration epoch before (left) and after (right) the approximation employed in C-KLAM. $\mathbf{x}_0$, $\mathbf{x}_4$ are the keyframes to be retained, and $\mathbf{x}_1$, $\mathbf{x}_2$, and $\mathbf{x}_3$ are the non-keyframes to be marginalized. Similarly, $\mathbf{f}_1$, $\mathbf{f}_5$ are key landmarks (observed from the keyframes) to be retained, while $\mathbf{f}_2$, $\mathbf{f}_3$, and $\mathbf{f}_4$ are non-key landmarks (observed exclusively from the non-keyframes) to be marginalized. In the left figure, the arrows pictorially depict the measurements between different states. In the right figure, the blue arrow represents the pose constraint generated between the keyframes using C-KLAM and the structure of the resulting measurement graph.

times to generate relative pose-to-pose constraints, especially in loop closure events. Re-using information results in inconsistent estimates, hence degrading the estimation accuracy.

In this chapter, we present the Constrained Keyframe-based Localization and Mapping (C-KLAM), an approximate batch MAP-based algorithm, which estimates only keyframes (key robot poses) and key landmarks while also exploiting information (e.g., visual observations and odometry measurements) available to the non-keyframes. In particular, this information is projected onto the keyframes, by generating pose constraints between them. Our main contributions are as follows:

- In contrast to existing keyframe methods, C-KLAM utilizes both proprioceptive [e.g., from an inertial measurement unit (IMU)] and exteroceptive (e.g., from a camera) measurements from non-keyframes to generate pose constraints between the keyframes. This is achieved by marginalizing the non-keyframes along with the landmarks observed from them.

- In contrast to sliding-window approaches, C-KLAM incorporates information from marginalized frames and landmarks *without* destroying the sparsity of the information matrix, and hence generates fast and efficient solutions.

- The cost of marginalization in C-KLAM is cubic, $O(M_r^3)$, only in the number of non-keyframes, M_r, between consecutive keyframes, and *linear* in the number of landmarks, M_f, observed exclusively from the M_r non-keyframes, where $M_r \ll M_f$.

- The keyframes and the associated key landmarks are maintained over the entire robot trajectory, and thus C-KLAM enables efficient loop closures, necessary for ensuring accurate and consistent long-term navigation.

3.2 Algorithm Description

In this section, we first present a brief overview of batch MAP-based SLAM, followed by the details of the proposed C-KLAM algorithm. Moreover, to facilitate the description of these estimation algorithms, we will use the specific example scenario depicted in Fig. 3.1. Note, however, that C-KLAM is a general approach that can be used for any number of key and non-key poses and landmarks.

3.2.1 Batch MAP-based SLAM

Consider a robot, equipped with proprioceptive (e.g., IMU) and exteroceptive (e.g., camera) sensors, navigating in a 3D environment. The motion model for the robot is given by:

$$\mathbf{g}_{i+1} = \mathbf{x}_{i+1} - \mathbf{f}(\mathbf{x}_i, \mathbf{u}_i - \mathbf{w}_i) = \mathbf{0} \tag{3.1}$$

where $\mathbf{f}$ is a general nonlinear function[2] , $\mathbf{x}_i$ and $\mathbf{x}_{i+1}$ denote the robot poses at time-steps i and $i+1$, respectively, $\mathbf{u}_i = \mathbf{u}_{i_t} + \mathbf{w}_i$, is the measured control input (e.g., linear acceleration and rotational velocity for an IMU), where $\mathbf{u}_{i_t}$ denotes the true control input, and $\mathbf{w}_i$ is the zero-mean, white Gaussian measurement noise with covariance $\mathbf{Q}_i$. The measurement model for the robot at time-step i, obtaining an observation, $\mathbf{z}_{ij}$, to a landmark with position $\mathbf{f}_j$ is given by:

$$\mathbf{z}_{ij} = \mathbf{h}_{ij}(\mathbf{x}_i, \mathbf{f}_j) + \mathbf{v}_{ij} \tag{3.2}$$

[2] The details of the IMU motion model can be found in [74].

where $\mathbf{h}_{ij}$ is a general nonlinear measurement function[3] and $\mathbf{v}_{ij}$ is the zero-mean, white Gaussian measurement noise with covariance $\mathbf{R}_{ij}$.

Consider the current exploration epoch shown in Fig. 3.1. Here, the state vector $\mathbf{x}^{MAP} = \begin{bmatrix}\mathbf{x}_i^T & \mathbf{f}_j^T\end{bmatrix}^T$ consists of five robot poses, $\mathbf{x}_i$, $i = 0, 1, \ldots, 4$, and five point landmarks' positions, $\mathbf{f}_j$, $j = 1, 2, \ldots, 5$, observed from these poses. The objective of the batch MAP estimator is to calculate the state estimates, $\hat{\mathbf{x}}^{MAP}$, i.e., $\hat{\mathbf{x}}_{0:4}, \hat{\mathbf{f}}_{1:5}$, of *all* robot poses, $\mathbf{x}_{0:4} = \begin{bmatrix}\mathbf{x}_0^T \ldots \mathbf{x}_4^T\end{bmatrix}^T$, and *all* landmark positions, $\mathbf{f}_{1:5} = \begin{bmatrix}\mathbf{f}_1^T \ldots \mathbf{f}_5^T\end{bmatrix}^T$, using *all* available proprioceptive, $\mathbf{u}_{0:3} = \begin{bmatrix}\mathbf{u}_0^T \ldots \mathbf{u}_3^T\end{bmatrix}^T$, and exteroceptive, $\mathcal{Z}_{0:4} = \begin{bmatrix}\mathbf{z}_{01}^T & \mathbf{z}_{11}^T \ldots & \mathbf{z}_{35}^T & \mathbf{z}_{45}^T\end{bmatrix}^T$, measurements, where $\mathcal{Z}_i$ denotes the set of all exteroceptive measurements obtained at robot pose $\mathbf{x}_i$, $i = 0, 1, \ldots, 4$. Specifically, the batch MAP estimates, $\hat{\mathbf{x}}^{MAP}$, are obtained by solving the following optimization problem:

$$\hat{\mathbf{x}}^{MAP} \triangleq \arg\max_{\mathbf{x}_{0:4}, \mathbf{f}_{1:5}} \; p(\mathbf{x}_{0:4}, \mathbf{f}_{1:5} | \mathcal{Z}_{0:4}, \mathbf{u}_{0:3}) \tag{3.3}$$

$$\propto \arg\max_{\mathbf{x}_{0:4}, \mathbf{f}_{1:5}} \; p(\mathbf{x}_0) \prod_{i=0}^{3} p(\mathbf{x}_{i+1} | \mathbf{x}_i, \mathbf{u}_i) \prod_{\mathbf{z}_{ij} \in \mathcal{Z}_{0:4}} p(\mathbf{z}_{ij} | \mathbf{x}_i, \mathbf{f}_j) \tag{3.4}$$

Note that the equivalent optimization problem in (3.4) is obtained from (3.3) using Bayes' rule, the Markovian assumption, and the assumption of independence of the process and measurement noise. Here, $p(\mathbf{x}_0)$ denotes the Gaussian prior, $\mathbf{x}_0 \sim \mathcal{N}(\hat{\mathbf{x}}_{0|0}, \mathbf{P}_{0|0})$, for the robot pose at time-step 0. Finally, using the Gaussian noise assumptions in (3.1) and (3.2), and the monotonicity of the logarithmic function, (3.4) is equivalent to minimizing the following nonlinear least-squares problem:

$$\hat{\mathbf{x}}^{MAP} \triangleq \arg\min_{\mathbf{x}_{0:4}, \mathbf{f}_{1:5}} \left(\frac{1}{2} ||\mathbf{x}_0 - \hat{\mathbf{x}}_{0|0}||^2_{\mathbf{P}_{0|0}} + \sum_{i=0}^{3} \frac{1}{2} ||\mathbf{x}_{i+1} - \mathbf{f}(\mathbf{x}_i, \mathbf{u}_i)||^2_{\mathbf{Q}'_i} + \sum_{\mathbf{z}_{ij} \in \mathcal{Z}_{0:4}} \frac{1}{2} ||\mathbf{z}_{ij} - \mathbf{h}_{ij}(\mathbf{x}_i, \mathbf{f}_j)||^2_{\mathbf{R}_{ij}} \right) \tag{3.5}$$

$$\triangleq \arg\min_{\mathbf{x}_{0:4}, \mathbf{f}_{1:5}} \left(\mathcal{C}_P(\mathbf{x}_0; \hat{\mathbf{x}}_{0|0}) + \sum_{i=0}^{3} \mathcal{C}_M(\mathbf{x}_{i+1}, \mathbf{x}_i; \mathbf{u}_i) + \sum_{\mathbf{z}_{ij} \in \mathcal{Z}_{0:4}} \mathcal{C}_O(\mathbf{x}_i, \mathbf{f}_j; \mathbf{z}_{ij}) \right) \tag{3.6}$$

$$\triangleq \arg\min_{\mathbf{x}_{0:4}, \mathbf{f}_{1:5}} \mathcal{C}(\mathbf{x}_{0:4}, \mathbf{f}_{1:5}; \mathcal{Z}_{0:4}, \mathbf{u}_{0:3}) \triangleq \arg\min_{\mathbf{x}_{0:4}, \mathbf{f}_{1:5}} \mathcal{C}(\mathbf{x}^{MAP}) \tag{3.7}$$

[3] The details of the camera measurement model for point features, used in our experiments, can be found in [74].

where $||\mathbf{e}||^2_{\mathbf{W}} = \mathbf{e}^T\mathbf{W}^{-1}\mathbf{e}$ is the weighted squared L_2-norm for a given covariance $\mathbf{W}$, $\mathbf{Q}'_i = \mathbf{G}_i\mathbf{Q}_i\mathbf{G}_i^T$, and $\mathbf{G}_i$ is the Jacobian of $\mathbf{f}(\cdot)$ in (3.1) with respect to the noise $\mathbf{w}_i$. In (3.6), we denote the cost terms arising from the prior, the robot motion model, and the exteroceptive observations by $\mathcal{C}_P$, $\mathcal{C}_M$, and $\mathcal{C}_O$, respectively [see (3.5)].

$\mathcal{C}(\mathbf{x}^{MAP})$ is a nonlinear least squares cost function, and a standard approach to determine its minimum is to employ Gauss-Newton iterative minimization [83]. Specifically, at the ℓ-th iteration of this method, a correction, $\delta\mathbf{x}^{MAP^{(\ell)}}$, to the current estimate, $\hat{\mathbf{x}}^{MAP}$, is computed by minimizing the second-order Taylor-series approximation of the cost function which is given by:

$$\mathcal{C}(\hat{\mathbf{x}}^{MAP^{(\ell)}} + \delta\mathbf{x}^{MAP^{(\ell)}}) \simeq \mathcal{C}(\hat{\mathbf{x}}^{MAP^{(\ell)}}) + \mathbf{g}^{(\ell)^T}\delta\mathbf{x}^{MAP^{(\ell)}} + \frac{1}{2}\delta\mathbf{x}^{MAP^{(\ell)^T}}\mathbf{H}^{(\ell)}\delta\mathbf{x}^{MAP^{(\ell)}} \tag{3.8}$$

where

$$\mathbf{g}^{(\ell)} \triangleq \nabla_{\mathbf{x}^{MAP}}\mathcal{C}(\cdot)\Big|_{\{\hat{\mathbf{x}}^{MAP^{(\ell)}}\}} \tag{3.9}$$

$$\mathbf{H}^{(\ell)} \triangleq \nabla^2_{\mathbf{x}^{MAP}}\mathcal{C}(\cdot)\Big|_{\{\hat{\mathbf{x}}^{MAP^{(\ell)}}\}} \tag{3.10}$$

are the gradient and Hessian of $\mathcal{C}(\cdot)$ with respect to $\mathbf{x}^{MAP}$, evaluated at the current state estimate, $\hat{\mathbf{x}}^{MAP^{(\ell)}}$, respectively.

We now examine the structure of the Jacobian and Hessian matrices which will be used in the ensuing analysis. Specifically, at the ℓ-th iteration, $\mathbf{g}^{(\ell)}$ is [see (3.12) and (3.14)]:

$$\begin{aligned}\mathbf{g}^{(\ell)} = \mathbf{\Pi}^T\mathbf{P}^{-1}_{0|0}\left(\hat{\mathbf{x}}^{(\ell)}_0 - \hat{\mathbf{x}}_{0|0}\right) + \sum_{i=0}^{3}\mathbf{\Phi}^{(\ell)^T}_i\mathbf{Q}'^{-1}_i\left(\hat{\mathbf{x}}^{(\ell)}_{i+1} - \mathbf{f}(\hat{\mathbf{x}}^{(\ell)}_i, \mathbf{u}_i)\right) \\ - \sum_{\mathbf{z}_{ij}\in\mathcal{Z}_{0:4}}\mathbf{H}^{(\ell)^T}_{ij}\mathbf{R}^{-1}_{ij}\Big(\mathbf{z}_{ij} - \mathbf{h}_{ij}(\hat{\mathbf{x}}^{(\ell)}_i, \hat{\mathbf{f}}^{(\ell)}_j)\Big)\end{aligned} \tag{3.11}$$

where

$$\begin{aligned}
\mathbf{\Pi} &= \begin{bmatrix}\mathbf{I}_{|\mathbf{x}|} & \mathbf{0} & \cdots & \mathbf{0}\end{bmatrix} \\
\mathbf{\Phi}_i &\triangleq \left.\frac{\partial \mathbf{g}_i}{\partial \mathbf{x}^{MAP}}\right|_{\{\hat{\mathbf{x}}^{MAP},\mathbf{0}\}} \\
&= \begin{bmatrix}\mathbf{0}_{|\mathbf{x}|\times|\mathbf{x}|} & \cdots & \mathbf{\Phi}_{x_i} & \mathbf{I}_{|\mathbf{x}|} & \mathbf{0}_{|\mathbf{x}|\times|\mathbf{x}|} & \cdots & \mathbf{0}_{|\mathbf{x}|\times|\mathbf{f}|} & \cdots & \mathbf{0}_{|\mathbf{x}|\times|\mathbf{f}|}\end{bmatrix} \qquad (3.12) \\
\mathbf{G}_i &\triangleq \left.\frac{\partial \mathbf{g}_i}{\partial \mathbf{w}_i}\right|_{\{\hat{\mathbf{x}}^{MAP},\mathbf{0}\}} = \begin{bmatrix}\mathbf{0}_{|\mathbf{x}|\times|\mathbf{w}|} & \cdots & \mathbf{G}_{x_i} & \mathbf{0}_{|\mathbf{x}|\times|\mathbf{w}|} & \cdots & \mathbf{0}_{|\mathbf{x}|\times|\mathbf{w}|}\end{bmatrix} \qquad (3.13) \\
\mathbf{H}_{ij} &\triangleq \left.\frac{\partial \mathbf{h}_{ij}}{\partial \mathbf{x}^{MAP}}\right|_{\{\hat{\mathbf{x}}^{MAP}\}} \\
&= \begin{bmatrix}\mathbf{0}_{|\mathbf{z}|\times|\mathbf{x}|} & \cdots & \mathbf{H}_{x_{ij}} & \mathbf{0}_{|\mathbf{z}|\times|\mathbf{x}|} & \cdots & \mathbf{H}_{f_{ij}} & \mathbf{0}_{|\mathbf{z}|\times|\mathbf{f}|} & \cdots & \mathbf{0}_{|\mathbf{z}|\times|\mathbf{f}|}\end{bmatrix} \qquad (3.14)
\end{aligned}$$

In the above equations, $|\mathbf{x}|$, $|\mathbf{f}|$, $|\mathbf{w}|$, and $|\mathbf{z}|$ denote the cardinality of the robot pose, $\mathbf{x}_i$, the landmark position, $\mathbf{f}_j$, the process noise, $\mathbf{w}_i$, and the exteroceptive measurement, $\mathbf{z}_{ij}$, respectively. In addition, $\mathbf{I}_{|\mathbf{x}|}$ is an identity matrix of dimensions $|\mathbf{x}|$ and

$$\mathbf{\Phi}_{x_i} \triangleq \left.\frac{\partial \mathbf{g}_i}{\partial \mathbf{x}_i}\right|_{\{\hat{\mathbf{x}}_i,\mathbf{0}\}},\ \mathbf{G}_{x_i} \triangleq \left.\frac{\partial \mathbf{g}_i}{\partial \mathbf{w}_i}\right|_{\{\hat{\mathbf{x}}_i,\mathbf{0}\}},\ \mathbf{H}_{x_{ij}} \triangleq \left.\frac{\partial \mathbf{h}_{ij}}{\partial \mathbf{x}_i}\right|_{\{\hat{\mathbf{x}}_i,\hat{\mathbf{f}}_j\}},\ \mathbf{H}_{f_{ij}} \triangleq \left.\frac{\partial \mathbf{h}_{ij}}{\partial \mathbf{f}_j}\right|_{\{\hat{\mathbf{x}}_i,\hat{\mathbf{f}}_j\}} \qquad (3.15)$$

The Hessian matrix, $\mathbf{H}^{(\ell)}$, is approximated in the Gauss-Newton method by [see (3.12) and (3.14)]:

$$\mathbf{H}^{(\ell)} = \mathbf{\Pi}^T \mathbf{P}_{0|0}^{-1} \mathbf{\Pi} + \sum_{i=0}^{3} \mathbf{\Phi}_i^{(\ell)^T} \mathbf{Q}_i'^{-1} \mathbf{\Phi}_i^{(\ell)} + \sum_{\mathbf{z}_{ij}\in\mathcal{Z}_{0:4}} \mathbf{H}_{ij}^{(\ell)^T} \mathbf{R}_{ij}^{-1} \mathbf{H}_{ij}^{(\ell)} \qquad (3.16)$$

which is a good approximation for small-residual problems. The value $\delta\mathbf{x}^{MAP^{(\ell)}}$ that minimizes (3.8) is found by solving the following system of linear equations:

$$\mathbf{H}^{(\ell)} \delta\mathbf{x}^{MAP^{(\ell)}} = -\mathbf{g}^{(\ell)} \qquad (3.17)$$

Due to the sparse structure of the matrices $\mathbf{H}_{ij}^{(\ell)}$ and $\mathbf{\Phi}_i^{(\ell)}$ [see (3.12) and (3.14)], the matrix $\mathbf{H}^{(\ell)}$ is also sparse, which can be exploited to speed-up the solution of the linear system in (3.17) [83]. Once $\delta\mathbf{x}^{MAP^{(\ell)}}$ is found, the new state estimate is updated as:

$$\hat{\mathbf{x}}^{MAP^{(\ell+1)}} = \hat{\mathbf{x}}^{MAP^{(\ell)}} \oplus \delta\mathbf{x}^{MAP^{(\ell)}} \qquad (3.18)$$

where $\oplus$ is the corresponding update rule. Given an initial estimate $\hat{\mathbf{x}}^{MAP^{(0)}}$ that resides within the attraction basin of the global optimum, this iterative algorithm will compute the global minimum (i.e., MAP estimate) for the entire state given all measurements up to time-step k.

As stated above, the cost of solving the normal equations in (3.17) depends upon the sparsity-pattern of the Hessian and the worst-case computational complexity can be as high as $O([K+N]^3)$, where K and N are the number of robot poses and landmarks, respectively. The Hessian matrix $\mathbf{H}$ corresponding to batch MAP-based SLAM has a typical sparse structure shown in Fig. 3.2. Specifically, $\mathbf{H}$ is a symmetric positive definite matrix, where the diagonal block corresponding to the robot poses has a block tri-diagonal structure, the diagonal block corresponding to landmark positions has block-diagonal structure, while the off-diagonal blocks are generally sparse depending upon the robot's motion. This sparse structure of $\mathbf{H}$ has been exploited by existing approaches to reduce the computational cost of solving these systems of linear equations. However, note that as the robot explores the environment and observes new landmarks, the size of the optimization problem (both K and N) in (3.5) continuously increases. Therefore, for long trajectories with many features and frequent loop closures, the cost of solving (3.5) may prohibit real-time operation.

3.2.2 Marginalization and Naïve Approximation-based SLAM

In order to reduce the computational complexity of MAP-based SLAM and ensure accurate and real-time navigation over long time durations, several variants of keyframe-based approaches have been proposed in the literature. The key idea behind all these approaches is to limit the size of the optimization problem in (3.7) by building a sparse map of the environment consisting of *only* the key robot poses and the distinctive landmarks observed from these key poses.

Specifically, for the example in Fig. 3.1, let us assume that we retain $\mathbf{x}_K = \begin{bmatrix}\mathbf{x}_0^T & \mathbf{x}_4^T & \mathbf{f}_1^T & \mathbf{f}_5\end{bmatrix}^T$ where: (i) $\mathbf{x}_0$ and $\mathbf{x}_4$ are key poses, and (ii) $\mathbf{f}_1$ and $\mathbf{f}_5$, are key landmarks observed from these key poses.[4] Non-key poses and landmarks are denoted by

[4] Note that we retain only two key poses/landmarks in this example, in order to simplify the explanation. However, any number of key poses/landmarks can be retained in keyframe-based approaches. The key poses are selected based on certain criteria, e.g., distance traveled between two key poses, poses that observe points of interest, uniqueness of a image, etc. Furthermore, for the example in Fig. 3.1,

$\mathbf{x}_M = \left[\mathbf{x}_{1:3}^T \ \mathbf{f}_{2:4}^T\right]^T$. In this case, (3.5) can be split into two parts as follows:

$$\mathcal{C} = \underbrace{\mathcal{C}_P(\mathbf{x}_0; \hat{\mathbf{x}}_{0|0}) + \mathcal{C}_O(\mathbf{x}_0, \mathbf{f}_1; \mathbf{z}_{01}) + \mathcal{C}_O(\mathbf{x}_4, \mathbf{f}_5; \mathbf{z}_{45})}_{\mathcal{C}_1(\mathbf{x}_0, \mathbf{x}_4, \mathbf{f}_1, \mathbf{f}_5; \hat{\mathbf{x}}_{0|0}, \mathbf{z}_{01}, \mathbf{z}_{45})} + \underbrace{\sum_{i=0}^{3} \mathcal{C}_M(\mathbf{x}_{i+1}, \mathbf{x}_i; \mathbf{u}_i) + \sum_{\mathbf{z}_{ij} \in \mathcal{Z}_{1:3}} \mathcal{C}_O(\mathbf{x}_i, \mathbf{f}_j; \mathbf{z}_{ij})}_{\mathcal{C}_2(\mathbf{x}_{1:3}, \mathbf{f}_{2:4}, \mathbf{x}_0, \mathbf{x}_4, \mathbf{f}_1, \mathbf{f}_5; \mathcal{Z}_{1:3}, \mathbf{u}_{0:3})} \tag{3.19}$$

The first part of the cost function, $\mathcal{C}_1$, depends only upon the key poses, key landmarks, and the measurements between them (denoted by thin black arrows in Fig. 3.1). This part consists of cost terms arising from the prior term and from the two exteroceptive measurements, $\mathbf{z}_{01}$ and $\mathbf{z}_{45}$, obtained at the key poses $\mathbf{x}_0$ and $\mathbf{x}_4$, respectively. The second part of the cost function, $\mathcal{C}_2$, contains all cost terms that involve non-key poses and non-key landmarks. Specifically, these correspond to two types of cost terms: (i) those that involve *only* non-key poses and non-key landmarks (corresponding to measurements denoted by solid red lines in Fig. 3.1), e.g., $\mathcal{C}_O(\mathbf{x}_1, \mathbf{f}_2; \mathbf{z}_{12})$, and (ii) those that involve *both* key and non-key elements (corresponding to measurements denoted by dashed red lines in Fig. 3.1), e.g., $\mathcal{C}_O(\mathbf{x}_1, \mathbf{f}_1; \mathbf{z}_{11})$ and $\mathcal{C}_M(\mathbf{x}_1, \mathbf{x}_0; \mathbf{u}_0)$.

Classical keyframe-based approaches, such as PTAM [76], optimize only over $\mathcal{C}_1$ in order to reduce the computational complexity, i.e., the cost terms in $\mathcal{C}_2$ and the corresponding measurements are discarded, resulting in substantial computational savings but also significant information loss. An alternative approach, to retain a part of the information in $\mathcal{C}_2$ instead of discarding it completely, is to *marginalize* the non-key poses and landmarks in $\mathbf{x}_M$. Mathematically, this is formulated as the following optimization problem:

$$\min_{\mathbf{x}_K, \mathbf{x}_M} \mathcal{C}(\mathbf{x}_K, \mathbf{x}_M) = \min_{\mathbf{x}_K} \left(\min_{\mathbf{x}_M} \mathcal{C}(\mathbf{x}_K, \mathbf{x}_M) \right) = \min_{\mathbf{x}_K} \left(\mathcal{C}_1(\mathbf{x}_K) + \min_{\mathbf{x}_M} \mathcal{C}_2(\mathbf{x}_K, \mathbf{x}_M) \right) \tag{3.20}$$

By employing the second-order Taylor-series approximation to $\mathcal{C}_2$, similar to (3.8), and minimizing with respect to $\mathbf{x}_M$, we can approximate $\mathcal{C}_2$ by another cost function

we assume that the depth to the features is available (e.g., from an RGB-D camera), in order to reduce the number of measurements and poses required. If a regular 2D imaging camera is used, at least two observations of a key feature and the corresponding poses will need to be retained.

$\mathcal{C}'_2$ as follows (see Fig. 3.3):

$$\begin{aligned}\mathcal{C} &\simeq \mathcal{C}'(\mathbf{x}_0, \mathbf{x}_4, \mathbf{f}_1, \mathbf{f}_5; \hat{\mathbf{x}}_{0|0}, \mathbf{z}_{01}, \mathbf{z}_{45}, \hat{\mathbf{x}}_0, \hat{\mathbf{x}}_4, \hat{\mathbf{f}}_1, \hat{\mathbf{f}}_5) \\ &= \mathcal{C}_1 + \mathcal{C}'_2(\mathbf{x}_0, \mathbf{x}_4, \mathbf{f}_1, \mathbf{f}_5; \hat{\mathbf{x}}_0, \hat{\mathbf{x}}_4, \hat{\mathbf{f}}_1, \hat{\mathbf{f}}_5) \end{aligned} \tag{3.21}$$

where,

$$\mathcal{C}'_2 = \alpha' + \mathbf{g}_{\mathcal{C}'_2}^T \begin{bmatrix} \mathbf{x}_0 - \hat{\mathbf{x}}_0 \\ \mathbf{x}_4 - \hat{\mathbf{x}}_4 \\ \mathbf{f}_1 - \hat{\mathbf{f}}_1 \\ \mathbf{f}_5 - \hat{\mathbf{f}}_5 \end{bmatrix} + \frac{1}{2} \begin{bmatrix} \mathbf{x}_0 - \hat{\mathbf{x}}_0 \\ \mathbf{x}_4 - \hat{\mathbf{x}}_4 \\ \mathbf{f}_1 - \hat{\mathbf{f}}_1 \\ \mathbf{f}_5 - \hat{\mathbf{f}}_5 \end{bmatrix}^T \mathbf{H}_{\mathcal{C}'_2} \begin{bmatrix} \mathbf{x}_0 - \hat{\mathbf{x}}_0 \\ \mathbf{x}_4 - \hat{\mathbf{x}}_4 \\ \mathbf{f}_1 - \hat{\mathbf{f}}_1 \\ \mathbf{f}_5 - \hat{\mathbf{f}}_5 \end{bmatrix} \tag{3.22}$$

with,

$$\mathbf{H}_{\mathcal{C}'_2} = \begin{bmatrix} \mathbf{A}_k & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_b \end{bmatrix} - \begin{bmatrix} \mathbf{B}_k & \mathbf{0} \\ \mathbf{B}_b & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{A}_r & \mathbf{A}_{rf} \\ \mathbf{A}_{fr} & \mathbf{A}_f \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{B}_k^T & \mathbf{B}_b^T \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \tag{3.23}$$

$$\mathbf{g}_{\mathcal{C}'_2} = \begin{bmatrix} \mathbf{g}_k \\ \mathbf{g}_b \end{bmatrix} - \begin{bmatrix} \mathbf{B}_k & \mathbf{0} \\ \mathbf{B}_b & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{A}_r & \mathbf{A}_{rf} \\ \mathbf{A}_{fr} & \mathbf{A}_f \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{g}_r \\ \mathbf{g}_f \end{bmatrix} \triangleq \begin{bmatrix} \mathbf{g}_{\mathcal{C}'_2,k} \\ \mathbf{g}_{\mathcal{C}'_2,b} \end{bmatrix}. \tag{3.24}$$

Here, $\hat{\mathbf{x}}_0$, $\hat{\mathbf{x}}_4$, $\hat{\mathbf{f}}_1$, and $\hat{\mathbf{f}}_5$ are the estimates of $\mathbf{x}_0$, $\mathbf{x}_4$, $\mathbf{f}_1$, and $\mathbf{f}_5$, respectively, at the time of marginalization, α' is a constant term independent of the optimization variables, and $\mathbf{g}_k, \mathbf{g}_b, \mathbf{g}_r$, and $\mathbf{g}_f$ are the gradient vectors of $\mathcal{C}_2$ with respect to $\{\mathbf{x}_0, \mathbf{x}_4\}, \{\mathbf{f}_1, \mathbf{f}_5\}, \{\mathbf{x}_{1:3}\}$, and $\{\mathbf{f}_{2:4}\}$, respectively. Also, $\mathbf{g}_{\mathcal{C}'_2}$ and $\mathbf{H}_{\mathcal{C}'_2}$ denote the Jacobian and Hessian matrix, respectively. Lastly, we note that $\mathbf{H}_{\mathcal{C}'_2}$, as expected, is the Schur complement of the diagonal block, corresponding to non-key poses and non-key landmarks, of the Hessian, $\mathbf{H}_{\mathcal{C}_2}$, of the original cost function, $\mathcal{C}_2$ (see Fig. 3.3).

As expected, however, marginalization of non-key elements creates additional constraints between the key poses and the key landmarks, which directly translates into fill-ins in the reduced Hessian matrix, $\mathbf{H}_{\mathcal{C}'_2}$. This destroys the sparse structure of the Hessian matrix, $\mathbf{H}_{\mathcal{C}'} = \mathbf{H}_{\mathcal{C}_1} + \mathbf{H}_{\mathcal{C}'_2}$, that corresponds to the cost function $\mathcal{C}'$ [see (3.21)], where $\mathbf{H}_{\mathcal{C}_1}$ is the Hessian corresponding to cost function $\mathcal{C}_1$, and substantially increases the computational cost of obtaining a solution to the minimization problem, even with the reduced size of the optimization problem. By studying the relationship between the measurement graph corresponding to Fig. 3.1 and the sparsity pattern of the resulting Hessian matrix, we note that the exteroceptive measurements from non-key poses to key features, i.e., $\mathbf{z}_{11}$ and $\mathbf{z}_{35}$, are the ones responsible for generating fill-ins in the Hessian

matrix, $\mathbf{H}_{\mathcal{C}'}$, after marginalization.[5]

A straightforward solution to retain the sparsity of the Hessian matrix would be to first discard any exteroceptive measurements between non-key poses and key features (e.g., $\mathbf{z}_{11}$ and $\mathbf{z}_{35}$ in Fig. 3.1), and then proceed with the marginalization of non-key elements. However, in real-world scenarios, $\mathbf{f}_1$ and $\mathbf{f}_5$ are not single features, but they each correspond to a group of features. Hence, such an approximation would discard numerous measurements, resulting in substantial information loss.

3.2.3 C-KLAM Algorithm

In order to address this problem and maintain the sparse structure of the Hessian (information) matrix while incorporating information from $\mathcal{C}_2$, the proposed C-KLAM approach (i) builds a sparse map of the environment consisting of *only* the key robot poses and the distinctive landmarks observed from these key poses, and (ii) uses measurement information from non-key poses to create constraints between the key poses, so as to improve estimation accuracy. To achieve these two objectives, C-KLAM carries out an additional approximation step, i.e., it further approximates $\mathcal{C}_2'$ in (3.21) by a quadratic cost term, $\mathcal{C}_2''(\mathbf{x}_0, \mathbf{x}_4; \hat{\mathbf{x}}_0, \hat{\mathbf{x}}_4)$ that constraints *only* the key poses $\mathbf{x}_0$ and $\mathbf{x}_4$.

Specifically, along with the non-key poses/landmarks, C-KLAM *marginalizes the key landmarks* $\mathbf{f}_1$ *and* $\mathbf{f}_5$, but *only* from $\mathcal{C}_2$; these key landmarks will still appear as optimization variables in $\mathcal{C}_1$ [see (3.19)]. Moreover, marginalizing $\mathbf{f}_1$ and $\mathbf{f}_5$ from $\mathcal{C}_2$, while retaining them in $\mathcal{C}_1$, implies that we ignore their data association[6] and treat them as different features (say $\mathbf{f}_1'$ and $\mathbf{f}_5'$) in $\mathcal{C}_2$. Mathematically, this process can be described by first considering the following equivalent optimization problems [see (3.5), (3.19), and

[5] Note that the proprioceptive measurements between key and non-key poses, i.e., $\mathbf{u}_0$ and $\mathbf{u}_3$, also generate fill-ins, but these fill-ins are desirable for our problem as they represent constraints between two consecutive key poses after marginalization.

[6] Besides the inability to relinearize marginalized states, ignoring this data association is the main information loss incurred by C-KLAM as compared to the batch MAP-based SLAM.

Fig. 3.4]:

$$\begin{aligned}
&\min \mathcal{C}(\mathbf{x}_{0:4}, \mathbf{f}_{1:5}; \mathcal{Z}_{0:4}, \mathbf{u}_{0:3}) \\
&\Leftrightarrow \min \bar{\mathcal{C}}(\mathbf{x}_{0:4}, \mathbf{f}_{1:5}, \mathbf{f}'_1, \mathbf{f}'_5; \mathcal{Z}_{0:4}, \mathbf{u}_{0:3}) \\
&\quad \text{s.t. } \mathbf{f}_1 = \mathbf{f}'_1, \mathbf{f}_5 = \mathbf{f}'_5
\end{aligned} \tag{3.25}$$

where,

$$\bar{\mathcal{C}} = \mathcal{C}_1(\mathbf{x}_0, \mathbf{x}_4, \mathbf{f}_1, \mathbf{f}_5; \hat{\mathbf{x}}_{0|0}, \mathbf{z}_{01}, \mathbf{z}_{45}) + \bar{\mathcal{C}}_2(\mathbf{x}_{1:3}, \mathbf{f}_{2:4}, \mathbf{x}_0, \mathbf{x}_4, \mathbf{f}'_1, \mathbf{f}'_5; \mathcal{Z}_{1:3}, \mathbf{u}_{0:3}) \tag{3.26}$$

Note that minimizing the batch-MAP cost function in (3.5) is *exactly* equivalent to the constrained optimization problem presented in (3.25). Now, in order to maintain the sparsity of the Hessian matrix after marginalizing the non-key elements, C-KLAM *discards* the constraint in (3.25) and hence assumes that the features $\mathbf{f}'_1$ and $\mathbf{f}'_5$ are distinct from $\mathbf{f}_1$ and $\mathbf{f}_5$, respectively (see Fig. 3.4). Due to this relaxation, $\bar{\mathcal{C}}_2$ no longer depends on the key features $\mathbf{f}_1$ and $\mathbf{f}_5$, and hence has *no* cost terms corresponding to measurements between non-key poses and key features. Due to this approximation, C-KLAM can now marginalize the features $\mathbf{f}'_1$ and $\mathbf{f}'_5$, along with the non-key elements $\mathbf{x}_{1:3}$ and $\mathbf{f}_{2:4}$, from $\bar{\mathcal{C}}$ in (3.26), thus ensuring that the resulting Hessian matrix remains sparse.

Specifically, C-KLAM approximates $\bar{\mathcal{C}}_2$ in (3.26) by $\mathcal{C}''_2$ by minimizing the second-order Taylor-series approximation of $\bar{\mathcal{C}}_2$ with respect to $\mathbf{f}'_1$ and $\mathbf{f}'_5$ [see Figs. 3.3 and 3.5]:

$$\begin{aligned}
\bar{\mathcal{C}}_2 &\simeq \mathcal{C}''_2(\mathbf{x}_0, \mathbf{x}_4; \hat{\mathbf{x}}_0, \hat{\mathbf{x}}_4) \\
&= \alpha'' + \mathbf{g}_{\mathcal{C}''_2}^T \begin{bmatrix} \mathbf{x}_0 - \hat{\mathbf{x}}_0 \\ \mathbf{x}_4 - \hat{\mathbf{x}}_4 \end{bmatrix} + \frac{1}{2} \begin{bmatrix} \mathbf{x}_0 - \hat{\mathbf{x}}_0 \\ \mathbf{x}_4 - \hat{\mathbf{x}}_4 \end{bmatrix}^T \mathbf{H}_{\mathcal{C}''_2} \begin{bmatrix} \mathbf{x}_0 - \hat{\mathbf{x}}_0 \\ \mathbf{x}_4 - \hat{\mathbf{x}}_4 \end{bmatrix}
\end{aligned} \tag{3.27}$$

where,

$$\mathbf{H}_{\mathcal{C}''_2} = \mathbf{A}_k - \mathbf{B}_k(\mathbf{D} - \mathbf{B}_b^T \mathbf{A}_b^{-1} \mathbf{B}_b)^{-1} \mathbf{B}_k^T \tag{3.28}$$

$$\begin{aligned}
\mathbf{g}_{\mathcal{C}''_2} &= \mathbf{g}_{\mathcal{C}'_2,k} + \mathbf{B}_k \mathbf{D}^{-1} \mathbf{B}_b^T \\
&\quad \cdot (\mathbf{A}_b^{-1} + \mathbf{A}_b^{-1} \mathbf{B}_b (\mathbf{D} - \mathbf{B}_b^T \mathbf{A}_b^{-1} \mathbf{B}_b)^{-1} \mathbf{B}_b^T \mathbf{A}_b^{-1}) \mathbf{g}_{\mathcal{C}'_2,b}
\end{aligned} \tag{3.29}$$

and

$$\mathbf{D} = \mathbf{A}_r - \mathbf{A}_{rf} \mathbf{A}_f^{-1} \mathbf{A}_{fr}. \tag{3.30}$$

where α'' is a constant, independent of the optimization variables, and $\mathbf{g}_{\mathcal{C}''_2}$, $\mathbf{H}_{\mathcal{C}''_2}$ denote the Jacobian and Hessian matrix, respectively.

After this approximation, the final C-KLAM cost function becomes:

$$\mathcal{C}_{CKLAM} = \mathcal{C}_1(\mathbf{x}_0, \mathbf{x}_4, \mathbf{f}_1, \mathbf{f}_5; \hat{\mathbf{x}}_{0|0}, \mathbf{z}_{01}, \mathbf{z}_{45}) + \mathcal{C}_2''(\mathbf{x}_0, \mathbf{x}_4; \hat{\mathbf{x}}_0, \hat{\mathbf{x}}_4) \tag{3.31}$$

whose corresponding Hessian would be the same as that of $\mathcal{C}_1$ (and thus sparse) plus an additional information (relative pose) constraint between $\mathbf{x}_0$ and $\mathbf{x}_4$ due to $\mathcal{C}_2''$. In summary, by approximating $\mathcal{C}_2$ by $\mathcal{C}_2''$, C-KLAM is able to incorporate most of the information from the non-key poses/landmarks, while maintaining the sparsity of the Hessian matrix. Moreover, the part of the cost function, $\mathcal{C}_1$, corresponding to the key poses/landmarks, remains intact.

Lastly, we show that the approximation (marginalization) described above can be carried out with cost cubic in the number of marginalized non-key poses, and only linear in the number of marginalized non-key landmarks. For the complexity analysis, let us assume that we have M_r non-key poses and M_f non-key features to be marginalized, and M_b features that are observed from both key and non-key frames, where $M_f \gg M_r$ and $M_f \gg M_b$. The marginalization step involves the computation of the Hessian matrix, $\mathbf{H}_{\mathcal{C}_2''}$, and the Jacobian, $\mathbf{g}_{\mathcal{C}_2''}$, according to (3.28) - (3.30). For computing both the Hessian and the Jacobian, we first need to calculate $\mathbf{D}$ in (3.30). Since $\mathbf{A}_f$ is block-diagonal, $\mathbf{A}_f^{-1}$ in (3.30) can be computed with cost only $\mathcal{O}(M_f)$. Moreover, since the number of marginalized non-key features, M_f, far exceeds M_r and M_b, the cost of computing $\mathbf{D}$ remains $\mathcal{O}(M_f)$. To compute the Hessian [see (3.28)], note that $\mathbf{A}_b$ is also block-diagonal, hence obtaining $(\mathbf{D} - \mathbf{B}_b^T \mathbf{A}_b^{-1} \mathbf{B}_b)^{-1}$, which is the most computationally-intensive operation in (3.28), requires $\mathcal{O}(M_r^3)$ operations. The cost of calculating the remaining matrix multiplications and additions in (3.28) is significantly lower as compared to this cubic cost.

To compute the Jacobian, $\mathbf{g}_{\mathcal{C}_2''}$ [see (3.29)], we can reuse the values of $\mathbf{D}$, $(\mathbf{D} - \mathbf{B}_b^T \mathbf{A}_b^{-1} \mathbf{B}_b)^{-1}$, and $\mathbf{A}_b^{-1}$, which have already been calculated when computing the Hessian. In addition, we need to compute $\mathbf{D}^{-1}$, which can be found with complexity $\mathcal{O}(M_r^3)$. The rest of the computations involve only matrix-vector multiplications and vector additions at a negligible cost.

Hence, the overall cost of the marginalization step is cubic in the number of marginalized non-key poses, and only linear in the number of marginalized non-key landmarks. Since M_r is bounded (user defined), the marginalization in C-KLAM can be carried out

with minimal computational overhead.

3.3 Experimental and Simulation Results

3.3.1 Experimental Results

The experimental setup consists of a PointGrey Chameleon camera and a Navchip IMU, rigidly attached on a light-weight (100 g) platform. The IMU signals were sampled at a frequency of 100 Hz while camera images were acquired at 7.5 Hz. SIFT features [16] were detected in the camera images and matched using a vocabulary tree [84]. The experiment was conducted in an indoor environment where the sensor platform performed a 3D rectangular trajectory, with a total length of 144 m and returned back to the initial position in order to provide an estimate of the final position error.

In the C-KLAM implementation, the corresponding approximate batch MAP optimization problem was solved every 20 incoming camera frames. The exploration epoch was set to 60 camera frames, from which the first and last 10 consecutive camera frames were retained as keyframes, while the rest were marginalized using the C-KLAM algorithm. We compared the performance of C-KLAM to that of the computationally-intensive, batch MAP-based SLAM [bundle adjustment (BA)], which optimizes over all camera poses and landmarks, using all available measurements, to provide high-accuracy estimates. In the BA implementation, the batch MAP optimization problem was solved every 20 incoming camera frames.

Fig. 3.6 shows the $x - y$ view of the estimated trajectory and landmark positions. As evident, the estimates of the robot trajectory and landmark positions generated by C-KLAM are almost identical to those of the BA. Loop closure was performed and the final position error was 7 cm for C-KLAM, only 5% more than that of the BA.

In terms of speed, the C-KLAM algorithm took only 4% of the time required for the entire BA. At the end of this experiment, C-KLAM retained 238 keyframes and 349 key landmarks, while BA had 1038 camera frames and 1281 landmarks. This significant reduction in the number of estimated states in C-KLAM led to substantial improvement in efficiency. Moreover, by using information from non-keyframes to constrain the keyframes, C-KLAM was able to achieve estimation performance comparable to that of the BA.

Another experiment was conducted using the same IMU-camera sensor package mounted on a Parrot AR.Drone quadrotor, flying in an indoor environment with a total trajectory length of 126 m. However, in this experiment, the drone did not return to the exact starting position and there were no loop closures. In the C-KLAM implementation, the resulting optimization problem was solved every 20 incoming camera frames. The exploration epoch was set to 100 camera frames, from which the first and last 20 consecutive camera frames were retained as keyframes, while the rest were marginalized using the C-KLAM algorithm. At the end of the experiment, C-KLAM retained 330 keyframes and 348 key landmarks, compared to 1110 camera poses and 1083 landmarks in BA.

Fig. 3.7 shows the estimated 3D trajectory and landmarks for both BA and C-KLAM. From the figure, we see that, similar to the previous experiment, the estimates of the robot trajectory and landmark positions generated by C-KLAM almost coincide with those generated by the BA, although no loop closure was performed in either C-KLAM or BA. Since the quadrotor did not return to the exact starting position, the final position error cannot be determined for this experiment. However, the difference between the final position estimates of BA and C-KLAM was 0.4% of the length of the total trajectory.

3.3.2 Simulation Results

The performance of C-KLAM was extensively tested in simulations for a variety of conditions. The simulation results corroborate our experimental results, both in terms of the accuracy and speed of C-KLAM. In a particular simulation setup, the IMU-camera platform traversed a helical trajectory of radius 5 m at an average velocity of 0.6 m/s and the camera observed features distributed on the interior wall of a circumscribing cylinder with radius 6 m and height 2 m. The camera had a 90 degrees field of view, with measurement noise standard deviation of 1 pixel, while the IMU was modeled with MEMS quality sensors. The C-KLAM approximate batch MAP optimization problem was solved every 10 incoming camera frames. The exploration epoch was set to 20 camera frames, from which 4 consecutive camera frames were retained as keyframes, while the rest were marginalized using the C-KLAM algorithm. In the BA implementation, the batch MAP optimization problem was solved every 10 camera frames.

Table 3.1: RMSE results for BA and C-KLAM.

	BA	C-KLAM
Robot Orientation (rad)	3.92e-4	5.02e-4
Robot Position (m)	2.24e-2	2.75e-2
Landmark Position (m)	2.78e-2	5.31e-2

Table 3.1 shows the Root Mean Square Error (RMSE) for the platform's position and orientation, and for the landmarks' position (averaged over all key landmarks). From the table, we see that, as expected, the performance of C-KLAM, in terms of accuracy, is comparable to that of the BA.

3.4 Summary

In this chapter, we presented C-KLAM, an approximate MAP estimator-based SLAM algorithm. In order to reduce the computational complexity of batch MAP-based SLAM, C-KLAM estimates only the keyframes and key landmarks, observed from these keyframes. However, instead of discarding the measurement information from non-keyframes and non-key landmarks, C-KLAM uses most of this information to generate pose constraints between the keyframes, resulting in substantial information gain. Moreover, the approximations performed in C-KLAM retain the sparsity of the information matrix, and hence the resulting optimization problem can be solved efficiently. Our results demonstrated that C-KLAM not only obtains substantial speed-up, but also achieves estimation accuracy comparable to that of the batch MAP-based SLAM that uses all available measurement information.

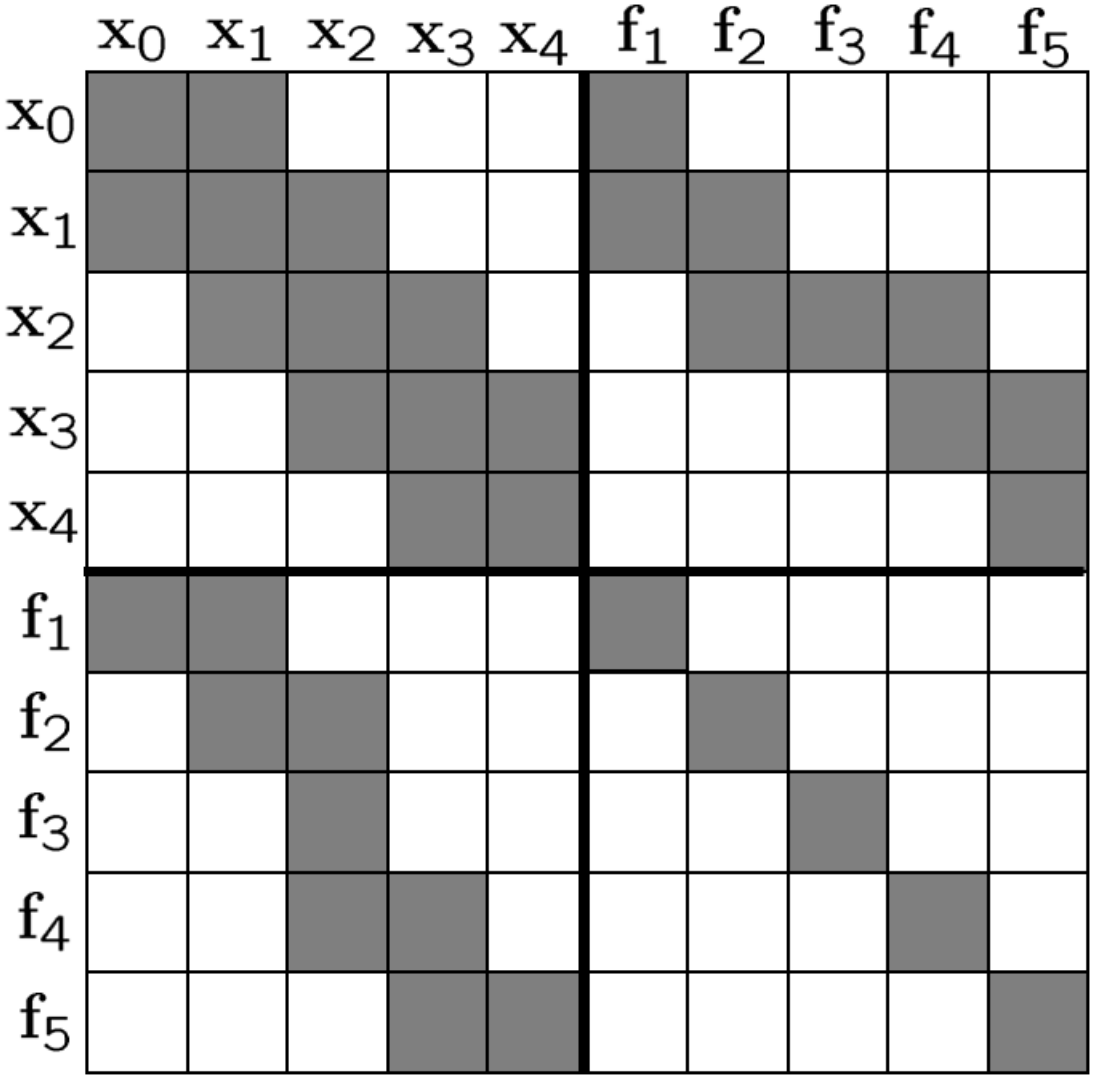


Figure 3.2: Structure of the Hessian matrix, **H**, corresponding to (3.16) for the example depicted in Fig. 3.1(left).

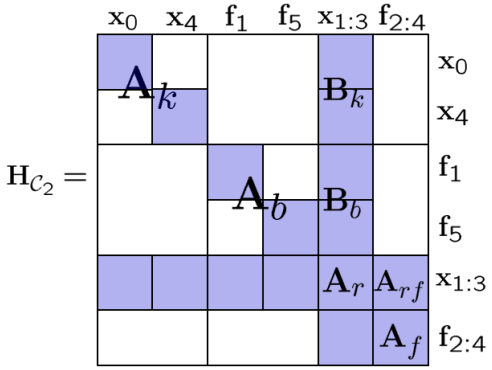


Figure 3.3: Structure of the sparse symmetric positive definite information (Hessian) matrix corresponding to the cost function $\mathcal{C}_2$ in (3.19) (measurements shown with red arrows in Fig. 1). The colored blocks denote non-zero elements. The block-diagonal sub-matrices $\mathbf{A}_k$ and $\mathbf{A}_b$ correspond to key poses and key landmarks, respectively. $\mathbf{A}_r$ and $\mathbf{A}_f$ correspond to non-key poses and non-key landmarks to be marginalized, respectively. Here $\mathbf{A}_k$ and $\mathbf{A}_r$ are, in general, block tri-diagonal, while $\mathbf{A}_b$ and $\mathbf{A}_f$ are block diagonal.

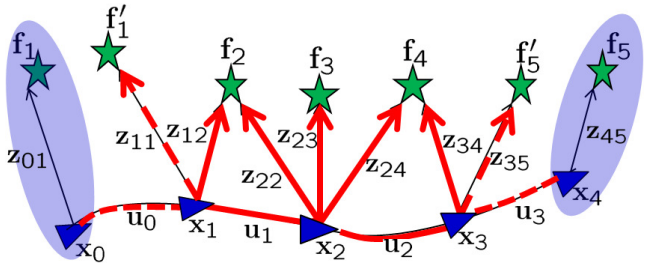


Figure 3.4: Pictorial depiction of the approximation carried out by C-KLAM in order to ensure sparsity of the Hessian matrix. Instead of associating the measurements $\mathbf{z}_{11}$ and $\mathbf{z}_{35}$, to the key features $\mathbf{f}_1$ and $\mathbf{f}_5$ (see Fig. 1), respectively, C-KLAM assumes that these are measurements to different landmarks $\mathbf{f}'_1$ and $\mathbf{f}'_5$.

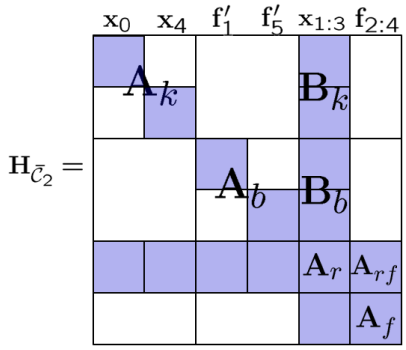


Figure 3.5: Structure of the Hessian matrix, $\mathbf{H}_{\bar{\mathcal{C}}_2}$, corresponding to the cost function $\bar{\mathcal{C}}_2$ [see (3.26)]. The colored blocks denote non-zero elements. Note that this Hessian matrix does not have any entries corresponding to the key features $\mathbf{f}_1$ and $\mathbf{f}_5$. Instead, it has entries for the features $\mathbf{f}_1'$ and $\mathbf{f}_5'$.

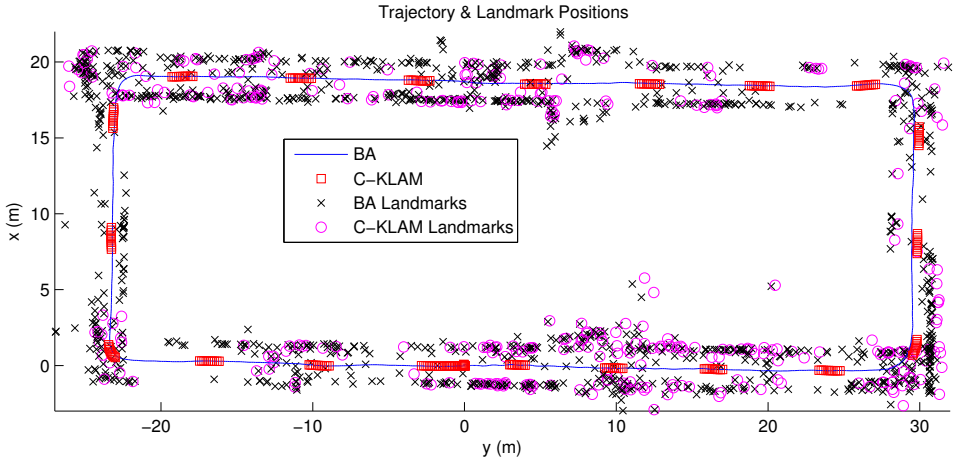


Figure 3.6: Overhead $x-y$ view of the estimated 3D trajectory and landmark positions. The C-KLAM estimates only keyframes (marked with red squares) and key features (marked with magenta circles), while BA estimates the entire trajectory (marked by black line) and all features (marked by black x-s).

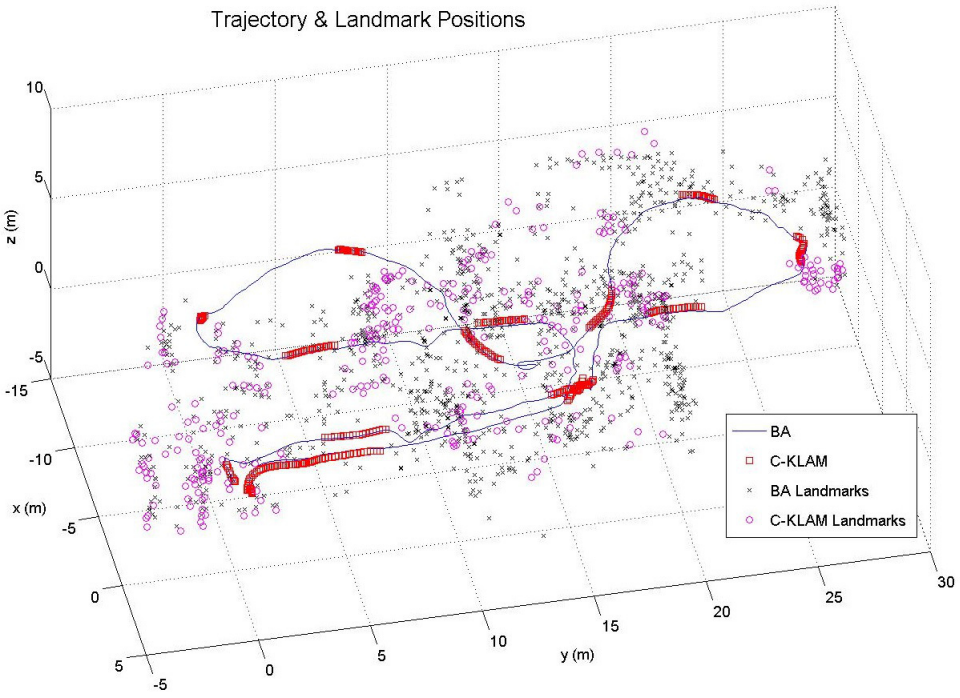


Figure 3.7: 3D view of the estimated trajectory and landmark positions for the AR.Drone experiment. C-KLAM estimates only keyframes (marked with red squares) and key features (marked with magenta circles) while BA estimates the entire trajectory (marked by black line) and all features (marked by x-s).

Chapter 4

Distributed MAP-based CL

In this chapter, we present a distributed Maximum A Posteriori (MAP) estimator for multi-robot Cooperative Localization (CL). In centralized MAP-based CL, a Fusion Center (FC) collects and single-handedly processes measurements from all robots. In contrast, this work presents a systematic data-distribution technique that harnesses the processing and storage resources of all robots in order to substantially speed-up the CL algorithm, without compromising on estimation accuracy. Moreover, for applications with computational constraints, we present a resource-aware extension of this approach that can trade estimation accuracy for processing resources. Parts of this chapter have appeared in [34].

4.1 Introduction

Cooperating teams of robots can be used in a variety of applications such as exploring and mapping extensive terrains in space/underwater [12], search and rescue operations spanning large disaster sites [5], and for mass surveillance [10]. Since these tasks require tight coordination amongst robots, a prerequisite for their successful execution is accurate robot localization, i.e., accurately estimating the position and orientation (pose) of each robot. Moreover, for time-critical operations (e.g., obstacle avoidance or activity recognition) it is necessary for the robot localization algorithm to operate efficiently in the background, with maximum accuracy and minimum resource utilization.

While GPS-based localization ensures bounded uncertainty in the robots' pose estimates, GPS signals are often unreliable (e.g., within urban canyons and under tree foliage) or unavailable (e.g., in space, underwater, and inside caves). In these GPS-denied environments, each robot in the team can localize independently using dead-reckoning. However, the uncertainty in the robots' pose estimates, generated using this technique, increases continuously and finally renders these estimates useless. In such scenarios, an alternative approach for multi-robot applications is Cooperative Localization (CL), where groups of communicating robots use relative measurements (e.g., distance, bearing, and orientation) to *jointly* estimate their poses, resulting in increased accuracy for the entire team.

A variety of estimators such as the Extended Kalman Filter (EKF) [28], Particle Filters (PF) [85], the Maximum Likelihood Estimator (MLE) [32], and the Maximum A Posteriori (MAP) estimator [38] have been used to solve the CL problem. However, *centralized* implementations of these estimators require *all* robots' measurements to be communicated to a Fusion Center (FC) or a leader robot that single-handedly processes these measurements to generate improved estimates. This complete dependence on the FC makes these approaches susceptible to single-point failures. Moreover, the communication and computational requirements, for large teams of robots, can overwhelm the FC and prohibit real-time performance. Furthermore, processing and storage resources are utilized sub-optimally, i.e., while the FC is overburdened with computations and data storage, other robots remain idle. Overcoming these limitations requires distributing the estimation algorithm's computations between the robots. However, existing approaches are either optimal but partially-distributed or suboptimal and fully-distributed. Moreover, approximations evoked in the latter case provide no guarantee for the convergence of the resulting algorithms (see Section 4.2.2).

In this work, we present *fully-distributed*, optimal MAP-based CL that divides the processing for *every step* amongst the robots. In contrast to centralized approaches, it harnesses the computational and storage resources of *all* robots in the team to reduce the resource requirements *per robot*. We choose the MAP estimator because it improves the accuracy of the robots' pose estimates over the *entire* trajectory by acting as a *smoother* and reducing linearization errors. Additionally, the MAP estimator can efficiently process time-delayed measurements, which further increases its estimation

accuracy.

Specifically, we formulate MAP-based CL as a nonlinear least squares (NLS) minimization problem and solve it iteratively using the Levenberg-Marquardt (LM) algorithm (see Section 4.3). We discuss the shortcomings of centralized MAP-based CL and motivate its distributed implementation in Section 4.4. Our proposed distributed algorithm's storage, computation, and communication efficiency stems from: (i) The *distributed data-storage scheme*, which divides data amongst the robots, hence allowing for parallel processing of information locally available to each robot (see Section 4.5.1). (ii) The *Distributed Conjugate Gradient* (DCG) algorithm employed at each iteration of the LM minimization process (to solve a system of linear equations) with cost *at most quadratic* in the number of robots (see Section 4.5.2). (iii) The *distributed marginalization* of past robot poses that limits the size of the optimization problem and whose computational complexity is *quadratic* in the number of robots (see Section 4.5.3). Next, we provide the detailed description and complexity analysis for the proposed distributed algorithm. Finally, we present simulation and experimental results (see Sections 4.6 and 4.7) that compare the performance of the proposed algorithm with the standard EKF-based CL and the centralized MAP-based CL.

4.2 Related Work

Depending on the assumptions made for the existence of a FC, the existing approaches for CL can be classified into the following two main categories:

4.2.1 Centralized Cooperative Localization

Early work on CL considered robots operating as "portable beacons" [86, 87, 27]. The robot team is divided into two sub-teams, one of which is moving while the other remains stationary, acting as beacons. The moving sub-team obtains relative measurements to these beacons and uses a triangulation-based scheme to generate pose estimates for itself. This process is alternated till all robots reach their final destination. Experimental results and improvements to this algorithm are presented in [88, 89, 90].

Compared to the technique discussed above, a more accurate estimate for the robots' poses can be obtained by processing all available measurements in a filtering, smoothing,

or batch estimation framework. To achieve this, an EKF-based algorithm for CL is introduced in [37, 91, 92]. However, for dense measurement graphs, the total number of robot-to-robot measurements, for a team of N robots, can be as high as $O(N(N-1))$ per time step, resulting in processing cost of $O(N^4)$. Recently, [93] showed that by using a modified version of the Householder QR [64], the computational complexity of centralized EKF-based CL can be reduced from $O(N^4)$ to $O(N^3)$ per time step.

Compared to EKF-based CL, approaches based on MAP estimation and MLE improve the accuracy of the robots' pose estimates by providing a batch solution. An MLE-based approach to CL is presented in [32], where the resulting NLS minimization problem is solved using the Conjugate Gradient (CG) algorithm [94] with a cost of $O(K^2N^3)$, where K is the number of time steps considered. Another MLE-based approach has been presented in [87, 95], where the NLS problem is solved using the Broyden-Fletcher-Goldfarb-Shanno (BFGS) [94] optimization algorithm. Similar in spirit is the MAP-based CL algorithm of [38] that minimizes the NLS cost function using LM [96]. The authors employ sparse QR decomposition [97] to solve the resulting system of linear equations at each LM iteration.

A key drawback of all these centralized approaches is their susceptibility to single-point failures, since all computations are carried out by the FC. Moreover, due to their high processing and storage requirements, these methods guarantee real-time performance only for small groups of robots. Lastly, for robot teams navigating within large areas, communication range limitations may prevent robots from exchanging information with the FC.

4.2.2 Decentralized Cooperative Localization

In order to reduce the computational complexity of CL, various optimal[1] and suboptimal decentralized approaches have been proposed. Under optimal approaches, we have algorithms that seek to distribute the processing of CL amongst the team members, in order to reduce the processing requirements per robot. A partially-distributed version of the centralized EKF-based CL in [37], is presented in [28]. This approach distributes the state-covariance propagation by decomposing the centralized EKF into N smaller

[1] Optimal up to linearization errors. These decentralized approaches use all available measurements to generate estimates identical to their centralized counterparts.

communicating EKFs, one on each robot. However, state-covariance updates, for every relative measurement, are centralized and require the broadcast of: (i) the measurement and its associated covariance, and (ii) the state estimates for all robots and the corresponding cross-correlations, to the entire team. Another partially-distributed algorithm for CL, using the information filter [98], is presented in [99]. Here, each robot locally processes its own proprioceptive (odometry) measurements to generate pose estimates for itself. The robot-to-robot relative measurements, however, along with the associated information vector and information matrix, for both the sensing and sensed robots, have to be communicated to the FC for centralized processing. Thus, even though both these approaches generate optimal estimates, they are able to distribute the processing for only a part of the estimation algorithm.

Under suboptimal (approximate) approaches, we have algorithms that either discard a part of the measurement information, or ignore correlations between robots, in order to reduce the computational complexity of CL. An approximate version of the distributed EKF-based CL of [28], is presented in [100], where every robot processes and communicates only the most informative measurements. A similar approach, which selects the subset of measurements that minimizes the trace of the weighted covariance matrix, subject to processing and communication constraints, is proposed in [101]. A multi-centralized algorithm based on Covariance Intersection [102], for consistent fusing of estimates with unknown correlations, is presented in [103, 104].

We now discuss approaches, that in order to reduce the computational complexity of EKF-based CL, ignore some (or even all in the case of [105]) cross-correlations, which may lead to overly optimistic and inconsistent estimates. In this category, we have the Interlaced Kalman filter-based CL [105] and the state-estimates exchange CL [106]. Another algorithm, based on a hierarchy of EKFs, is proposed in [107]. Here, the robot group is divided into sub-teams. The states of the robots in a sub-team are estimated by its leader using an EKF. Depending on the number of leaders, the leaders themselves can also form sub-teams and the same division of processing is repeated in a hierarchical manner ensuring that the size of each sub-team remains bounded.

A decentralized version of the MLE-based CL algorithm of [32] is presented in [108]. In this case, the NLS cost function is divided into N sub-problems, one for each robot. In this approximation, every robot independently minimizes the part of the cost function

that contains terms corresponding to: (i) its proprioceptive measurements, and (ii) the subset of exteroceptive (robot-to-robot relative pose) measurements that involve itself (the robot). During this process, the pose estimates of the other robots are considered constant. All robots periodically broadcast their updated pose estimates and the same process is repeated. A shortcoming of this algorithm is that there exists no proof that it will converge even to a local minimum. Additionally, the authors provide no information about its processing requirements.

A multi-centralized approach for CL was proposed in [43]. Here, the authors assume a dynamically changing communication/measurement graph. Each robot stores a dataset containing its previous state estimates (up to a particular time step) and sensor measurements from that time step onwards. The robots exchange these measurements with other robots within their communication range. Finally, when certain conditions are met, each robot independently computes the pose estimates for the entire team and carries out marginalization of the previous time steps. Note that since this is a multi-centralized approach, the computational and storage complexity for *each* robot is the same as that of the centralized formulation. Thus, this approach does not scale well to large robot teams. An extension of this approach, with adjustable communication bandwidth requirements per link, has been proposed in [109].

4.2.3 Overview of the Proposed Approach

In this work, MAP-based CL is formulated as a NLS minimization problem. The solution is obtained by employing the iterative LM minimization algorithm that *guarantees fast convergence* to at least a local minimum.[2] During each iteration of LM, the resulting system of linear equations is solved *in parallel* by all robots using the DCG algorithm.[3] This in effect, reduces the computational complexity of CL by a factor of N. A key advantage of DCG (iterative) over direct algorithms (e.g., distributed Gauss-Elimination or QR factorization used in [38]) is that it provides an *intermediate solution* at every iteration. Furthermore, contrary to other iterative methods (such as

[2] Note that LM is optimal up to linearization errors. However, as is the case for all nonlinear minimization algorithms, convergence to the global minimum is guaranteed only when the initial estimate is within the region of attraction of the optimum point.

[3] Note that in contrast to [32], where the centralized CG is used as an iterative nonlinear minimization algorithm, we employ the LM algorithm for minimizing the NLS cost and use the DCG for solving the system of linear equations arising at each iteration of the LM algorithm.

the Jacobi algorithm [110]) that converge only asymptotically, DCG converges within a *bounded number of iterations*. These important features of DCG allow the robots to trade processing for accuracy when computing resources are scarce (e.g., during time-critical tasks). This approach to CL along with the *distributed marginalization* of past robot poses, enables the team to perform real-time CL using limited computation and communication resources.

4.3 Problem Formulation

4.3.1 Problem Setup

Consider a team of N communicating robots navigating in 2D while performing CL. The state vector $\mathbf{x}_k = [\mathbf{x}_k^{1^T}, \mathbf{x}_k^{2^T}, \ldots, \mathbf{x}_k^{N^T}]^T$, where $\mathbf{x}_k^i = [x_k^i, y_k^i, \phi_k^i]^T$, $i = 1, \ldots, N$, contains the position and orientation of all robots at time-step k. The continuous-time dynamics for robot i are given by the following motion model:

$$\dot{\mathbf{x}}^i(t) = \mathbf{f}(\mathbf{x}^i(t), \mathbf{u}^i(t)), \quad \mathbf{u}^i(t) = \mathbf{u}_m^i(t) + \mathbf{w}_c^i(t) \tag{4.1}$$

where $\mathbf{f}$ is a general nonlinear function. Each robot is equipped with proprioceptive (odometry) sensors that provide linear, $v_m^i(t)$, and rotational, $\omega_m^i(t)$, velocity measurements. Here, $\mathbf{u}^i(t) = [v^i(t), \omega^i(t)]^T$, denotes robot i's true linear and rotational velocity respectively, $\mathbf{u}_m^i(t)$ denotes their measured values, and $\mathbf{w}_c^i(t)$ is the continuous-time noise in these measurements, assumed to be zero-mean, white, Gaussian with autocorrelation $E[\mathbf{w}_c^i(t)\mathbf{w}_c^i(\tau)^T] = \mathbf{Q}_c^i\delta(t-\tau)$. The first-order discretized model with time-step δt is given by:

$$\mathbf{x}_{k+1}^i = \mathbf{x}_k^i + \mathbf{f}(\mathbf{x}_k^i, \mathbf{u}_{m_k}^i)\delta t + \mathbf{w}_k^i \tag{4.2}$$

where $\mathbf{w}_k^i$ is the discrete-time noise with covariance $E[\mathbf{w}_k^i \mathbf{w}_k^{i^T}] = \mathbf{Q}_k^i$ and

$$\mathbf{Q}_k^i = \int_{t_k}^{t_{k+1}} \mathbf{\Phi}^i(t_{k+1}, \tau)\mathbf{G}_c^i(\tau)\mathbf{Q}_c^i\mathbf{G}_c^i(\tau)^T\mathbf{\Phi}^i(t_{k+1}, \tau)^T d\tau \tag{4.3}$$

with $\mathbf{\Phi}^i(t_{k+1}, t_k) = \exp\left(\int_{t_k}^{t_{k+1}} \mathbf{F}_c^i(\tau)d\tau\right)$, where

$$\mathbf{F}_c^i = \nabla_{\mathbf{x}^i(t)}\mathbf{f}(\mathbf{x}^i(t), \mathbf{u}^i(t))|_{\mathbf{x}^i(t)=\hat{\mathbf{x}}^i(t), \mathbf{u}^i(t)=\mathbf{u}_m^i(t)}$$
$$\mathbf{G}_c^i = \nabla_{\mathbf{u}^i(t)}\mathbf{f}(\mathbf{x}^i(t), \mathbf{u}^i(t))|_{\mathbf{x}^i(t)=\hat{\mathbf{x}}^i(t), \mathbf{u}^i(t)=\mathbf{u}_m^i(t)}$$

Additionally, all robots have exteroceptive sensors that allow them to uniquely identify other robots in the team and measure their relative distance and bearing. The discrete-time measurement model for robot i measuring robot j is:

$$\mathbf{z}_k^{i,j} = \mathbf{h}(\mathbf{x}_k^i, \mathbf{x}_k^j) + \mathbf{n}_k^{i,j} \tag{4.4}$$

with $\mathbf{h} = [d_k^{i,j}, \theta_k^{i,j}]^T$, where $d_k^{i,j}$ and $\theta_k^{i,j}$ are the true distance and bearing respectively, from robot i to robot j at time-step k and $\mathbf{n}_k^{i,j} = [n_{d_k}^{i,j}, n_{\theta_k}^{i,j}]^T$ is additive, zero-mean, white Gaussian measurement noise with covariance $\mathbf{R}_k^{i,j}$. Furthermore, we assume a fully-connected communication graph, i.e., each robot can communicate with all other robots in the team.

4.3.2 Maximum A Posteriori Estimator Formulation for CL

Our objective is to compute the MAP estimates of the robots' poses, $\mathbf{x}_{0:K-1}$, from time-step 0 up to time-step $K-1$ given exteroceptive measurements, $\mathbf{z}_{0:K-1}$, and proprioceptive measurements, $\mathbf{u}_{0:K-2}$. The MAP estimator is formulated as:

$$\hat{\mathbf{x}}_{0:K-1} = \arg\max p(\mathbf{x}_{0:K-1}|\mathbf{z}_{0:K-1}, \mathbf{u}_{0:K-2}) \tag{4.5}$$

$$= \arg\max \frac{1}{p(\mathbf{z}_{0:K-1})} p(\mathbf{z}_{0:K-1}|\mathbf{x}_{0:K-1}) p(\mathbf{x}_{0:K-1}|\mathbf{u}_{0:K-2}) \tag{4.6}$$

$$= \arg\max \frac{1}{p(\mathbf{z}_{0:K-1})} \prod_{k=0}^{K-1} p(\mathbf{z}_k|\mathbf{x}_k) \prod_{k=0}^{K-2} p(\mathbf{x}_{k+1}|\mathbf{x}_k, \mathbf{u}_k) \cdot p(\mathbf{x}_0) \tag{4.7}$$

$$= \arg\min \Bigg(- \sum_{k=0}^{K-1} \log p(\mathbf{z}_k|\mathbf{x}_k) - \sum_{k=0}^{K-2} \log p(\mathbf{x}_{k+1}|\mathbf{x}_k, \mathbf{u}_k) - \log p(\mathbf{x}_0) \Bigg) \tag{4.8}$$

where $p(\mathbf{x}_0)$ is the prior on the robots' initial poses. Note that since $\frac{1}{p(\mathbf{z}_{0:K-1})}$ is a normalizing constant, it can be neglected during optimization. Using Bayes' rule, we obtain (4.6) from (4.5). Due to the independence assumption of the noises in the measurement and motion models, and employing the Markov assumption (a robot's current pose depends only on its previous pose and the motion between them), (4.6) simplifies into (4.7). Further, based on the monotonicity of the logarithmic function, we obtain (4.8). Finally, due to the Gaussian-noise assumption, (4.8) simplifies into the following NLS problem:

$$\hat{\mathbf{x}}_{0:K-1} = \arg\min\Bigg(\sum_{i=1}^{N}\sum_{k=0}^{K-2} ||\mathbf{x}_{k+1}^{i} - \mathbf{x}_{k}^{i} - \mathbf{f}(\mathbf{x}_{k}^{i}, \mathbf{u}_{m_k}^{i})\delta t||_{\mathbf{Q}_k^i}^2 + \sum_{i=1}^{N}\sum_{\substack{j=1 \\ j\neq i}}^{N}\sum_{k=0}^{K-1} ||\mathbf{z}_{k}^{i,j} - \mathbf{h}(\mathbf{x}_{k}^{i}, \mathbf{x}_{k}^{j})||_{\mathbf{R}_k^{i,j}}^2 + \sum_{i=1}^{N} ||\mathbf{x}_0^i - \mathbf{x}_{init}^i||_{\mathbf{P}_0^i}^2\Bigg) \tag{4.9}$$

where $\mathbf{x}_{init}^i$ and $\mathbf{P}_0^i$ are the mean and covariance respectively of the prior for the pose of robot i and $||\mathbf{e}||_{\mathbf{W}}^2 = \mathbf{e}^T\mathbf{W}^{-1}\mathbf{e}$ is the weighted squared L_2-norm for a given covariance, $\mathbf{W}$.

Since the motion [see (4.2)] and the measurement [see (4.4)] models are non-linear, the minimization problem in (4.9) is solved by iteratively linearizing about the latest estimates for the robots' poses. For the motion model [see (4.2)], by linearizing about the current estimates for the robots' poses, $\hat{\mathbf{x}}_{k+1}^i$ and $\hat{\mathbf{x}}_k^i$, we obtain:

$$\hat{\mathbf{x}}_{k+1}^i + \delta\mathbf{x}_{k+1}^i \simeq \hat{\mathbf{x}}_k^i + \delta\mathbf{x}_k^i + \mathbf{f}(\hat{\mathbf{x}}_k^i, \mathbf{u}_{m_k}^i)\delta t + \mathbf{F}_k^i\delta t\delta\mathbf{x}_k^i + \mathbf{w}_k^i \tag{4.10}$$

where $\delta\mathbf{x}_{k+1}^i$ and $\delta\mathbf{x}_k^i$ are the errors in the estimates $\hat{\mathbf{x}}_{k+1}^i$ and $\hat{\mathbf{x}}_k^i$ respectively, i.e., $\mathbf{x}_k^i = \hat{\mathbf{x}}_k^i + \delta\mathbf{x}_k^i$, where $\mathbf{x}_k^i$ is the true pose of robot i at time-step k. The Jacobian $\mathbf{F}_k^i$ is calculated as follows:

$$\mathbf{F}_k^i = \nabla_{\mathbf{x}_k^i}\mathbf{f}(\mathbf{x}_k^i, \mathbf{u}_{m_k}^i)|_{\mathbf{x}_k^i=\hat{\mathbf{x}}_k^i} \tag{4.11}$$

Similarly for the measurement model [see (4.4)], by linearizing about the current estimates for the robots' poses $\hat{\mathbf{x}}_k^i$ and $\hat{\mathbf{x}}_k^j$, we obtain:

$$\mathbf{z}_k^{i,j} \simeq \mathbf{h}(\hat{\mathbf{x}}_k^i, \hat{\mathbf{x}}_k^j) + {}^i\mathbf{H}_k^{i,j}\delta\mathbf{x}_k^i + {}^j\mathbf{H}_k^{i,j}\delta\mathbf{x}_k^j + \mathbf{n}_k^{i,j} \tag{4.12}$$

where the Jacobians ${}^i\mathbf{H}_k^{i,j}$ and ${}^j\mathbf{H}_k^{i,j}$ are calculated as:

$${}^i\mathbf{H}_k^{i,j} = \nabla_{\mathbf{x}_k^i}\mathbf{h}(\mathbf{x}_k^i, \mathbf{x}_k^j)|_{\mathbf{x}_k^i=\hat{\mathbf{x}}_k^i, \mathbf{x}_k^j=\hat{\mathbf{x}}_k^j} \tag{4.13}$$

$${}^j\mathbf{H}_k^{i,j} = \nabla_{\mathbf{x}_k^j}\mathbf{h}(\mathbf{x}_k^i, \mathbf{x}_k^j)|_{\mathbf{x}_k^i=\hat{\mathbf{x}}_k^i, \mathbf{x}_k^j=\hat{\mathbf{x}}_k^j} \tag{4.14}$$

The uncertainty in the initial pose of the i^{th} robot is modeled as zero-mean Gaussian with covariance $\mathbf{P}_0^i$ as follows:

$$\begin{aligned} ||\mathbf{x}_0^i - \mathbf{x}_{init}^i||_{\mathbf{P}_0^i}^2 &= ||\hat{\mathbf{x}}_0^i + \delta\mathbf{x}_0^i - \mathbf{x}_{init}^i||_{\mathbf{P}_0^i}^2 \\ &= ||\delta\mathbf{x}_0^i + (\hat{\mathbf{x}}_0^i - \mathbf{x}_{init}^i)||_{\mathbf{P}_0^i}^2 \end{aligned} \tag{4.15}$$

Now, we can transform the weighted squared L_2-norm into the regular squared L_2-norm as:

$$||\mathbf{e}||^2_{\mathbf{W}} = \mathbf{e}^T\mathbf{W}^{-1}\mathbf{e} = ||\mathbf{W}^{-1/2}\mathbf{e}||^2_2 \tag{4.16}$$

From (4.9), (4.10), (4.12), (4.15), (4.16) and using the following notations for the prewhitened Jacobians:

$$\overline{\mathbf{F}}^i_k = \left(\mathbf{Q}^i_k\right)^{-1/2}\left(\mathbf{I} + \mathbf{F}^i_k\delta t\right) \tag{4.17}$$

$${}^i\overline{\mathbf{H}}^{i,j}_k = (\mathbf{R}^{i,j}_k)^{-1/2}({}^i\mathbf{H}^{i,j}_k) \tag{4.18}$$

each iteration of the non-linear minimization problem has the form:

$$\begin{aligned}\delta\mathbf{x}^* = \arg\min \Bigg(& \sum_{i=1}^{N}\sum_{k=0}^{K-2}\Big\{||\left(\mathbf{Q}^i_k\right)^{-1/2}\delta\mathbf{x}^i_{k+1} - \overline{\mathbf{F}}^i_k\delta\mathbf{x}^i_k \\ & \qquad + \left(\mathbf{Q}^i_k\right)^{-1/2}(\hat{\mathbf{x}}^i_{k+1} - \hat{\mathbf{x}}^i_k - \mathbf{f}(\hat{\mathbf{x}}^i_k, \mathbf{u}^i_{m_k})\delta t)||^2_2\Big\} \\ & + \sum_{i=1}^{N}\sum_{\substack{j=1\\ j\neq i}}^{N}\sum_{k=0}^{K-1}\Big\{|| - {}^i\overline{\mathbf{H}}^{i,j}_k\delta\mathbf{x}^i_k - {}^j\overline{\mathbf{H}}^{i,j}_k\delta\mathbf{x}^j_k \\ & \qquad + (\mathbf{R}^{i,j}_k)^{-1/2}(\mathbf{z}^{i,j}_k - \mathbf{h}(\hat{\mathbf{x}}^i_k, \hat{\mathbf{x}}^j_k))||^2_2\Big\} \\ & + \sum_{i=1}^{N}||\left(\mathbf{P}^i_0\right)^{-1/2}\delta\mathbf{x}^i_0 + \left(\mathbf{P}^i_0\right)^{-1/2}(\hat{\mathbf{x}}^i_0 - \mathbf{x}^i_{init})||^2_2\Bigg)\end{aligned} \tag{4.19}$$

By stacking the different terms from (4.19) in a matrix $\mathbf{A}$ and a vector $\mathbf{b}$, the p^{th} iteration of the iterative minimization process is represented as:

$$\begin{aligned}\delta\mathbf{x}^* &= \arg\min ||\mathbf{A}(\hat{\mathbf{x}}^{(p)}_{0:K-1})\delta\mathbf{x} - \mathbf{b}(\hat{\mathbf{x}}^{(p)}_{0:K-1})||^2_2 \\ \hat{\mathbf{x}}^{(p+1)}_{0:K-1} &= \hat{\mathbf{x}}^{(p)}_{0:K-1} + \delta\mathbf{x}^*\end{aligned} \tag{4.20}$$

where $\mathbf{A}$ and $\mathbf{b}$ depend on the current iterate $\hat{\mathbf{x}}^{(p)}_{0:K-1}$ and $\delta\mathbf{x} = [(\delta\mathbf{x}_0)^T, \ldots, (\delta\mathbf{x}_{K-1})^T]^T$, where $\delta\mathbf{x}_k = [(\delta\mathbf{x}^1_k)^T, \ldots, (\delta\mathbf{x}^N_k)^T]^T$, for $k = 0, \ldots, K-1$. An example of the structure of $\mathbf{A}$ and $\mathbf{b}$ when $K = N = 3$ is shown in (4.25) and (4.26).

4.3.3 Structure of the Minimization Problem

Typically, algorithms such as the LM (see Alg. 2, [96]), that combine the Gauss-Newton and the Gradient Descent methods, are used to solve the minimization problem in (4.20). Each iteration of the LM algorithm solves modified normal equations of the form (see

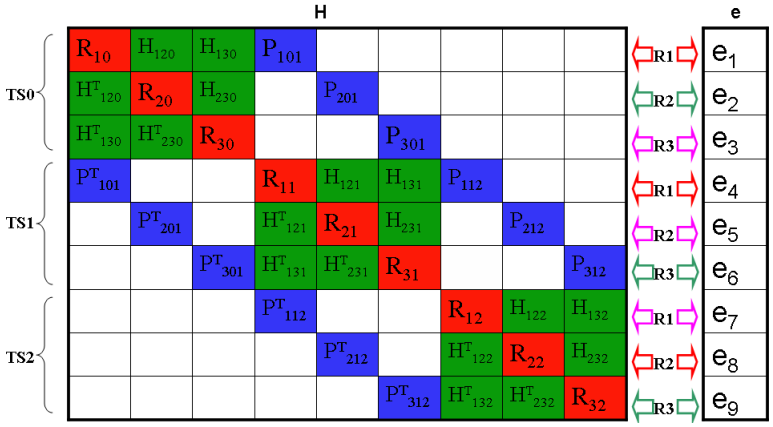


Figure 4.1: Distribution of matrix $\mathbf{H}$ and vector $\mathbf{e}$ amongst the robots. TS: time step.

Step 3 of Alg. 2):

$$(\mathbf{H} + \lambda \mathbf{I})\delta\mathbf{x} = \mathbf{e} \tag{4.21}$$

where $\mathbf{I}$ is the identity matrix, λ is a scalar parameter, $\mathbf{H} = \mathbf{A}^T\mathbf{A}$ and $\mathbf{e} = \mathbf{A}^T\mathbf{b}$. Since the computational and communication complexity analyses presented in this work primarily depend on the dimensions and structure of $\mathbf{H}$, the rest of this section presents some of its important properties. Fig. 4.1 depicts the structure of $\mathbf{H}$ over three time steps[4] for a team of three robots with a dense measurement graph, i.e, when every robot measures the relative position of all other robots at every time step. The dimensions of the symmetric positive definite matrix $\mathbf{H}$ are $KN \times KN$, where K is the number of time steps considered and N is the number of robots. This block-tridiagonal matrix has $N + 2$ non-zero elements in each row/column. These elements of $\mathbf{H}$ represent three information sources:

1. The diagonal elements, $\mathbf{R}_{ik}$ (red), correspond to the information about the pose of robot i at time-step k.

2. The off-diagonal elements, $\mathbf{H}_{ijk}$, correspond to the information from the relative position measurements between robot i and robot j at time-step k.

[4] In order to simplify the presentation of the computational and communication complexity, we will consider the 1D case where each block of $\mathbf{H}$ (see Fig. 4.1) reduces to a scalar. Note that the ensuing analysis also holds for the case of robots navigating in 2D and 3D.

Algorithm 2 LM Algorithm

Require: Initial guess $\hat{\mathbf{x}}^{(p)}_{0:K-1}$

Ensure: $\chi^2(\hat{\mathbf{x}}^{(p)}_{0:K-1} + \delta\mathbf{x}) - \chi^2(\hat{\mathbf{x}}^{(p)}_{0:K-1}) \leq 0.01\chi^2(\hat{\mathbf{x}}^{(p)}_{0:K-1})$ {Stopping Criterion}

1: Compute $\chi^2(\hat{\mathbf{x}}^{(p)}_{0:K-1}) = (\mathbf{A}^T\mathbf{b})^T\mathbf{A}^T\mathbf{b}$
2: Initialize $\lambda \leftarrow 0.001$ {Typical}
3: Solve $(\mathbf{A}^T\mathbf{A} + \lambda\mathbf{I})\delta\mathbf{x} = \mathbf{A}^T\mathbf{b}$ and Evaluate $\chi^2(\hat{\mathbf{x}}^{(p)}_{0:K-1} + \delta\mathbf{x})$
4: **if** $\chi^2(\hat{\mathbf{x}}^{(p)}_{0:K-1} + \delta\mathbf{x}) \geq \chi^2(\hat{\mathbf{x}}^{(p)}_{0:K-1})$ **then**
5: $\quad\lambda \leftarrow \lambda \times 10$, Goto 3
6: **end if**
7: **if** $\chi^2(\hat{\mathbf{x}}^{(p)}_{0:K-1} + \delta\mathbf{x}) < \chi^2(\hat{\mathbf{x}}^{(p)}_{0:K-1})$ **then**
8: $\quad\lambda \leftarrow \lambda/10$, $\hat{\mathbf{x}}^{(p)}_{0:K-1} \leftarrow \hat{\mathbf{x}}^{(p)}_{0:K-1} + \delta\mathbf{x}$, Goto 3
9: **end if**
10: **return** $\hat{\mathbf{x}}_{0:K-1} \leftarrow \hat{\mathbf{x}}^{(p)}_{0:K-1}$

3. The off-diagonal elements, $\mathbf{P}_{ik\ell}$ (blue), correspond to the information from the motion of robot i from time-step k to time-step $\ell = k + 1$.

4.4 Centralized Cooperative Localization

In the centralized implementation of CL, all robots in the team periodically send their proprioceptive and exteroceptive measurements to a leader robot, or a FC, that solves (4.19) and provides updated estimates for the robots' poses. We hereafter discuss the drawbacks of a centralized approach to the minimization problem:

1. Since all computations are carried out by the FC, this approach is susceptible to failures of the FC.

2. There is significant loss, in terms of efficiency and speed, since the centralized algorithm does not utilize all available computation and storage resources, i.e., while the FC is burdened with computations, the other robots in the team remain idle after communicating their measurements.

3. At each LM iteration, the FC must solve the modified normal equations [see (4.21)]. Since $\mathbf{H}$ and $\mathbf{e}$ have dimensions, $KN \times KN$ and $KN \times 1$ respectively, where K is the number of time steps considered, this process requires $O(KN^3)$ operations [64,

Section 4.3]. Since the cost of solving these normal equations is cubic in N and linear in K, for large robot teams deployed over long periods, real-time performance cannot be achieved.

4. Moreover, once $\delta\mathbf{x}$ is computed, the FC must calculate the new estimates ($\hat{\mathbf{x}}^{(p)}_{0:K-1} + \delta\mathbf{x}$) and update $\mathbf{H}$ and $\mathbf{e}$ single-handedly. As K grows, the FC will have increasing difficulty not only in generating real-time solutions but also in handling the memory requirements for evaluating and storing $\mathbf{H}$, $\mathbf{e}$, and other intermediate results.

In the next section, we address these limitations of the centralized approach with a distributed algorithm that leverages the memory and processing resources of all the robots in order to reduce the computational and storage complexity of CL.

4.5 Distributed Cooperative Localization

4.5.1 Distributed Data Storage and Updating

In contrast to the centralized formulation that requires communication of all measurements to the FC, in the proposed algorithm, each robot i constructs and updates rows/columns[5] $i,\ i+N,\ \ldots,\ i+(K-1)N$ of $\mathbf{H}$ and the corresponding elements of $\mathbf{e}$. As an example, in Fig. 4.1, robot 1 is responsible for rows/columns 1, 4, and 7, robot 2 for rows/columns 2, 5, and 8, and robot 3 for rows/columns 3, 6, and 9 of $\mathbf{H}$, and each robot is also responsible for the corresponding elements of $\mathbf{e}$.

Consider the fifth row of $\mathbf{H}$ stored by robot 2 which contains the following three types of terms: (i) Off-diagonal terms (green), H_{231} and H_{121}, involving relative position measurement Jacobians evaluated at the robot pose estimates ($\hat{x}^2_1,\ \hat{x}^3_1$) and ($\hat{x}^1_1,\ \hat{x}^2_1$), respectively. (ii) Off-diagonal terms (blue), P_{201} and P_{212}, involving motion model Jacobians between time-steps 0 and 1, and time-steps 1 and 2, evaluated at $\hat{x}^2_0$ and $\hat{x}^2_1$, respectively. (iii) Diagonal term (red), R_{21}, which contains Jacobians from the robot-to-robot measurements involving robot 2 at time-step 1 and from the motion model of robot 2 between: (a) time-steps 0 and 1, and (b) time-steps 1 and 2. Additionally,

[5] We would like to remind the reader that we are presenting the complexity analysis for the 1D scenario.

computing the fifth element, e_5, of $\mathbf{e}$ requires estimates, $\hat{x}_0^2$, $\hat{x}_1^2$ and $\hat{x}_2^2$ of robot 2, estimates $\hat{x}_1^j$ of robot j ($j = 1, 3$) and the measurements, $z_1^{2,j}$ and $z_1^{j,2}$, between robots 2 and j.

Amongst all the relative position measurements and robots' pose estimates appearing above, robot 2's own pose estimates at time-steps 0, 1 and 2, i.e., $\hat{x}_0^2$, $\hat{x}_1^2$ and $\hat{x}_2^2$ respectively, and relative position measurements, $z_1^{2,j}$, at time-step 1 to robot j, are locally available to robot 2. The remaining quantities, i.e., measurements, $z_1^{j,2}$, and estimates, $\hat{x}_1^j$, of robot j, are necessary in order to construct the fifth row of $\mathbf{H}$ and e_5. These quantities can be easily obtained if at time-step 1, when robots 2 and j observe each other, robot j communicates[6] its measurement and current state estimate to robot 2. By imposing a rule that these 2 quantities are communicated by robot i to robot j whenever robot i obtains a relative position measurement of robot j, each robot can construct its assigned rows of $\mathbf{H}$ and elements of $\mathbf{e}$ with minimal communication overhead of $O(N)$ per robot, where $i, j \in \{1, \ldots, N\}$ and $j \neq i$.

A key advantage of this distributed storage scheme is that whenever a new state estimate becomes available, the elements of the Hessian $\mathbf{H}$ and the residual $\mathbf{e}$ can be updated in parallel by the corresponding robots, hence reducing the time required for updating $\mathbf{H}$ and $\mathbf{e}$ by a factor of N.

Based on this distributed storage scheme, in the next section, we present the distributed conjugate gradient (DCG) algorithm for computing the updated state estimates during each iteration of the LM algorithm.

4.5.2 Distributed Conjugate Gradient

As stated in Section 4.4, each LM iteration solves a system of modified normal equations [see (4.21)]. Two types of algorithms are available for solving this system of equations: *direct* and *iterative* [110]. Direct algorithms have computational complexity $O(KN^3)$ for banded systems [64] and include methods such as Gauss-Elimination and its variants, Odd-Even Reduction, and Givens Rotations. Even though the computational complexity of distributed implementations of these algorithms is $O(KN^2)$ [110], their practical implementations suffer from several disadvantages. The distributed implementation of Odd-Even Reduction requires the inversion of $N \times N$ matrices at each time step, making

[6] Here we assume that the communication radius of the robots is greater than their sensing radius.

it numerically unstable, while distributed Givens Rotations incurs excessive communication overhead.

Moreover, direct algorithms provide *no intermediate solution.* This is a major drawback especially for robots communicating via wireless connections susceptible to intermittent failures. If inter-robot communication is interrupted before the direct algorithm has completed all its steps, the robots will not obtain a new updated solution for their pose estimates. Thus, they will have to revert to their previous estimates after having wasted valuable computational and communication resources. If the communication link fails often, the resulting solution will suffer from the same drawbacks as that of dead-reckoning.

In contrast, iterative algorithms, also referred to as *any-time* algorithms, generate an approximate solution at every iteration with increasing accuracy [110]. However, most of the commonly used iterative algorithms such as Jacobi, Gauss-Seidel, Jacobi overrelaxation, and Successive overrelaxation, converge only asymptotically (i.e., after infinite number of steps) [110]. Alternatively, the Conjugate Gradient (CG) algorithm is guaranteed to converge in at most KN iterations. Moreover, and for the special class of large systems of equations considered here, where $\mathbf{H}$ is a symmetric positive definite $KN \times KN$ matrix, the CG yields sufficiently accurate solutions with significantly fewer iterations [110].

We now analyze the computational and communication complexity of the CG algorithm for two types of measurement graphs:

1. **Type 1:** *Complete* graph, i.e., when each robot observes all other robots at every time step, leading to a total of $N(N-1)$ relative position measurements per time step.

2. **Type 2:** *General* graph with αN relative position measurements per time step, where $\alpha \in \{(n+2)/N : n = 0, 2, 4, \ldots, N(N-1)-2\}$.[7]

[7] This analysis assumes bi-directionality of relative position measurements, i.e., when robot i measures robot j, robot j also obtains a measurement to robot i. This is necessary to ensure a one-to-one relation between the number of non-zero elements in $\mathbf{H}$ and the number of relative position measurements.

Next we compare and contrast the centralized CG (CCG), (single processor implementation) with the distributed CG (DCG) (multi-processor implementation). Tables 4.1, 4.2 list the steps involved during each iteration of CG along with their computational and communication complexity.

Table 4.1: Computational Complexity Analysis of the Conjugate Gradient Method (one iteration).

		Computation			
Algorithm		Centralized		Distributed	
Number of Measurements		$N(N-1)$	αN	$N(N-1)$	αN
Step 1	$\mathbf{g}_m = \mathbf{H}\delta\mathbf{x}_m - \mathbf{e}$	$O(KN^2)$	$O(\alpha KN)$	$O(KN)$	$O(\alpha K)$
Step 2	$\beta_m = {\mathbf{g}_m}^T\mathbf{g}_m / {\mathbf{g}_{m-1}}^T\mathbf{g}_{m-1}$	$O(KN)$	$O(KN)$	$O(K + log(N))$	$O(K + log(N))$
Step 3	$\mathbf{s}_m = -\mathbf{g}_m + \beta_m\mathbf{s}_{m-1}$	$O(KN)$	$O(KN)$	$O(K)$	$O(K)$
Step 4a	${\mathbf{s}_m}^T\mathbf{g}_m$	$O(KN)$	$O(KN)$	$O(K + log(N))$	$O(K + log(N))$
Step 4b	$\mathbf{h}_m = \mathbf{H}\mathbf{s}_m$	$O(KN^2)$	$O(\alpha KN)$	$O(KN)$	$O(\alpha K)$
Step 4c	$\gamma_m = -{\mathbf{s}_m}^T\mathbf{g}_m / {\mathbf{s}_m}^T\mathbf{h}_m$	$O(KN)$	$O(KN)$	$O(K + log(N))$	$O(K + log(N))$
Step 5	$\delta\mathbf{x}_{m+1} = \delta\mathbf{x}_m + \gamma_m\mathbf{s}_m$	$O(KN)$	$O(KN)$	$O(KN)$	$O(KN)$

Table 4.2: Communication Complexity Analysis of the Conjugate Gradient Method (one iteration)

		Communication	
Algorithm		Distributed	
Number of Measurements		$N(N-1)$	αN
Step 1	$\mathbf{g}_m = \mathbf{H}\delta\mathbf{x}_m - \mathbf{e}$	0	0
Step 2	$\beta_m = {\mathbf{g}_m}^T\mathbf{g}_m / {\mathbf{g}_{m-1}}^T\mathbf{g}_{m-1}$	$O(1)$	$O(1)$
Step 3	$\mathbf{s}_m = -\mathbf{g}_m + \beta_m\mathbf{s}_{m-1}$	0	0
Step 4a	${\mathbf{s}_m}^T\mathbf{g}_m$	$O(1)$	$O(1)$
Step 4b	$\mathbf{h}_m = \mathbf{H}\mathbf{s}_m$	$O(K)$	$O(K)$
Step 4c	$\gamma_m = -{\mathbf{s}_m}^T\mathbf{g}_m / {\mathbf{s}_m}^T\mathbf{h}_m$	$O(1)$	$O(1)$
Step 5	$\delta\mathbf{x}_{m+1} = \delta\mathbf{x}_m + \gamma_m\mathbf{s}_m$	0	0

Specifically, each iteration m, where $m \in \{0, \ldots, KN-1\}$, consists of the following steps:

Step 1: $\mathbf{g}_m = \mathbf{H}\delta\mathbf{x}_m - \mathbf{e}$

Here $\mathbf{g}_m = [g_m(1), \ldots, g_m(KN)]^T$ is a $KN \times 1$ vector. All robots initialize $\delta\mathbf{x}_0$ to a vector of zeros.

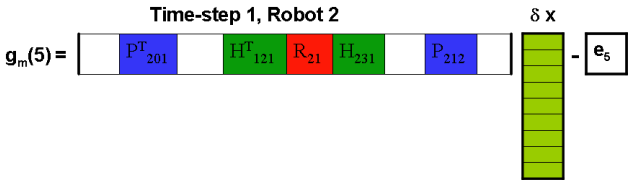


Figure 4.2: Example of Robot 2 calculating element $g_m(5)$ of $\mathbf{g}_m$.

Type 1: Due to the special block-tridiagonal structure of $\mathbf{H}$, each row of $\mathbf{H}$ contains $N+2$ non-zero elements (see Fig. 4.1). Hence computing each element $g_m(j)$ of $\mathbf{g}_m$, where $j = 1, \ldots, KN$, requires $O(N)$ operations.

1. **CCG**: Calculating KN elements of $\mathbf{g}_m$ requires $O(KN^2)$ operations.

2. **DCG**: Given the distribution of rows of $\mathbf{H}$ and $\mathbf{e}$ amongst the robots (see Section 4.5.1), robot i calculates $g_m(j)$, where $j \in S_i = \{i,\ i+N,\ i+2N, \ldots,\ i+(K-1)N\}$, locally, i.e., each robot calculates K terms of $\mathbf{g}_m$, requiring $O(KN)$ operations per robot (see Fig. 4.2). Continuing with the example of three robots over three time steps (see Fig. 4.1), robot 1 computes $g_m(1)$, $g_m(4)$, and $g_m(7)$, robot 2 computes $g_m(2)$, $g_m(5)$, and $g_m(8)$ and robot 3 computes $g_m(3)$, $g_m(6)$, and $g_m(9)$.

Type 2: The rows of $\mathbf{H}$ corresponding to time-step i for all N robots (e.g., rows 4, 5, and 6 in Fig. 4.1 corresponding to time-step 1), have αN non-zero elements (green) corresponding to αN relative position measurements, N non-zero elements (red) corresponding to N robots' poses, and $2N$ non-zero elements (blue) resulting from the correlations introduced due to propagation (odometry measurements). Thus, calculating the elements of $\mathbf{g}_m$ corresponding to time-step i (e.g., $g_m(4)$, $g_m(5)$, and $g_m(6)$ at time-step 1) requires $O(\alpha N)$ operations.

1. **CCG**: For K time steps, the computational complexity is $O(\alpha KN)$.

2. **DCG**: When distributed amongst N robots, the computational complexity, on the *average*, becomes $O(\alpha K)$ per robot.

For this step, the communication cost is zero, as all computations are carried out locally by the robots.

Step 2: $\beta_m = {\mathbf{g}_m}^T \mathbf{g}_m / {\mathbf{g}_{m-1}}^T \mathbf{g}_{m-1}$

For initialization, $\beta_0 = 0$. Since $\mathbf{g}_m$ is generally a dense $KN \times 1$ vector, the analysis for this step is independent of the type of measurement graph.

1. **CCG**: The cost for calculating the inner-product, ${\mathbf{g}_m}^T\mathbf{g}_m$, and hence for computing β_m is $O(KN)$.

2. **DCG**: Each robot i calculates the dot-product $b_i = \sum_{j \in S_i} g_m(j)^2$, $i = 1, \ldots, N$, of its K local elements of $\mathbf{g}_m$ at a cost of $O(K)$. For example (see Fig. 4.1), robot 1 calculates $b_1 = g_m(1)^2 + g_m(4)^2 + g_m(7)^2$, robot 2 calculates $b_2 = g_m(2)^2 + g_m(5)^2 + g_m(8)^2$, and robot 3 calculates $b_3 = g_m(3)^2 + g_m(6)^2 + g_m(9)^2$. As is well known, adding these N scalars (i.e., b_i's) in a distributed way requires $O(\log(N))$ steps[8] incurring a communication cost of $O(1)$ per robot [110, Section 1.2.3]. At the end of this process, one of the robots acquires the final result for ${\mathbf{g}_m}^T\mathbf{g}_m$ and calculates β_m using the value of ${\mathbf{g}_{m-1}}^T\mathbf{g}_{m-1}$ from the previous iteration, at a cost of $O(1)$. Once β_m is available, it is broadcast to all other robots at a communication cost of $O(1)$. Thus, for this step, the computational and communication cost per robot is $O(K + \log(N))$ and $O(1)$, respectively.

Step 3: $\mathbf{s}_m = -\mathbf{g}_m + \beta_m \mathbf{s}_{m-1}$

For $m = 0$, $\mathbf{s}_0$ is initialized to $-\mathbf{g}_0$. This step incurs no communication overhead since all computations are local. Also, the analysis is independent of the type of measurement graph.

1. **CCG**: Since both $\mathbf{g}_m$ and $\mathbf{s}_{m-1}$ are vectors of dimension $KN \times 1$ and β_m is a scalar, the computational complexity for calculating $\mathbf{s}_m$ is $O(KN)$.

2. **DCG**: Due to the distribution of $\mathbf{g}_m$ (see Step 1), robot i calculates $s_m(j)$, where $j \in S_i$, locally, i.e., each robot evaluates K terms of $\mathbf{s}_m$, which requires $O(K)$ operations per robot. Therefore, in our example, robot 1 calculates $s_m(1)$, $s_m(4)$, and $s_m(7)$, robot 2 calculates $s_m(2)$, $s_m(5)$, and $s_m(8)$ and robot 3 calculates $s_m(3)$, $s_m(6)$, and $s_m(9)$.

Step 4: $\gamma_m = -{\mathbf{s}_m}^T\mathbf{g}_m / {\mathbf{s}_m}^T\mathbf{H}\mathbf{s}_m$

We analyze the cost of calculating ${\mathbf{s}_m}^T\mathbf{g}_m$ and $d_m = {\mathbf{s}_m}^T\mathbf{H}\mathbf{s}_m$ separately.

[8] Note that the computational cost per robot is constant, $O(1)$. However, we are primarily interested in the time required for performing these additions and thus we adopt $O(\log(N))$ as the computational cost of these operations.

- **Calculate** ${\mathbf{s}_m}^T\mathbf{g}_m$: Since both $\mathbf{s}_m$ and $\mathbf{g}_m$ are generally dense vectors, this analysis is independent of the type of the measurement graph.

  1. **CCG**: The computational complexity is $O(KN)$.
  2. **DCG**: As in Step 2, robot i has K elements each of $s_m(j)$ and $g_m(j)$, $j \in S_i$, locally available. The partial dot-product $p_i = \sum_{j \in S_i} s_m(j)g_m(j)$ computed by robot i requires $O(K)$ operations, resulting into N scalars. For example, robot 1 calculates $p_1 = s_m(1)g_m(1) + s_m(4)g_m(4) + s_m(7)g_m(7)$, robot 2 calculates $p_2 = s_m(2)g_m(2)+s_m(5)g_m(5)+s_m(8)g_m(8)$, and robot 3 calculates $p_3 = s_m(3)g_m(3) + s_m(6)g_m(6) + s_m(9)g_m(9)$. The rest of the analysis is identical to Step 2.

- **Calculate** d_m:

  Type 1:

  1. **CCG**: Similar to Step 1, for the complete measurement graph, calculating $\mathbf{h}_m = \mathbf{H}\mathbf{s}_m$ requires $O(KN^2)$ operations, while computing ${\mathbf{s}_m}^T\mathbf{h}_m$ has cost $O(KN)$.
  2. **DCG**: For calculating $\mathbf{h}_m$, all robots must acquire the *entire* $\mathbf{s}_m$ vector. Thus, each robot must broadcast its K elements of $\mathbf{s}_m$. Hence the total communication cost is $O(KN)$, or $O(K)$ per robot. Once all the robots obtain $\mathbf{s}_m$, each robot i calculates $h_m(j)$, $j \in S_i$, i.e., robot 1 calculates $h_m(1)$, $h_m(4)$ and $h_m(7)$, robot 2 calculates $h_m(2)$, $h_m(5)$ and $h_m(8)$ and robot 3 calculates $h_m(3)$, $h_m(6)$ and $h_m(9)$, locally with cost $O(KN)$ per robot. Computation of the dot-product, ${\mathbf{s}_m}^T\mathbf{h}_m$, is similar to Step 2, and has computational and communication cost of $O(K + \log(N))$ and $O(1)$, respectively. Once d_m and ${\mathbf{s}_m}^T\mathbf{g}_m$ are available, γ_m is calculated for the computational cost of $O(1)$ and broadcasted.

  Type 2:

  1. **CCG**: Similar to Step 1, for the general measurement graph, calculating $\mathbf{h}_m$ requires $O(\alpha KN)$ operations while computing ${\mathbf{s}_m}^T\mathbf{h}_m$ has cost $O(KN)$.

2. **DCG**: For this step, the complexity analysis is identical to that of the complete measurement graph, except that once all robots obtain $\mathbf{s}_m$, each robot i calculates $h_m(j)$, $j \in S_i$, locally with cost $O(\alpha K)$ per robot (see Step 1).

Step 5: $\delta\mathbf{x}_{m+1} = \delta\mathbf{x}_m + \gamma_m \mathbf{s}_m$

Since γ_m, $\mathbf{s}_m$ and $\delta\mathbf{x}_m$ are locally available, each robot calculates $\delta\mathbf{x}_{m+1}$ at the computational cost of $O(KN)$.

Steps 1 to 5 are repeated until convergence, i.e., $\mathbf{g}_m = \mathbf{0}$. For each iteration of the CCG, the computational complexity is $O(KN^2)$, while for the DCG it is $O(KN)$. Theoretically, the CG requires KN such iterations for convergence. Therefore, the complexity of CCG is $O(K^2N^3)$, while that of DCG is $O(K^2N^2)$. Thus, DCG successfully reduces the computational complexity by a factor of N. Moreover, since $\mathbf{H}$ and $\mathbf{e}$ are stored in a distributed fashion, the time required for updating them is also reduced by a factor of N. Additionally, since processing is distributed, the system is more robust to failures. If robot i fails, the team simply discards the rows/columns of $\mathbf{H}$ and elements of $\mathbf{e}$ corresponding to robot i and carries out CL on the remaining data.

Given the solution $\delta\mathbf{x}$ from the DCG, the robots compute new estimates for $\mathbf{x}$ [see (4.20)] and update $\mathbf{H}$ and $\mathbf{e}$. This constitutes a single iteration of the LM algorithm (see Step 3 of Alg. 2). Since *no* approximations have been introduced in the distributed algorithm, the performance of DCG is *identical* to that of centralized MAP-based CL, but with reduced computational cost per robot.

A limitation of MAP-based CL is that as the number of time steps, K, increases, so do the computation, communication, and storage requirements. Typically, this problem is addressed by marginalizing past robot poses and maintaining a constant-length time window. In the next section, we demonstrate how marginalization is efficiently carried out within our distributed framework at reduced computational cost.

4.5.3 Marginalization

In this section, we first discuss marginalization in the context of CL and then present its distributed implementation along with its complexity analysis.

Set-up for Marginalization

Let us assume that depending on the computational and communication resources available to the robots, we restrict the size of the minimization problem to J, i.e., the dimensions of $\mathbf{H}$ are restricted to $J \times J$. Therefore, when $KN = J$, we need to marginalize the robots' poses from the earliest time step in order to reduce the size of the problem to $(K-1)N$. This ensures that there will be sufficient resources for processing measurements corresponding to the next time step.

Consider our example of three robots ($i, j = 1, 2, 3$) over three time steps ($k = 0, 1, 2$) with a complete measurement graph and let $J = 9$. The corresponding cost function [see (4.19)], linearized about the latest available estimates at the time of marginalization, $\hat{\mathbf{x}}_{0M}$, $\hat{\mathbf{x}}_{1M}$ and $\hat{\mathbf{x}}_{2M}$ is:[9]

$$\begin{aligned}
\eta = & \sum_{i=1}^{3}\sum_{k=0}^{1}\big\{||(\mathbf{Q}_k^i)^{-1/2}\delta\mathbf{x}_{k+1}^i - \overline{\mathbf{F}}_k^i\delta\mathbf{x}_k^i \\
& \qquad + (\mathbf{Q}_k^i)^{-1/2}(\hat{\mathbf{x}}_{(k+1)\,M}^i - \hat{\mathbf{x}}_{k\,M}^i - \mathbf{f}(\hat{\mathbf{x}}_{k\,M}^i, \mathbf{u}_{m_k}^i)\delta t)||_2^2\big\} \\
& + \sum_{i=1}^{3}\sum_{\substack{j=1\\ j\neq i}}^{3}\sum_{k=0}^{2}\big\{||-{}^i\overline{\mathbf{H}}_k^{i,j}\delta\mathbf{x}_k^i - {}^j\overline{\mathbf{H}}_k^{i,j}\delta\mathbf{x}_k^j \\
& \qquad + (\mathbf{R}_k^{i,j})^{-1/2}(\mathbf{z}_k^{i,j} - \mathbf{h}(\hat{\mathbf{x}}_{k\,M}^i, \hat{\mathbf{x}}_{k\,M}^j))||_2^2\big\} \\
& + \sum_{i=1}^{3}||(\mathbf{P}_0^i)^{-1/2}\delta\mathbf{x}_0^i + (\mathbf{P}_0^i)^{-1/2}(\hat{\mathbf{x}}_{0\,M}^i - \mathbf{x}_{init}^i)||_2^2
\end{aligned} \tag{4.22}$$

For simplicity, the cost function in (4.22) can be re-written in a compact form as:

$$\begin{aligned}
\eta = & \sum_{k=0}^{1}\big\{||(\mathbf{Q}_k)^{-1/2}\delta\mathbf{x}_{k+1} - \overline{\mathbf{F}}_k\delta\mathbf{x}_k \\
& \qquad + (\mathbf{Q}_k)^{-1/2}(\hat{\mathbf{x}}_{(k+1)\,M} - \hat{\mathbf{x}}_{k\,M} - \mathbf{f}(\hat{\mathbf{x}}_{k\,M}, \mathbf{u}_{m_k})\delta t)||_2^2\big\} \\
& + \sum_{k=0}^{2}||-\overline{\mathbf{H}}_k\delta\mathbf{x}_k + (\mathbf{R}_k)^{-1/2}(\mathbf{z}_k - \mathbf{h}(\hat{\mathbf{x}}_{k\,M}))||_2^2 \\
& + ||(\mathbf{P}_0)^{-1/2}\delta\mathbf{x}_0 + (\mathbf{P}_0)^{-1/2}(\hat{\mathbf{x}}_{0\,M} - \mathbf{x}_{init})||_2^2 \qquad (4.23) \\
= & \; ||\mathbf{A}\delta\mathbf{x} - \mathbf{b}||_2^2 \qquad (4.24)
\end{aligned}$$

[9] For generality, we maintain the vector notation in this section. But note that we are still analyzing the 1D scenario.

where

$$\begin{aligned}
\delta\mathbf{x}_k &= [(\delta\mathbf{x}_k^1)^T, (\delta\mathbf{x}_k^2)^T, (\delta\mathbf{x}_k^3)^T]^T \\
\hat{\mathbf{x}}_{kM} &= [(\hat{\mathbf{x}}_{kM}^1)^T, (\hat{\mathbf{x}}_{kM}^2)^T, (\hat{\mathbf{x}}_{kM}^3)^T]^T \\
\mathbf{u}_{m_k} &= [(\mathbf{u}_{m_k}^1)^T, (\mathbf{u}_{m_k}^2)^T, (\mathbf{u}_{m_k}^3)^T]^T \\
\mathbf{z}_k &= [(\mathbf{z}_k^{1,2})^T, (\mathbf{z}_k^{2,1})^T, (\mathbf{z}_k^{1,3})^T, (\mathbf{z}_k^{3,1})^T, (\mathbf{z}_k^{2,3})^T, (\mathbf{z}_k^{3,2})^T]^T \\
\mathbf{x}_{init} &= [(\mathbf{x}_{init}^1)^T, (\mathbf{x}_{init}^2)^T, (\mathbf{x}_{init}^3)^T]^T
\end{aligned}$$

and the corresponding explicit expressions for $\mathbf{Q}_k$, $\overline{\mathbf{F}}_k$, $\mathbf{R}_k$, $\mathbf{P}_0$, and $\overline{\mathbf{H}}_k$ can be found from (4.22). Also, the structure of $\mathbf{A}$ and $\mathbf{b}$, when linearized about the estimates $\hat{\mathbf{x}}_0 = \hat{\mathbf{x}}_{0M}$, $\hat{\mathbf{x}}_1 = \hat{\mathbf{x}}_{1M}$, and $\hat{\mathbf{x}}_2 = \hat{\mathbf{x}}_{2M}$, is shown in detail in (4.25) and (4.26).

Before proceeding with the marginalization process, we examine the structure and the elements of the Hessian matrix $\mathbf{H} = \mathbf{A}^T\mathbf{A}$ and the residual vector $\mathbf{e} = \mathbf{A}^T\mathbf{b}$, in the normal equations corresponding to (4.23). The Hessian matrix $\mathbf{H}$ and vector $\mathbf{e}$ can be split into sub-matrices and sub-vectors as shown in Fig. 4.3.

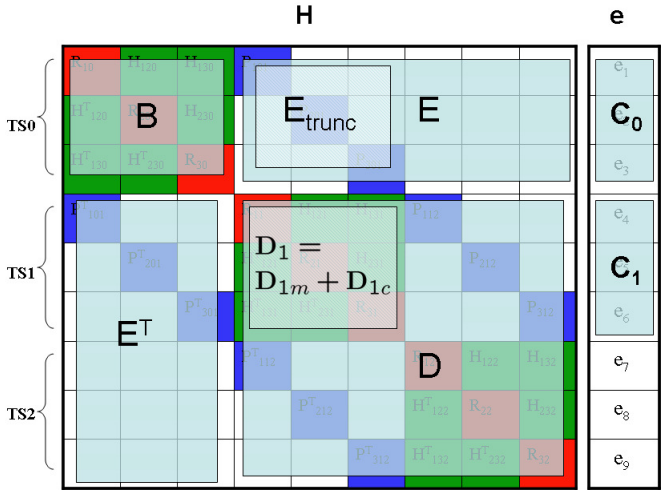


Figure 4.3: $\mathbf{H}$ and $\mathbf{e}$ before marginalization.

Here $\mathbf{B}$ and $\mathbf{E}$ are the $N \times N$ diagonal block and the $N \times (K-1)N$ off-diagonal block of $\mathbf{H}$, respectively, that depend on $\mathbf{x}_0$, i.e., contain Jacobians evaluated at $\hat{\mathbf{x}}_{0M}$.

Note that $\mathbf{E}$ is a sparse matrix with only a $N \times N$ non-zero diagonal sub-matrix denoted as $\mathbf{E}_{trunc}$. Also, $\mathbf{c}_0$ is a $N \times 1$ sub-vector of $\mathbf{e}$ that depends on $\mathbf{x}_0$ and $\mathbf{x}_1$. Specifically, $\mathbf{B}$, $\mathbf{E}_{trunc}$, and $\mathbf{c}_0$ are expressed as [see Figs. 4.1, 4.3,and equations (4.25) and (4.26)]:

$$\mathbf{B} = \overline{\mathbf{F}}_0^T \overline{\mathbf{F}}_0 + \overline{\mathbf{H}}_0^T \overline{\mathbf{H}}_0 + (\mathbf{P}_0)^{-1} \tag{4.27}$$

$$\mathbf{E}_{trunc} = -\overline{\mathbf{F}}_0^T (\mathbf{Q}_0)^{-1/2} \tag{4.28}$$

$$\begin{aligned} \mathbf{c}_0 = {} & \overline{\mathbf{F}}_0{}^T (\mathbf{Q}_0)^{-1/2} (\hat{\mathbf{x}}_{1M} - \hat{\mathbf{x}}_{0M} - \mathbf{f}(\hat{\mathbf{x}}_{0M}, \mathbf{u}_{m_0})\delta t) \\ & + \overline{\mathbf{H}}_0^T (\mathbf{R}_0)^{-1/2} (\mathbf{z}_0 - \mathbf{h}(\hat{\mathbf{x}}_{0M})) \\ & - (\mathbf{P}_0)^{-1} (\hat{\mathbf{x}}_{0M} - \mathbf{x}_{init}) \end{aligned} \tag{4.29}$$

From the $(K-1)N \times (K-1)N$ sub-matrix $\mathbf{D}$ of $\mathbf{H}$, let us specifically concentrate on its $N \times N$ block $\mathbf{D}_1$ (see Fig. 4.3). Note that $\mathbf{D}_1 = \mathbf{D}_{1M} + \mathbf{D}_{1C}$, where $\mathbf{D}_{1M} = \overline{\mathbf{F}}_1{}^T \overline{\mathbf{F}}_1 + \overline{\mathbf{H}}_1{}^T \overline{\mathbf{H}}_1$, and $\mathbf{D}_{1C} = \mathbf{Q}_0^{-1}$, is expressed as the sum of two components: $\mathbf{D}_{1M}$ that depends on $\mathbf{x}_1$, and $\mathbf{D}_{1C}$, a *diagonal* matrix, which depends only on $\mathbf{x}_0$. Similarly, $\mathbf{c}_1 = \mathbf{c}_{1M} + \mathbf{c}_{1C}$, is a $N \times 1$ sub-vector of $\mathbf{e}$ (see Fig. 4.3), where $\mathbf{c}_{1M}$ depends on $\mathbf{x}_1$ and $\mathbf{x}_2$ and $\mathbf{c}_{1C}$ depends on $\mathbf{x}_0$ and $\mathbf{x}_1$[10] , i.e.,

$$\begin{aligned} \mathbf{c}_{1M} = {} & \overline{\mathbf{H}}_1^T (\mathbf{R}_1)^{-1/2} (\mathbf{z}_1 - \mathbf{h}(\hat{\mathbf{x}}_{1M})) \\ & + \overline{\mathbf{F}}_1^T (\mathbf{Q}_1)^{-1/2} (\hat{\mathbf{x}}_{2M} - \hat{\mathbf{x}}_{1M} - \mathbf{f}(\hat{\mathbf{x}}_{1M}, \mathbf{u}_{m_1})\delta t) \end{aligned} \tag{4.30}$$

$$\mathbf{c}_{1C} = -\mathbf{Q}_0^{-1} (\hat{\mathbf{x}}_{1M} - \hat{\mathbf{x}}_{0M} - \mathbf{f}(\hat{\mathbf{x}}_{0M}, \mathbf{u}_{m_0})\delta t) \tag{4.31}$$

Having studied the structure of $\mathbf{H}$ and $\mathbf{e}$, we present the marginalization of past robots' poses in the next subsection.

Marginalization of Past Robot Poses

When measurements from the next time step, i.e., time-step 3, become available, we need to marginalize the robot poses from time-step 0. Once this marginalization step is complete, J will be of dimension 6 and we will be able to process the measurements from time-step 3.

[10] While the structures of the Hessian $\mathbf{H}$ and the vector $\mathbf{e}$ have been described for a 1D scenario, they can be easily generalized to 2D. In 2D, each element of $\mathbf{H}$, which is a scalar in 1D, will correspond to a 3×3 matrix and each element of $\mathbf{e}$ will correspond to a 3×1 vector. For example, in 2D, the dimensions of $\mathbf{B}$, $\mathbf{c}_0$ will be $3N \times 3N$, $3N \times 1$ respectively and the $N \times N$ diagonal matrix $\mathbf{D}_{1C}$ will map to a $3N \times 3N$ block-diagonal matrix with the dimension of each diagonal block being 3×3.

Marginalizing the robots' poses from time-step 0 requires fixing value of the estimate for $\mathbf{x}_0$ in (4.22), treating it as a constant, and not estimating it in the future. The value of $\hat{\mathbf{x}}_0$ is determined by differentiating the original nonlinear cost function about $\mathbf{x}_0$ [see (4.23)] and setting the derivative equal to zero. Fixing the value of $\hat{\mathbf{x}}_0$ ensures that the linearization point of $\mathbf{x}_0$ remains constant. In this example, this linearization point is $\hat{\mathbf{x}}_{0M}$ since it is the latest available estimate for $\mathbf{x}_0$ at the time of marginalization. For the corresponding linearized system, calculating the partial derivative of (4.23) w.r.t $\delta\mathbf{x}_0$ gives:

$$\begin{aligned}\frac{\partial \eta}{\partial \delta \mathbf{x}_0} = &-2\overline{\mathbf{F}}_0{}^T\big((\mathbf{Q}_0)^{-1/2}\delta\mathbf{x}_1 - \overline{\mathbf{F}}_0\delta\mathbf{x}_0 \\ &\qquad + (\mathbf{Q}_0)^{-1/2}(\hat{\mathbf{x}}_{1M} - \hat{\mathbf{x}}_{0M} - \mathbf{f}(\hat{\mathbf{x}}_{0M}, \mathbf{u}_{m_0})\delta t)\big) \\ &- 2\overline{\mathbf{H}}_0^T\big(- \overline{\mathbf{H}}_0\delta\mathbf{x}_0 + (\mathbf{R}_0)^{-1/2}(\mathbf{z}_0 - \mathbf{h}(\hat{\mathbf{x}}_{0M}))\big) \\ &+ 2(\mathbf{P}_0)^{-T/2}\big((\mathbf{P}_0)^{-1/2}\delta\mathbf{x}_0 + (\mathbf{P}_0)^{-1/2}(\hat{\mathbf{x}}_{0M} - \mathbf{x}_{init})\big)\end{aligned} \tag{4.32}$$

Simplifying further and equating (4.32) to zero we obtain:

$$\begin{aligned}&\big(\overline{\mathbf{F}}_0{}^T\overline{\mathbf{F}}_0 + \overline{\mathbf{H}}_0{}^T\overline{\mathbf{H}}_0 + (\mathbf{P}_0)^{-1}\big)\delta\mathbf{x}_0 - \big(\overline{\mathbf{F}}_0{}^T(\mathbf{Q}_0)^{-1/2}\big)\delta\mathbf{x}_1 \\ &- \overline{\mathbf{F}}_0{}^T(\mathbf{Q}_0)^{-1/2}(\hat{\mathbf{x}}_{1M} - \hat{\mathbf{x}}_{0M} - \mathbf{f}(\hat{\mathbf{x}}_{0M}, \mathbf{u}_{m_0})\delta t) \\ &- \overline{\mathbf{H}}_0{}^T(\mathbf{R}_0)^{-1/2}(\mathbf{z}_0 - \mathbf{h}(\hat{\mathbf{x}}_{0M})) + (\mathbf{P}_0)^{-1}(\hat{\mathbf{x}}_{0M} - \mathbf{x}_{init}) = \mathbf{0}\end{aligned} \tag{4.33}$$

From (4.27), (4.28), and (4.29) we see that (4.33) corresponds to:

$$\delta\mathbf{x}_0 = \mathbf{B}^{-1}\mathbf{c}_0 - \mathbf{B}^{-1}\mathbf{E}_{trunc}\delta\mathbf{x}_1 \tag{4.34}$$

Substituting the value of $\delta\mathbf{x}_0$ from (4.34) into the linearized cost function (4.23) yields the new marginalized cost function as follows:

$$\begin{aligned}\eta_{margin} = &\big\{||(\mathbf{Q}_0)^{-1/2}\delta\mathbf{x}_1 - \overline{\mathbf{F}}_0(\mathbf{B}^{-1}\mathbf{c}_0 - \mathbf{B}^{-1}\mathbf{E}_{trunc}\delta\mathbf{x}_1) \\ &\qquad + (\mathbf{Q}_0)^{-1/2}(\hat{\mathbf{x}}_{1M} - \hat{\mathbf{x}}_{0M} - \mathbf{f}(\hat{\mathbf{x}}_{0M}, \mathbf{u}_{m_0})\delta t)||_2^2\big\} \\ &+ \big\{||(\mathbf{Q}_1)^{-1/2}\delta\mathbf{x}_2 - \overline{\mathbf{F}}_1\delta\mathbf{x}_1 \\ &\qquad + (\mathbf{Q}_1)^{-1/2}(\hat{\mathbf{x}}_{2M} - \hat{\mathbf{x}}_{1M} - \mathbf{f}(\hat{\mathbf{x}}_{1M}, \mathbf{u}_{m_1})\delta t)||_2^2\big\} \\ &+ ||-\overline{\mathbf{H}}_0(\mathbf{B}^{-1}\mathbf{c}_0 - \mathbf{B}^{-1}\mathbf{E}_{trunc}\delta\mathbf{x}_1) + (\mathbf{R}_0)^{-1/2}(\mathbf{z}_0 - \mathbf{h}(\hat{\mathbf{x}}_{0M}))||_2^2 \\ &+ \sum_{k=1}^{2}||-\overline{\mathbf{H}}_k\delta\mathbf{x}_k + (\mathbf{R}_k)^{-1/2}(\mathbf{z}_k - \mathbf{h}(\hat{\mathbf{x}}_{kM}))||_2^2 \\ &+ ||(\mathbf{P}_0)^{-1/2}(\mathbf{B}^{-1}\mathbf{c}_0 - \mathbf{B}^{-1}\mathbf{E}_{trunc}\delta\mathbf{x}_1) + (\mathbf{P}_0)^{-1/2}(\hat{\mathbf{x}}_{0M} - \mathbf{x}_{init})||_2^2\end{aligned} \tag{4.35}$$

Due to this substitution step, the new marginalized cost function no longer contains terms involving $\delta\mathbf{x}_0$, i.e., it is independent of $\delta\mathbf{x}_0$. Therefore, the estimate of $\mathbf{x}_0$ will not be updated in the future and will remain fixed at $\hat{\mathbf{x}}_{0M}$. Furthermore, due to marginalization of robots' poses at time-step 0, $J = 6$ and measurements from time-step 3 can now be processed. Rearranging the terms in the above equation and including the measurements at time-step 3 we obtain:

$$\begin{aligned}
\eta_{new} &= \big\{||\big((\mathbf{Q}_0)^{-1/2} + \overline{\mathbf{F}}_0\mathbf{B}^{-1}\mathbf{E}_{trunc}\big)\delta\mathbf{x}_1 - \overline{\mathbf{F}}_0\mathbf{B}^{-1}\mathbf{c}_0 \\
&\qquad + (\mathbf{Q}_0)^{-1/2}(\hat{\mathbf{x}}_1 - \hat{\mathbf{x}}_{0M} - \mathbf{f}(\hat{\mathbf{x}}_{0M}, \mathbf{u}_{m_0})\delta t)||_2^2\big\} \\
&+ \sum_{k=1}^{2}\big\{||(\mathbf{Q}_k)^{-1/2}\delta\mathbf{x}_{k+1} - \overline{\mathbf{F}}_k\delta\mathbf{x}_k \\
&\qquad + (\mathbf{Q}_k)^{-1/2}(\hat{\mathbf{x}}_{k+1} - \hat{\mathbf{x}}_k - \mathbf{f}(\hat{\mathbf{x}}_k, \mathbf{u}_{m_k})\delta t)||_2^2\big\} \\
&+ ||\overline{\mathbf{H}}_0\mathbf{B}^{-1}\mathbf{E}_{trunc}\delta\mathbf{x}_1 + (\mathbf{R}_0)^{-1/2}(\mathbf{z}_0 - \mathbf{h}(\hat{\mathbf{x}}_{0M})) - \overline{\mathbf{H}}_0\mathbf{B}^{-1}\mathbf{c}_0||_2^2 \\
&+ \sum_{k=1}^{3}|| - \overline{\mathbf{H}}_k\delta\mathbf{x}_k + (\mathbf{R}_k)^{-1/2}(\mathbf{z}_k - \mathbf{h}(\hat{\mathbf{x}}_k))||_2^2 \\
&+ \big\{|| - (\mathbf{P}_0)^{-1/2}\mathbf{B}^{-1}\mathbf{E}_{trunc}\delta\mathbf{x}_1 + (\mathbf{P}_0)^{-1/2}(\hat{\mathbf{x}}_{0M} - \mathbf{x}_{init}) \\
&\qquad + (\mathbf{P}_0)^{-1/2}\mathbf{B}^{-1}\mathbf{c}_0||_2^2\big\} \qquad (4.36) \\
&= ||\mathbf{A}_{new}\delta\mathbf{x} - \mathbf{b}_{new}||_2^2.
\end{aligned}$$

In (4.36) it is important to note that as the robots move around in the environment, new measurements corresponding to new time steps will be added to the cost function and new estimates will be generated for $\mathbf{x}_i$, where $i \geq 1$. Due to marginalization, $\mathbf{x}_0$ will remain fixed at $\hat{\mathbf{x}}_{0M}$, but $\hat{\mathbf{x}}_i \neq \hat{\mathbf{x}}_{iM}$ where $\hat{\mathbf{x}}_i$ is the latest estimate for $\mathbf{x}_i$. Therefore, the first term in (4.36) can be re-written as:

$$\begin{aligned}
&(\mathbf{Q}_0)^{-1/2}(\hat{\mathbf{x}}_1 - \hat{\mathbf{x}}_{0M} - \mathbf{f}(\hat{\mathbf{x}}_{0M}, \mathbf{u}_{m_0})\delta t) \\
&= (\mathbf{Q}_0)^{-1/2}(\hat{\mathbf{x}}_{1M} - \hat{\mathbf{x}}_{0M} - \mathbf{f}(\hat{\mathbf{x}}_{0M}, \mathbf{u}_{m_0})\delta t) + (\mathbf{Q}_0)^{-1/2}(\hat{\mathbf{x}}_1 - \hat{\mathbf{x}}_{1M}) \qquad (4.37)
\end{aligned}$$

The details of the structure of $\mathbf{A}_{new}$ and $\mathbf{b}_{new}$ are shown in (4.38) and (4.39).

The system of normal equations [see (4.21)] corresponding to the marginalized cost function is $(\mathbf{H}_{new} + \lambda\mathbf{I})\ [\delta\mathbf{x}_1{}^T\ \delta\mathbf{x}_2{}^T\ \delta\mathbf{x}_3{}^T]^T = \mathbf{e}_{new}$. The structure of $\mathbf{H}_{new} = \mathbf{A}_{new}^T\mathbf{A}_{new}$ and $\mathbf{e}_{new} = \mathbf{A}_{new}^T\mathbf{b}_{new}$ is shown in Figs. 4.4 and 4.5 respectively.

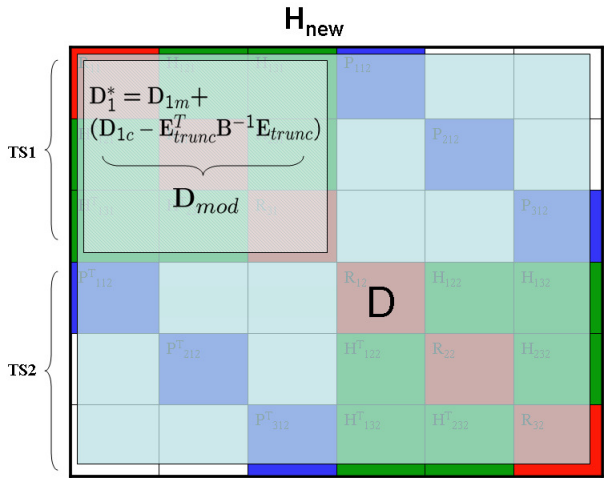


Figure 4.4: $\mathbf{H}_{new}$ after marginalization.

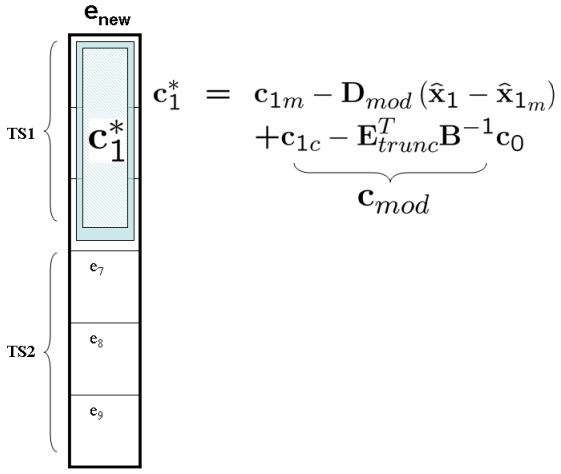


Figure 4.5: $\mathbf{e}_{new}$ after marginalization.

Note that as a result of the marginalization, the correlation between robot poses $\mathbf{x}_0$ and $\mathbf{x}_1$ (due to propagation) introduces additional terms in the quantities $\mathbf{D}_1$ and $\mathbf{c}_1$. We denote the modified quantities as $\mathbf{D}_1^*$ and $\mathbf{c}_1^*$ respectively, where:

$$\begin{aligned} \mathbf{D}_1^* &= \mathbf{D}_{1M} + \mathbf{D}_{mod} \\ \mathbf{D}_{mod} &= \mathbf{D}_{1C} - \mathbf{E}_{trunc}^T \mathbf{B}^{-1} \mathbf{E}_{trunc} \end{aligned} \tag{4.40}$$

and

$$\mathbf{c}_1^* = \mathbf{c}_{1M} - \mathbf{D}_{mod}\left(\hat{\mathbf{x}}_1 - \hat{\mathbf{x}}_{1_M}\right) + \mathbf{c}_{mod}$$
$$\mathbf{c}_{mod} = \mathbf{c}_{1C} - \mathbf{E}_{trunc}^T \mathbf{B}^{-1}\mathbf{c}_0. \tag{4.41}$$

Details of this derivation are included in Appendix A.1. Note that $\mathbf{B}$, $\mathbf{E}_{trunc}$, and $\mathbf{D}_{1C}$ involve Jacobians evaluated at the estimate of $\mathbf{x}_0$ at the time of marginalization, denoted by $\hat{\mathbf{x}}_{0_M}$. Thus, in the above expressions, $\mathbf{D}_{mod}$ is a constant and it is stored and added to the new Hessian matrix $\mathbf{H}_{new}$ at every iteration of the minimization algorithm. In contrast, $\mathbf{D}_{1M}$ consists of Jacobians evaluated at the latest estimate of $\mathbf{x}_1$ and hence changes as new measurements are obtained.

Similarly, out of the terms comprising $\mathbf{c}_1^*$, $\mathbf{c}_{1C}$ is evaluated at $\hat{\mathbf{x}}_{0_M}$, and hence remains constant. While both $\mathbf{c}_{1M}$ and $\mathbf{c}_0$ depend on the latest estimate of $\mathbf{x}_1$, we introduce an approximation by evaluating $\mathbf{c}_0$ at $\hat{\mathbf{x}}_{1_M}$ and treating it as a constant. Hence, the term $\mathbf{c}_{mod}$ remains constant and it can be stored and added to $\mathbf{e}_{new}$ at every iteration of the minimization algorithm. Finally, $\mathbf{c}_{1M}$, which depends on the current estimate of $\mathbf{x}_1$, will be updated when new measurements are obtained in future time steps.

In summary, after each marginalization step, three terms have to be stored: (i) $\mathbf{D}_{mod}$, (ii) $\mathbf{c}_{mod}$, and (iii) $\hat{\mathbf{x}}_{1_M}$. Note that the dimensions of the above three quantities remain the same, irrespective of the number of time steps being marginalized simultaneously. If the first p time steps, i.e., time-steps 0 to $(p-1)$, are marginalized, only $\hat{\mathbf{x}}_{p_M}$, of dimension $N \times 1$ will have to be stored as a result of the correlations existing between time-steps $p-1$ and p. The correlations from the previous time steps will not affect the new Hessian. Also, the dimensions of $\mathbf{D}_{mod}$ and $\mathbf{c}_{mod}$ remain $N \times N$ and $N \times 1$, respectively (see Appendix A.2 for details). Since these dimensions depend only upon the number of robots in the team, the necessary memory assignment can be easily made before the team is deployed.

Distributed Marginalization

We hereafter discuss the distributed implementation of the marginalization process. The 2$^{\text{nd}}$ column of Table 4.3 lists the steps involved in marginalization. Step 1 requires the inversion of the $lN \times lN$ matrix $\mathbf{B}$, where l is the number of time steps being

marginalized. Thus, when the number of robots N and/or steps l is large, computation of this dense inverse centrally can be a bottleneck [$O(lN^3)$] [64].

Table 4.3: Complexity Analysis of Marginalization ($l = 1$)

		Computation		Comm.
Algorithm		Centralized	Distributed	Distributed
Step 1	$\mathbf{B}^{-1}\mathbf{E}_{trunc}$ and $\mathbf{B}^{-1}\mathbf{c}_0$	$O(N^3)$	$O(N^2)$	$O(N)$
Step 2a	$\mathbf{E}_{trunc}^T\mathbf{B}^{-1}\mathbf{E}_{trunc}$	$O(N^2)$	$O(N)$	0
Step 2b	$\mathbf{E}_{trunc}^T\mathbf{B}^{-1}\mathbf{c}_0$	$O(N)$	$O(1)$	0
Step 3a	$\mathbf{D}_{1C} - \mathbf{E}_{trunc}^T\mathbf{B}^{-1}\mathbf{E}_{trunc}$	$O(N)$	$O(1)$	0
Step 3b	$\mathbf{c}_{1C} - \mathbf{E}_{trunc}^T\mathbf{B}^{-1}\mathbf{c}_0$	$O(N)$	$O(1)$	0

Algorithm 3 Distributed Gauss-Jordan

for $i = 1$ to $N - 1$ **do**
- Divide Row i by b_{ii}

Computation cost = $N + 2$ {$N - i + 1$ elements in the Row i of matrix $\mathbf{B}$, i elements in Row i of matrix $\mathbf{E}_{trunc}$ and i^{th} element of $\mathbf{c}_0$}
Communication cost = 0
- Broadcast Row i of robot i to all other robots

Computation cost = 0
Communication cost = $N + 1$ {$N - i$ elements in Row i of matrix $\mathbf{B}$ because all elements until b_{ii} do not need to be communicated, i elements in Row i of matrix $\mathbf{E}_{trunc}$ and the i^{th} element of $\mathbf{c}_0$}
for $j = 1$ to N ; $j \neq i$ **do**
- Compute $Row\ j = Row\ j - pivot \times Row\ i$ {No need to compute the pivot as the pivot element will be the same as b_{ji}}

Computation cost = $2N + 3$
Communication cost = 0
end for {This operation is simultaneously carried out by all j robots}
end for

To address this problem, we use the distributed Gauss-Jordan method [110]. Instead of inverting $\mathbf{B}$ separately, we calculate the quantities $\mathbf{B}^{-1}\mathbf{E}_{trunc}$ and $\mathbf{B}^{-1}\mathbf{c}_0$ directly. Specifically, a new augmented matrix $\mathbf{M} = [\mathbf{B}\ \mathbf{E}_{trunc}\ \mathbf{c}_0]$ is considered and using the Gauss-Jordan algorithm, it is reduced to $[\mathbf{I}\ \mathbf{E}^*_{trunc}\ \mathbf{c}^*_0]$, where $\mathbf{I}$ is the identity matrix of the same dimensions as $\mathbf{B}$. The resulting terms $\mathbf{E}^*_{trunc}$ and $\mathbf{c}^*_0$ are equal to $\mathbf{B}^{-1}\mathbf{E}_{trunc}$ and $\mathbf{B}^{-1}\mathbf{c}_0$, respectively. Algorithm 3 presents the distributed Gauss-Jordan method (for

$l = 1$) which requires $O(N^2)$ operations and has communication cost $O(N)$ per robot.[11] Here, we note that Gauss-Jordan for the positive definite matrix $\mathbf{B}$ is numerically stable and hence does not require pivoting [64]. This reduces the communication overhead.

Once Step 1 of the marginalization process is complete (see Table 4.3), each robot has a row of $\mathbf{B}^{-1}\mathbf{E}_{trunc}$ and an element of $\mathbf{B}^{-1}\mathbf{c}_0$ stored locally. In Steps 2 and 3, each robot calculates a row of $\mathbf{E}_{trunc}^T\mathbf{B}^{-1}\mathbf{E}_{trunc}$ and then $\mathbf{D}_{mod} = \mathbf{D}_{1C} - \mathbf{E}_{trunc}^T\mathbf{B}^{-1}\mathbf{E}_{trunc}$ (note that $\mathbf{D}_{1C}$ is diagonal), and an element of $\mathbf{E}_{trunc}^T\mathbf{B}^{-1}\mathbf{c}_0$, followed by $\mathbf{c}_{mod} = \mathbf{c}_{1C} - \mathbf{E}_{trunc}^T\mathbf{B}^{-1}\mathbf{c}_0$ locally with computational cost of $O(N)$ per robot. Thus, the computational complexity of the distributed implementation of marginalization is reduced by an order of magnitude to $O(N^2)$ (or $O(lN^2)$ for $l > 1$).

4.6 Simulation Results

The performance of the proposed distributed MAP-based CL algorithm was first tested in simulation. We consider a team of $N = 18$ robots moving in 2D following phase-shifted sinusoidal trajectories. The robots move in an area of approximately 25 m $\times$ 90 m for 450 time steps $\delta t = 0.05$ sec. Each robot measures its linear, v, and rotational, ω, velocity, as well as its distance, d, and bearing, θ, to all other robots in the team. The noise in all measurements is modeled as zero-mean, white Gaussian with standard deviation $\sigma_v = 2\%v$, $\sigma_\omega = 1$ deg/sec for the linear, rotational velocity measurements, respectively, and $\sigma_d = 2\%d$, $\sigma_\theta = 1$ deg for the corresponding distance, bearing measurements.

The linearized least-squares minimization problem is solved every 5 time steps (i.e., measurements over 5 time steps are batch processed). We consider a sliding time window of $K = 10$ time steps, while marginalization is carried out every 5 time steps. Specifically, once the state vector contains robots' poses from 10 time steps, poses from the earliest 5 time steps are marginalized simultaneously. Thus, the number of time steps considered in the estimated state vector varies between 5 and 10. We compare the performances of the following approaches for CL:

1. C-EKF: Centralized EKF (computational complexity $O(N^3)$).

[11] At this point, we would like to remind the reader that robot i stores the i^{th} row of $\mathbf{B}$, i^{th} diagonal element of $\mathbf{E}_{trunc}$ and i^{th} element of $\mathbf{c}_1$.

2. D-CG-MAP: Distributed MAP-based estimator using the DCG algorithm and marginalization[12] (computational complexity $O(K^2N^2)$).

3. aD-CG-MAP: Distributed MAP-based estimator using an approximate DCG algorithm and marginalization (computational complexity $O(KN^2)$). In this case, we allow the DCG algorithm to perform N iterations only while solving a system of linear equations of dimensions $KN \times KN$.

In general, the number of time steps, K, considered and the number of DCG iterations are design parameters that can be adjusted so as to trade processing for increased accuracy. As compared to C-EKF, our proposed algorithm, D-CG-MAP, has reduced computational complexity when a small (compared to the size of the team) number of time steps, K, is considered, i.e., when $K \ll N$, $O(K^2N^2) \ll O(N^3)$. However, when the robots need to consider a large number of time steps, in order to reduce the effect of linearization errors, the approximate version of the DCG algorithm, aD-CG-MAP, is used. In this case, the DCG is allowed to run for N iterations, instead of KN, for a total cost of $O(KN^2)$.

We employ the RMS error criterion to test the accuracy of these three approaches. Fig. 4.6(a) shows the RMS error in the robots' position estimates averaged over 30 runs. As evident, the distributed MAP-based estimators (D-CG-MAP and aD-CG-MAP) outperform the C-EKF in terms of accuracy. This is due to the fact that the MAP estimator reduces the linearization errors over all K time steps considered (sliding window smoothing), and thus improves the accuracy of the robots' pose estimates. Furthermore, we see that aD-CG-MAP, which is an approximation of the D-CG-MAP, is also more accurate than the C-EKF. This can be attributed to the fast convergence of the DCG for positive definite matrices (in this case the Hessian $\mathbf{H}$), resulting in accuracy comparable to that of D-CG-MAP, even when the DCG is allowed to run for only N iterations.

Fig. 4.6(b), that depicts the RMS error in the robots' orientation estimates corroborates the results of Fig 4.6(a). In this figure, the improvement in the accuracy of

[12] We have not compared the performance of the distributed MAP-estimator without marginalization as the computational requirements of this approach far exceeded the processing and memory capabilities typically available to a robot (hence the importance of developing real-time distributed implementations for the CL problem).

the robots' orientation estimates for the MAP-based algorithms, as compared to the C-EKF, is more pronounced. Here, D-CG-MAP is the most accurate, followed closely by aD-CG-MAP, which outperforms C-EKF in terms of accuracy, while requiring fewer operations.

4.7 Experimental Results

Experimental validation of our proposed approach was carried out using a team of four Pioneer-I robots. The robots move in a rectangular arena with area of approximately 4 m $\times$ 2.5 m over a period of about 14 minutes. An overhead camera is employed for obtaining the pose of the robots in a global coordinate frame (ground truth) and used for comparing the performance of the D-CG-MAP and aD-CG-MAP estimators to the C-EKF.

The robots move at an approximately constant velocity of 0.1 m/sec and at the same time ensure that they do not collide with the boundaries of the rectangular arena or the other robots in the team. The robots obtain linear and rotational velocity measurements at a frequency of 1 Hz. The noise in the rotational velocity measurements is modeled as zero-mean, white Gaussian with standard deviation of 0.02 rad/sec, 0.0078 rad/sec, 0.0144 rad/sec, and 0.0105 rad/sec for robots 1, 2, 3, and 4 respectively. Similarly, the standard deviations for the noise in the linear velocity measurements are 0.005 m/sec, 0.0032 m/sec, 0.00585 m/sec, and 0.005125 m/sec for robots 1, 2, 3, and 4 respectively.

Relative position measurements between the robots are generated synthetically by using the data from the overhead camera and adding noise to the calculated relative position measurements. The noise in these measurements is also modeled as zero-mean, white Gaussian with a standard deviation of $\sigma_d = 2\% * d_{max}$ for the distance measurements, where d_{max} is the maximum distance measurement that is obtained and standard deviation of $\sigma_\theta = 1$ degree for the bearing measurements. Since we have a team of $N = 4$ robots, we can obtain a maximum of $N(N-1) = 12$ relative position measurements at each time step, corresponding to a complete measurement graph.

In distributed MAP-based CL, the linearized least-squares minimization problem is solved every 5 time steps (i.e., measurements over 5 time steps are batch processed). Furthermore, we consider a sliding time window of a maximum size of $K = 15$ time

steps, while marginalization is carried out every 5 time steps. Hence, the number of time steps considered in the estimated state vector varies between 10 and 15. As opposed to the simulation setup, where we allow the aD-CG-MAP to run for N iterations, here we allow it to run for K iterations, since for this particular experimental setup, K iterations yield sufficient accuracy.

Fig. 4.7 shows the trajectories of robot 1 generated using the three approaches above and compares it to the trajectory obtained using the data from the overhead camera (ground truth).

We use the RMS criterion to test the accuracy of these three approaches. Figs. 4.8(a) and 4.8(b) show the plots of the RMS errors in the robots' position and orientation estimates respectively. Corroborating the simulation results, we see that the distributed MAP estimators (D-CG-MAP and aD-CG-MAP) perform better than the C-EKF in terms of accuracy. The MAP estimator, by acting as a sliding window smoother, improves the accuracy of the robots' pose estimates. Furthermore, we see that the performance of the aD-CG-MAP approach is very close to that of the D-CG-MAP. This is due to the fast convergence of the DCG for positive definite matrices (i.e., $\mathbf{H}$).

4.8 Summary

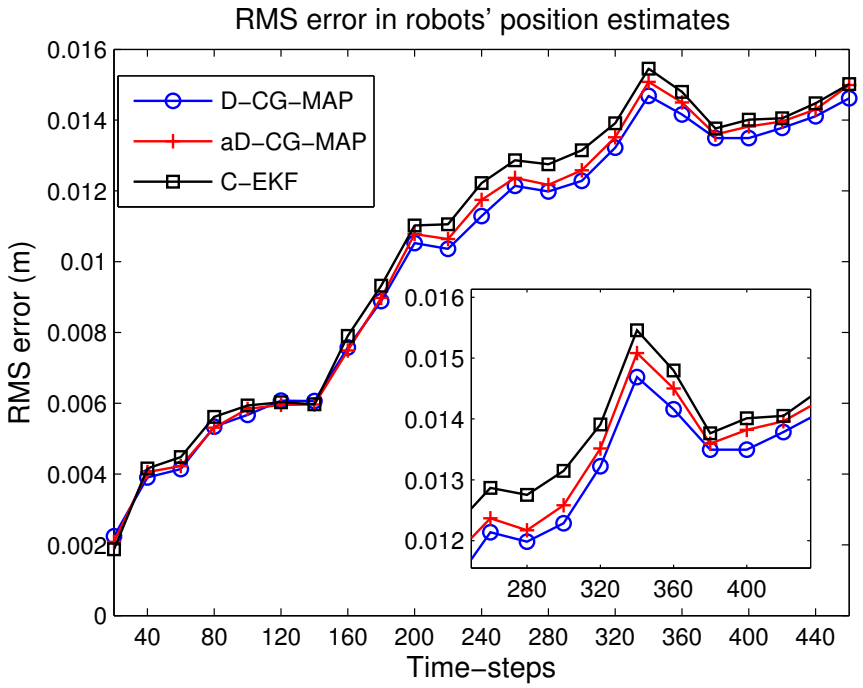
In this work, we introduced a novel distributed algorithm for MAP-based cooperative localization (CL) that takes advantage of all the available computational resources of a robot team to maximize performance while minimizing the processing load per robot. The proposed algorithm uses distributed data storage, the distributed conjugate gradient (DCG) algorithm, and distributed marginalization of past robot poses in order to divide the computations amongst the robots, and hence reduce the overall computational complexity of CL. Additionally, we have shown that by limiting the number of iterations of the DCG algorithm, the resulting approximate MAP estimator has accuracy almost indistinguishable from that of the MAP algorithm using the exact DCG, while significantly reducing the required number of operations.

$$
\mathbf{A} = \begin{bmatrix}
(\mathbf{P}_0^1)^{-1/2} & & & & & & & & & & & \\
& (\mathbf{P}_0^2)^{-1/2} & & & & & & & & & & \\
& & (\mathbf{P}_0^3)^{-1/2} & & & & & & & & & \\
-{}^1\overline{\mathbf{H}}_0^{1,2} & -{}^2\overline{\mathbf{H}}_0^{1,2} & & & & & & & & & & \\
-{}^1\overline{\mathbf{H}}_0^{2,1} & -{}^2\overline{\mathbf{H}}_0^{2,1} & & & & & & & & & & \\
-{}^1\overline{\mathbf{H}}_0^{1,3} & & -{}^3\overline{\mathbf{H}}_0^{1,3} & & & & & & & & & \\
-{}^1\overline{\mathbf{H}}_0^{3,1} & & -{}^3\overline{\mathbf{H}}_0^{3,1} & & & & & & & & & \\
& -{}^2\overline{\mathbf{H}}_0^{2,3} & -{}^3\overline{\mathbf{H}}_0^{2,3} & & & & & & & & & \\
& -{}^2\overline{\mathbf{H}}_0^{3,2} & -{}^3\overline{\mathbf{H}}_0^{3,2} & & & & & & & & & \\
-\overline{\mathbf{F}}_0^1 & & & (\mathbf{Q}_0^1)^{-1/2} & & & & & & & & \\
& -\overline{\mathbf{F}}_0^2 & & & (\mathbf{Q}_0^2)^{-1/2} & & & & & & & \\
& & -\overline{\mathbf{F}}_0^3 & & & (\mathbf{Q}_0^3)^{-1/2} & & & & & & \\
& & & -{}^1\overline{\mathbf{H}}_1^{1,2} & -{}^2\overline{\mathbf{H}}_1^{1,2} & & & & & & & \\
& & & -{}^1\overline{\mathbf{H}}_1^{2,1} & -{}^2\overline{\mathbf{H}}_1^{2,1} & & & & & & & \\
& & & -{}^1\overline{\mathbf{H}}_1^{1,3} & & -{}^3\overline{\mathbf{H}}_1^{1,3} & & & & & & \\
& & & -{}^1\overline{\mathbf{H}}_1^{3,1} & & -{}^3\overline{\mathbf{H}}_1^{3,1} & & & & & & \\
& & & & -{}^2\overline{\mathbf{H}}_1^{2,3} & -{}^3\overline{\mathbf{H}}_1^{2,3} & & & & & & \\
& & & & -{}^2\overline{\mathbf{H}}_1^{3,2} & -{}^3\overline{\mathbf{H}}_1^{3,2} & & & & & & \\
& & & -\overline{\mathbf{F}}_1^1 & & & (\mathbf{Q}_1^1)^{-1/2} & & & & & \\
& & & & -\overline{\mathbf{F}}_1^2 & & & (\mathbf{Q}_1^2)^{-1/2} & & & & \\
& & & & & -\overline{\mathbf{F}}_1^3 & & & (\mathbf{Q}_1^3)^{-1/2} & & & \\
& & & & & & -{}^1\overline{\mathbf{H}}_2^{1,2} & -{}^2\overline{\mathbf{H}}_2^{1,2} & & & & \\
& & & & & & -{}^1\overline{\mathbf{H}}_2^{2,1} & -{}^2\overline{\mathbf{H}}_2^{2,1} & & & & \\
& & & & & & -{}^1\overline{\mathbf{H}}_2^{1,3} & & -{}^3\overline{\mathbf{H}}_2^{1,3} & & & \\
& & & & & & -{}^1\overline{\mathbf{H}}_2^{3,1} & & -{}^3\overline{\mathbf{H}}_2^{3,1} & & & \\
& & & & & & & -{}^2\overline{\mathbf{H}}_2^{2,3} & -{}^3\overline{\mathbf{H}}_2^{2,3} & & & \\
& & & & & & & -{}^2\overline{\mathbf{H}}_2^{3,2} & -{}^3\overline{\mathbf{H}}_2^{3,2} & & &
\end{bmatrix} \tag{4.25}
$$

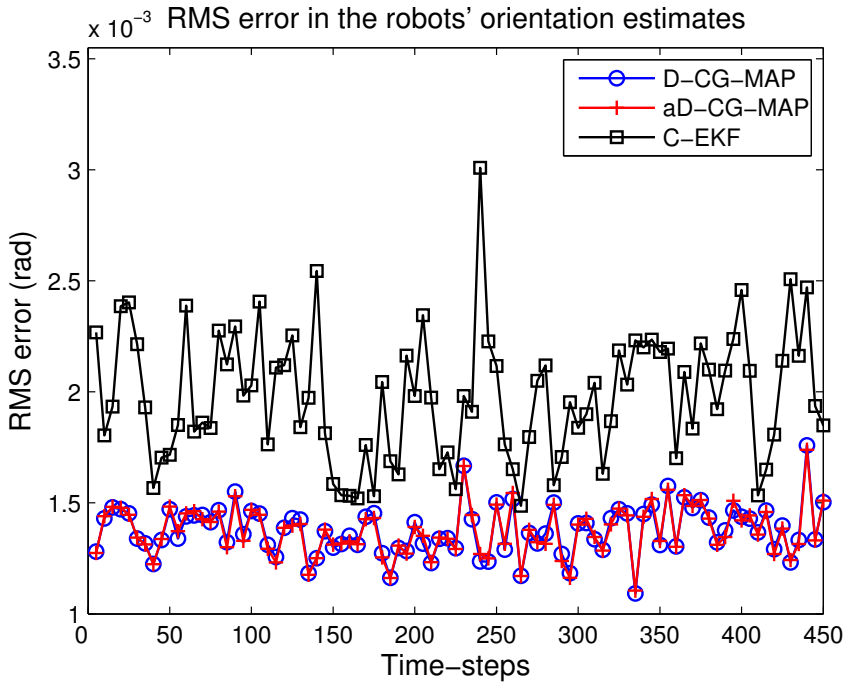
$$
\mathbf{b} = \begin{bmatrix}
-(\mathbf{P}_0^1)^{-1/2}(\hat{\mathbf{x}}_0^1 - \mathbf{x}_{init}^1) \\
-(\mathbf{P}_0^2)^{-1/2}(\hat{\mathbf{x}}_0^2 - \mathbf{x}_{init}^2) \\
-(\mathbf{P}_0^3)^{-1/2}(\hat{\mathbf{x}}_0^3 - \mathbf{x}_{init}^3) \\
-(\mathbf{R}_0^{1,2})^{-1/2}(\mathbf{z}_0^{1,2} - \mathbf{h}(\hat{\mathbf{x}}_0^1, \hat{\mathbf{x}}_0^2)) \\
-(\mathbf{R}_0^{2,1})^{-1/2}(\mathbf{z}_0^{2,1} - \mathbf{h}(\hat{\mathbf{x}}_0^2, \hat{\mathbf{x}}_0^1)) \\
-(\mathbf{R}_0^{1,3})^{-1/2}(\mathbf{z}_0^{1,3} - \mathbf{h}(\hat{\mathbf{x}}_0^1, \hat{\mathbf{x}}_0^3)) \\
-(\mathbf{R}_0^{3,1})^{-1/2}(\mathbf{z}_0^{3,1} - \mathbf{h}(\hat{\mathbf{x}}_0^3, \hat{\mathbf{x}}_0^1)) \\
-(\mathbf{R}_0^{2,3})^{-1/2}(\mathbf{z}_0^{2,3} - \mathbf{h}(\hat{\mathbf{x}}_0^2, \hat{\mathbf{x}}_0^3)) \\
-(\mathbf{R}_0^{3,2})^{-1/2}(\mathbf{z}_0^{3,2} - \mathbf{h}(\hat{\mathbf{x}}_0^3, \hat{\mathbf{x}}_0^2)) \\
-(\mathbf{Q}_0^1)^{-1/2}(\hat{\mathbf{x}}_1^1 - \hat{\mathbf{x}}_0^1 - \mathbf{f}(\hat{\mathbf{x}}_0^1, \mathbf{u}_{m_0}^1)\delta t) \\
-(\mathbf{Q}_0^2)^{-1/2}(\hat{\mathbf{x}}_1^2 - \hat{\mathbf{x}}_0^2 - \mathbf{f}(\hat{\mathbf{x}}_0^2, \mathbf{u}_{m_0}^2)\delta t) \\
-(\mathbf{Q}_0^3)^{-1/2}(\hat{\mathbf{x}}_1^3 - \hat{\mathbf{x}}_0^3 - \mathbf{f}(\hat{\mathbf{x}}_0^3, \mathbf{u}_{m_0}^3)\delta t) \\
-(\mathbf{R}_1^{1,2})^{-1/2}(\mathbf{z}_1^{1,2} - \mathbf{h}(\hat{\mathbf{x}}_1^1, \hat{\mathbf{x}}_1^2)) \\
-(\mathbf{R}_1^{2,1})^{-1/2}(\mathbf{z}_1^{2,1} - \mathbf{h}(\hat{\mathbf{x}}_1^2, \hat{\mathbf{x}}_1^1)) \\
-(\mathbf{R}_1^{1,3})^{-1/2}(\mathbf{z}_1^{1,3} - \mathbf{h}(\hat{\mathbf{x}}_1^1, \hat{\mathbf{x}}_1^3)) \\
-(\mathbf{R}_1^{3,1})^{-1/2}(\mathbf{z}_1^{3,1} - \mathbf{h}(\hat{\mathbf{x}}_1^3, \hat{\mathbf{x}}_1^1)) \\
-(\mathbf{R}_1^{2,3})^{-1/2}(\mathbf{z}_1^{2,3} - \mathbf{h}(\hat{\mathbf{x}}_1^2, \hat{\mathbf{x}}_1^3)) \\
-(\mathbf{R}_1^{3,2})^{-1/2}(\mathbf{z}_1^{3,2} - \mathbf{h}(\hat{\mathbf{x}}_1^3, \hat{\mathbf{x}}_1^2)) \\
-(\mathbf{Q}_1^1)^{-1/2}(\hat{\mathbf{x}}_2^1 - \hat{\mathbf{x}}_1^1 - \mathbf{f}(\hat{\mathbf{x}}_1^1, \mathbf{u}_{m_1}^1)\delta t) \\
-(\mathbf{Q}_1^2)^{-1/2}(\hat{\mathbf{x}}_2^2 - \hat{\mathbf{x}}_1^2 - \mathbf{f}(\hat{\mathbf{x}}_1^2, \mathbf{u}_{m_1}^2)\delta t) \\
-(\mathbf{Q}_1^3)^{-1/2}(\hat{\mathbf{x}}_2^3 - \hat{\mathbf{x}}_1^3 - \mathbf{f}(\hat{\mathbf{x}}_1^3, \mathbf{u}_{m_1}^3)\delta t) \\
-(\mathbf{R}_2^{1,2})^{-1/2}(\mathbf{z}_2^{1,2} - \mathbf{h}(\hat{\mathbf{x}}_2^1, \hat{\mathbf{x}}_2^2)) \\
-(\mathbf{R}_2^{2,1})^{-1/2}(\mathbf{z}_2^{2,1} - \mathbf{h}(\hat{\mathbf{x}}_2^2, \hat{\mathbf{x}}_2^1)) \\
-(\mathbf{R}_2^{1,3})^{-1/2}(\mathbf{z}_2^{1,3} - \mathbf{h}(\hat{\mathbf{x}}_2^1, \hat{\mathbf{x}}_2^3)) \\
-(\mathbf{R}_2^{3,1})^{-1/2}(\mathbf{z}_2^{3,1} - \mathbf{h}(\hat{\mathbf{x}}_2^3, \hat{\mathbf{x}}_2^1)) \\
-(\mathbf{R}_2^{2,3})^{-1/2}(\mathbf{z}_2^{2,3} - \mathbf{h}(\hat{\mathbf{x}}_2^2, \hat{\mathbf{x}}_2^3)) \\
-(\mathbf{R}_2^{3,2})^{-1/2}(\mathbf{z}_2^{3,2} - \mathbf{h}(\hat{\mathbf{x}}_2^3, \hat{\mathbf{x}}_2^2))
\end{bmatrix} \tag{4.26}
$$

$$\mathbf{A}_{new} = \begin{bmatrix} -(\mathbf{P}_0)^{-1/2}\mathbf{B}^{-1}\mathbf{E}_{trunc} & & \\ \left((\mathbf{Q}_0)^{-1/2} + \overline{\mathbf{F}}_0\mathbf{B}^{-1}\mathbf{E}_{trunc}\right) & & \\ \overline{\mathbf{H}}_0\mathbf{B}^{-1}\mathbf{E}_{trunc} & & \\ -\overline{\mathbf{H}}_1 & & \\ -\overline{\mathbf{F}}_1 & (\mathbf{Q}_1)^{-1/2} & \\ & -\overline{\mathbf{H}}_2 & \\ & -\overline{\mathbf{F}}_2 & (\mathbf{Q}_2)^{-1/2} \\ & & -\overline{\mathbf{H}}_3 \end{bmatrix} \tag{4.38}$$

$$\mathbf{b}_{new} = \begin{bmatrix} -(\mathbf{P}_0)^{-1/2}(\hat{\mathbf{x}}_{0M} - \mathbf{x}_{init}) - (\mathbf{P}_0)^{-1/2}\mathbf{B}^{-1}\mathbf{c}_0 \\ -(\mathbf{Q}_0)^{-1/2}(\hat{\mathbf{x}}_{1M} - \hat{\mathbf{x}}_{0M} - \mathbf{f}(\hat{\mathbf{x}}_{0M}, \mathbf{u}_{m_0})\delta t) - (\mathbf{Q}_0)^{-1/2}(\hat{\mathbf{x}}_1 - \hat{\mathbf{x}}_{1M}) + \overline{\mathbf{F}}_0\mathbf{B}^{-1}\mathbf{c}_0 \\ -(\mathbf{R}_0)^{-1/2}(\mathbf{z}_0 - \mathbf{h}(\hat{\mathbf{x}}_{0M})) + \overline{\mathbf{H}}_0\mathbf{B}^{-1}\mathbf{c}_0 \\ -(\mathbf{R}_1)^{-1/2}(\mathbf{z}_1 - \mathbf{h}(\hat{\mathbf{x}}_1)) \\ -(\mathbf{Q}_1)^{-1/2}(\hat{\mathbf{x}}_2 - \hat{\mathbf{x}}_1 - \mathbf{f}(\hat{\mathbf{x}}_1, \mathbf{u}_{m_1})\delta t) \\ -(\mathbf{R}_2)^{-1/2}(\mathbf{z}_2 - \mathbf{h}(\hat{\mathbf{x}}_2)) \\ -(\mathbf{Q}_2)^{-1/2}(\hat{\mathbf{x}}_3 - \hat{\mathbf{x}}_2 - \mathbf{f}(\hat{\mathbf{x}}_2, \mathbf{u}_{m_2})\delta t) \\ -(\mathbf{R}_3)^{-1/2}(\mathbf{z}_3 - \mathbf{h}(\hat{\mathbf{x}}_3)) \end{bmatrix} \tag{4.39}$$



(a) RMS error in the robots' position estimates.



(b) RMS error in the robots' orientation estimates.

Figure 4.6: RMS error in robots' position and orientation estimates.

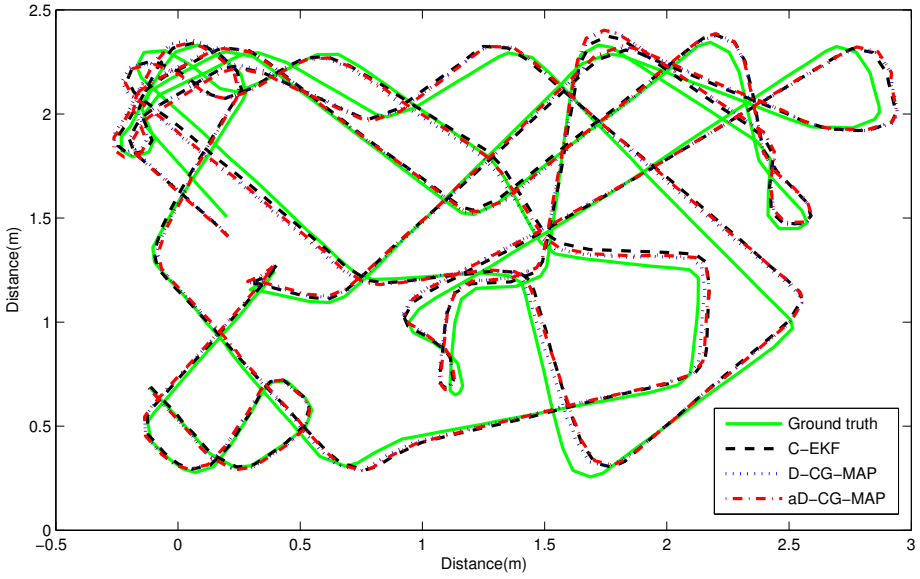
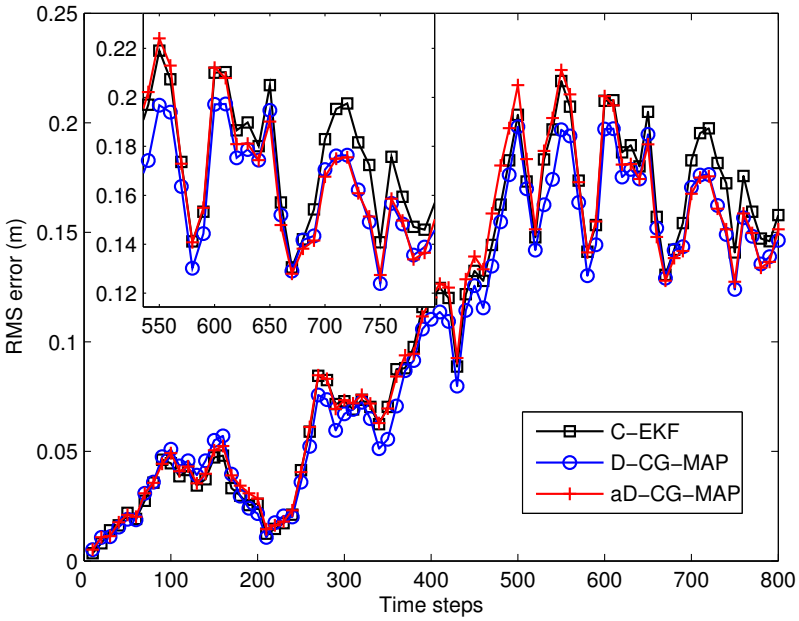
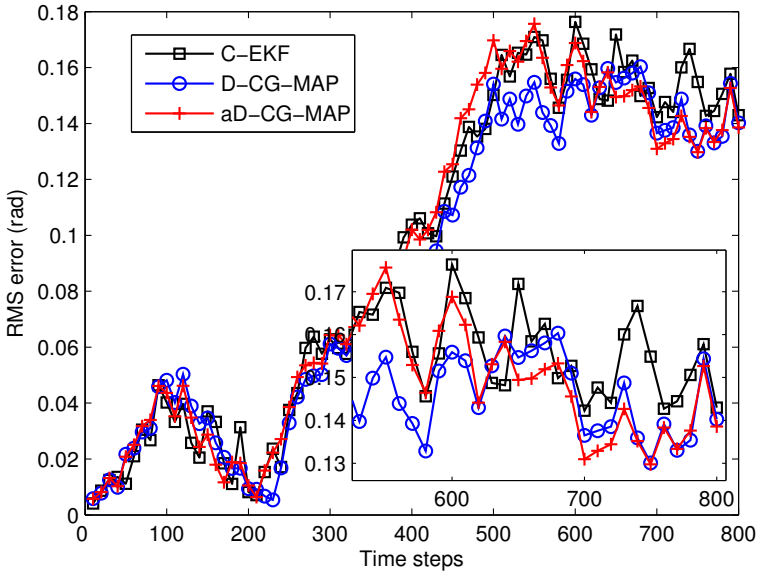


Figure 4.7: Robot 1 trajectories.



(a) RMS error in the robots' position estimates.



(b) RMS error in the robots' orientation estimates.

Figure 4.8: RMS error in the robots' position and orientation estimates.

Chapter 5

Multi-centralized CL under Asynchronous Communication Constraints

This chapter presents a generalized framework for inter-robot information-transfer schemes in Multi-Centralized Cooperative Localization (MC-CL) under *asynchronous communication*, i.e., when the communication graph associated with the mobile robot network is time-varying and intermittently disconnected. Specifically, two information-transfer schemes, which differ based on their communication bandwidth requirements per link, are discussed. Even under asynchronous communication constraints, these schemes enable robots to compute pose estimates *identical* to those generated using the centralized CL framework, albeit delayed. Parts of this chapter have been published in [109].

5.1 Introduction and Related Work

Cooperative Localization (CL) is a technique for multi-robot pose (i.e., position and orientation) determination. In CL, groups of communicating robots use their relative measurements (e.g., distance, bearing, and orientation) to *jointly* estimate their poses, resulting in increased accuracy for the entire team [37, 86, 27].

Traditionally in centralized CL, each robot communicates its *own* measurements to

a leader robot or a Fusion Center (FC) that processes these data to generate improved (centralized) pose estimates for the entire team. Depending upon the estimation framework used, various exact centralized algorithms have been proposed. Specifically, an Extended Kalman Filter-based (EKF) algorithm for CL has been introduced in [37], while in [32], the authors present a centralized Maximum Likelihood estimator-based approach to CL. The main drawback of these approaches is that all or most computations are performed centrally, rendering them susceptible to single-point failures of the FC. Moreover, for robot teams navigating in large environments, connectivity constraints (i.e., limited communication range), may prevent robots from sending their measurements to the FC.

An alternative approach that improves the robustness of the system is multi-centralized CL (MC-CL), wherein each robot acts as a FC, i.e., each robot broadcasts its *own* information to the entire team so that every robot can calculate the centralized pose estimates [111]. Although the MC-CL approach of [111] reduces the bandwidth requirements by communicating and processing quantized measurements, it requires a connected communication graph (associated with the mobile robot network) at the time of broadcast, i.e., it requires synchronous communication.

Sub-optimal EKF-based algorithms, that do not require uninterrupted inter-robot communication, are presented in [105], using the Interlaced Kalman filter, and in [106], using state-estimates exchange. An approach based on a hierarchy of EKFs is proposed in [107] where the robot group is divided into sub-teams with leaders estimating the state of their sub-team using an EKF. Furthermore, the leaders themselves can also form sub-teams, resulting into a hierarchical structure. The main drawback of these approaches is that in order to reduce the computational complexity of EKF-based CL, some (or even all in the case of [105]) correlations are ignored, which may lead to overly optimistic and inconsistent estimates.

An approach that maintains these cross-correlations by introducing a bank of EKFs at each robot, is presented in [112]. Each EKF in a bank corresponds to a relative measurement with another robot and accurate book-keeping is used to generate consistent estimates. However, the computational complexity of this approach grows exponentially with the team size. An exact distributed MAP-based algorithm for CL is presented in [34]. While this algorithm reduces the computational complexity of MAP-based CL,

it requires synchronous communication amongst the robots.

Recently, an exact MC-CL approach, that can handle both limited communication range and time-varying communication graphs (asynchronous communication) was proposed in [43]. To achieve this, the authors introduce an information-transfer scheme wherein each robot broadcasts *all* its locally-available information (its own past and present measurements, as well as past measurements previously received from other robots) to every robot within its communication radius at each time step. The proposed approach is independent of the estimation framework used and enables the robots to obtain delayed centralized estimates. The main drawback of this approach though is its high communication requirement *per link*, i.e., in communication resource-constrained applications, there might not exist sufficient bandwidth per link (or time during each exchange) for a robot to communicate all its local information.

The objective of our work is to develop a generalized framework for information-transfer schemes, which differ based on their bandwidth requirements per link, for performing MC-CL under asynchronous communication. Specifically, we present two information-transfer schemes, where each robot communicates: (i) only its *own* measurements, but for all time steps, and (ii) *all* measurement information available to it, but only from the oldest q time steps. By varying the parameter q, a family of information-transfer schemes can be generated, that includes the particular scheme proposed in [43]. Moreover, a trade-off can be achieved between the communication bandwidth requirement per link (increases with q) and the time delay in obtaining the centralized estimates (decreases with increasing q). By choosing an appropriate information-transfer scheme, based upon the communication resources available to the team, each robot can generate pose estimates *identical* to the centralized estimates (no approximations), but delayed. The proposed information-transfer schemes are independent of the estimator used, while the computational complexity per robot is identical to the corresponding centralized algorithm for CL.

In what follows, we first describe the problem formulation and each of the data-transfer schemes in detail along with their communication-complexity analysis. Then, for each of the proposed schemes, we develop necessary and sufficient conditions for communication-graph connectivity which, if satisfied, guarantee that each robot will be able to generate the centralized pose estimates, albeit delayed. Furthermore, we present

analytical results for the expected time delay in obtaining the centralized estimates for some of these schemes. Lastly, we present simulation results that compare the performance (trade-off between communication bandwidth per link and accuracy/delay) of the proposed information-transfer schemes.

5.2 Problem Formulation

Consider a team of N communicating robots navigating in 2D while performing CL. The state vector $\mathbf{x}_k = [{\mathbf{x}_k^1}^T, {\mathbf{x}_k^2}^T, \ldots, {\mathbf{x}_k^N}^T]^T$, where $\mathbf{x}_k^i = [x_k^i, y_k^i, \phi_k^i]^T$, $i = 1, \ldots, N$, contains the position and orientation of all robots at time-step k. Note that the team can use any estimation algorithm of its choice (e.g., EKF, MAP estimator, Particle filters, etc.) for pose determination. Each robot is equipped with proprioceptive (odometry) sensors that provide linear, $v_{m_k}^i$, and rotational, $\omega_{m_k}^i$, velocity measurements. The motion model for robot i is given by [109]:

$$\mathbf{x}_k^i = \mathbf{f}(\mathbf{x}_{k-1}^i, \mathbf{u}_{k-1}^i, \mathbf{w}_{k-1}^i), \quad \mathbf{w}_{k-1}^i \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_{k-1}^i), \tag{5.1}$$

where $\mathbf{f}$ is a general non-linear function, $\mathbf{u}_{k-1}^i = [v_{m_{k-1}}^i, \omega_{m_{k-1}}^i]^T$ is the control input and $\mathbf{w}_{k-1}^i = [w_{v_{k-1}}^i, w_{\omega_{k-1}}^i]^T$ is the process noise.

Additionally, all robots have exteroceptive sensors that allow them to uniquely identify other robots in the team and measure their relative distance and bearing. The measurement model for robot i measuring robot j is:

$$\mathbf{z}_k^{i,j} = \mathbf{h}(\mathbf{x}_k^i, \mathbf{x}_k^j) + \mathbf{n}_k^{i,j}, \quad \mathbf{n}_k^{i,j} \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_k^{i,j}), \tag{5.2}$$

with $\mathbf{h} = [d_k^{i,j}, \theta_k^{i,j}]^T$, where $d_k^{i,j}$, $\theta_k^{i,j}$ are the true distance, bearing respectively, from robot i to robot j at time-step k and $\mathbf{n}_k^{i,j} = [n_{d_k}^{i,j}, n_{\theta_k}^{i,j}]^T$ is the measurement noise.

Let $R = \{1, \ldots, N\}$ denote the set of indices of all robots in the team and let M_k^i denote the set of measurements generated by robot i at time-step k, i.e.,

$$M_k^i = \{\mathbf{u}_k^i, \mathbf{z}_k^{i,j}\}, \ \forall j \in R, \ j \neq i, \ d_k^{i,j} \leq d_{max}^i, \tag{5.3}$$

where d_{max}^i is the maximum sensing radius of robot i. At time-step k, robot i can communicate with robot j only if $d_k^{i,j} \leq r^i$, where r^i is the communication radius of robot i. Let C_k^i denote the set of indices of all robots that lie within the communication

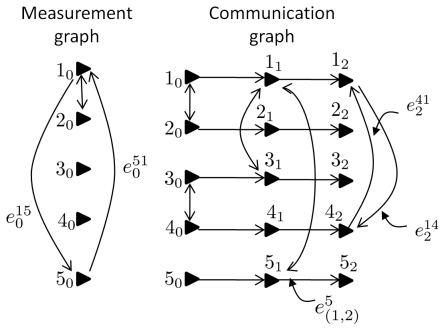


Figure 5.1: (Left) Measurement graph for a team of 5 robots at time-step 0. (Right) Communication graph for a team of 5 robots from time-steps 0 to 2.

radius of robot i at time-step k. Note that we assume bidirectional communication among the robots.

Depending on the robots' motion, each robot may sense and communicate with a different sub-team at every time step, i.e., both the measurement graph and the communication graph, associated with the mobile robot network, can be time-varying and incomplete. Specifically, the measurement graph for the robot team, at time-step k, is a directed graph where node i_k denotes robot i at time-step k and edge e_k^{ij} exists if robot i obtains a relative measurement to robot j (see Fig. 5.1). The communication graph considers a time-window from time-step k to $k+p$, and represents the flow of information in the robot network. Here, edge e_m^{ij}, $m \in \{k, \ldots, k+p\}$ exists, if robot i can communicate information to robot j at time-step m. Also, edge $e^i_{(m,m+1)}$ always exists, indicating that robot i's information from time-step m is always available to itself at time-step $m+1$.

We denote by $T_k^{i^-}$ the set of all information that is locally-available to robot i at time-step k, *before* it communicates with any other robot in the team. The set $T_k^{i^+}$ denotes the information that is locally-available to robot i *after* it has received information from all robots within communication range, i.e.,

$$T_k^{i^+} = T_k^{i^-} \cup \left(\bigcup_{j \in C_k^i} S_k^j \right), \tag{5.4}$$

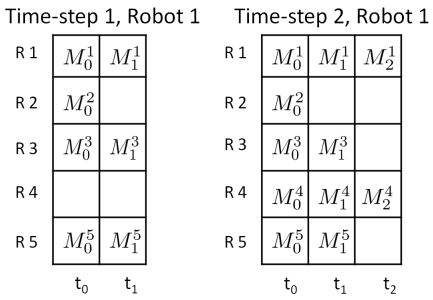


Figure 5.2: Information tables for robot 1 at time-steps 1 (left) and 2 (right). t_k: time-step k, Ri: Robot i.

where S_k^j denotes the information that is communicated by robot j to robot i at time-step k. The contents of S_k^j depend upon the information-transfer scheme used. Furthermore, $T_k^{i^-}$ can be expressed as:

$$T_k^{i^-} = T_{(k-1)}^{i^+} \cup M_k^i, \tag{5.5}$$

As the robot team moves around in the environment, depending on the evolution of the communication graph and the information-transfer scheme used, each robot starts accumulating information about the entire team. Let us assume that at time-step $k+m$, where $m \geq 0$, robot i obtains *all* information about the team up to time-step k, i.e., $T_{k+m}^{i^+} \supseteq \left(\bigcup_{j \in R} M_{0:k}^j\right)$. Using this information, robot i can calculate pose estimates[1] denoted by $\hat{\mathbf{x}}_{k|k}$, for the entire team, that are identical to those generated if using centralized CL. But since these estimates for time-step k can be calculated *no earlier than* time-step $k+m$ (due to the delay in information transfer), we denote these delayed estimates by $\hat{\mathbf{x}}_{k|k}^{k+m}$.

Consider the example shown in Fig. 5.1. The information tables in Fig. 5.2 depict the information that is available to robot 1 at time-steps 1 and 2. Note that robot 1 does not have any information about robot 4 at time-step 1. Now at time-step 2, assume that robot 4 communicates $S_2^4 = M_{0:2}^4$ to robot 1. Thus at time-step 2, robot 1 acquires all information about the team up to time-step 0 and can hence calculate the centralized pose estimates[2] , $\hat{\mathbf{x}}_{0|0}^2$.

[1] $\hat{\mathbf{x}}_{l|m}$ denotes pose estimates at time-step l, using all measurement information up to time-step m.

[2] Note that in addition to the delayed centralized estimate, each robot can compute a causal estimate for its pose given all measurements currently available to it.

Thus the flow of information between robots is governed not only by the communication graph, but also by the set S_k^j, which is determined by the information-transfer scheme used by the robot team. The choice of an appropriate information-transfer scheme, in turn is influenced by the communication bandwidth per link available to the robot team.

Before proceeding to the next section that presents our proposed information-transfer schemes, we briefly recap the notation used in this chapter:

- R: set of indices of all robots in the team.
- C_k^i: set of indices of robots that can communicate with robot i at time-step k.
- M_k^i: set of proprioceptive and exteroceptive measurements generated by robot i at time-step k.
- $T_k^{i^-}/T_k^{i^+}$: set of all information that is available to robot i up to and including time-step k, before/after communication with neighboring robots.
- S_k^i: set of information communicated by robot i to all robots within its communication radius at time-step k.

5.3 Information Transfer Schemes

5.3.1 Scheme 1: Own Information Transfer only

Description

In this scheme, each robot communicates only its *own* proprioceptive and exteroceptive measurement information to other robots. Assume that: (i) robot i and robot j last communicated with each other at time-step k, and (ii) the next communication opportunity for these two robots arises at time-step $k+p$, $p \geq 1$. Using Scheme 1, robot ℓ, $\ell \in \{i, j\}$, will communicate the set $S_{(k+p)}^{\ell} = M_{k+1:k+p}^{\ell}$ at time-step $k+p$. Therefore, after communication, the information set $T_{(k+p)}^{i^+}$ for each robot i, $i \in R$ will contain:

$$T_{(k+p)}^{i^+} = T_{(k+p)}^{i^-} \cup \left(\bigcup_{j \in C_{(k+p)}^i} S_{(k+p)}^j \right). \tag{5.6}$$

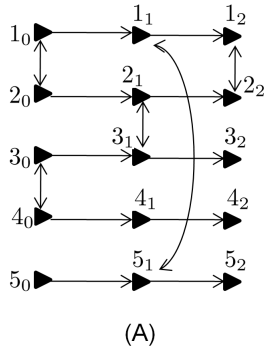


Figure 5.3: Scheme 1. Communication graph for a team of 5 robots from time-steps 0 to 2.

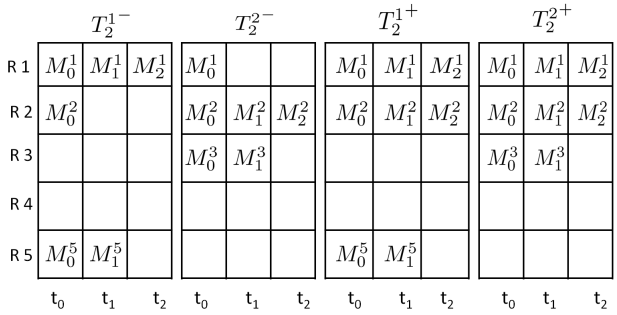


Figure 5.4: Scheme 1. Information tables for robots 1 and 2 at time-step 2, before and after communication.

Consider the example shown in Figs. 5.3 - 5.4. Since robots 1 and 2 last communicated at time-step 0, when they now meet at time-step 2, robot 1 will communicate only its own information from time-steps 1 and 2, i.e., $S_2^1 = M_{1:2}^1$, to robot 2. Similarly, robot 2 will also communicate only its own information from time-steps 1 and 2, i.e., $S_2^2 = M_{1:2}^2$, to robot 1. As a result, both robots will have identical information for each other but not for the rest of the team.

Note that in the information-transfer scheme proposed in [43], each robot communicates *all* its available information, i.e., information from all rows in its information table. By contrast, in Scheme 1 each robot communicates information *only* from a single row (corresponding to itself) in its information table.

Communication complexity analysis

When the two robots communicate after p time steps, each robot i has to communicate: (i) its p proprioceptive measurements, $\mathbf{u}^i_{k+1:k+p}$, and (ii) its exteroceptive measurements, $\mathbf{z}^{i,\ell}_K,\ \forall \ell \in R, \ell \neq i, d^{i,\ell}_K \leq d^i_{max},\ K = k+1, \ldots, k+p$. Therefore, the communication cost/amount of information that has to be transferred over the communication link is $O(|\mathbf{u}^i_{k+1:k+p}| + \sum_{K=k+1}^{k+p} |\mathbf{z}^{i,\ell}_K|)$, where $|g|$ denotes the cardinality of g. Here, the first term depends only on p which is determined by the frequency of inter-robot communication, while $|\mathbf{z}^{i,\ell}_K|$ also depends on the number of robots that are sensed by robot i per time step, i.e., the outdegree of robot i in the measurement graph. Therefore, for a particular application, if the frequency of inter-robot communication and the average outdegree of the measurement graph can be approximated before-hand, communication bandwidth per link can be reserved accordingly for CL. Moreover, this information-transfer scheme is well suited for applications where the robots communicate often (p remains small).

Communication graph connectivity analysis

We now present the necessary and sufficient conditions (on communication-graph connectivity), that if satisfied, guarantee that each robot in the team can compute the centralized pose estimates.

Lemma 1. *Robot i can compute*[3] $\hat{\mathbf{x}}^{max\{k_j\}}_{k|k}$ *if and only if robot j, $\forall j \in R,\ j \neq i$, communicates with robot i at time-step k_j, where $k_j \geq k$.*

Proof. Assume that robot j, $\forall j \in R, j \neq i$, communicates with robot i at time-step k_j, where $k_j \geq k$. Let $m_j < k$, denote the time step when robot j last communicated with robot i. If robot j never communicated with robot i before, we assume $m_j = -1$. At time-step k_j, robot j will communicate the set $S^j_{k_j} = \{M^j_{m_j+1:k_j}\}$ to robot i and thus:

$$T^{i^+}_{k_j} \supseteq \{M^i_{0:k_j}, M^j_{0:k_j}\} \supseteq \{M^i_{0:k}, M^j_{0:k}\}. \tag{5.7}$$

Let ℓ be the id of the robot that communicated with robot i at time-step $max\{k_j\}$.

[3] *Note that $\hat{\mathbf{x}}^m_{k|k}$ indicates that the centralized estimates for time-step k can be calculated* no earlier than *time-step m.*

Therefore, at the previous time-step $(max\{k_j\}-1)$:

$$T^{i^+}_{max\{k_j\}-1} \supseteq \{M^j_{0:k_j}\} \supseteq \{M^j_{0:k}\},\ \forall j \in R,\ j \neq \ell$$
$$T^{i^+}_{max\{k_j\}-1} \cap \{M^\ell_{m_\ell+1:k}\} = \emptyset, \tag{5.8}$$

i.e., at time-step $(max\{k_j\}-1)$, robot i will have information *up to time-step* k, for all robots in the team, *except* robot ℓ. Thus, robot i cannot calculate $\hat{\mathbf{x}}^{(max\{k_j\}-1)}_{k|k}$. But when robots ℓ and i communicate at time-step $max\{k_j\}$, $T^{i^+}_{max\{k_j\}} \supseteq \{M^j_{0:k}\},\ \forall j \in R$. Thus, robot i can compute $\hat{\mathbf{x}}^{max\{k_j\}}_{k|k}$.

We prove the second part using proof by contradiction, i.e., we show that if there exists a robot j that does not communicate with robot i at any time-step k_j, where $k_j \geq k$, then $\hat{\mathbf{x}}_{k|k}$ cannot be calculated. Since robot j last communicated with robot i at time-step m_j, ${T^i_{k_j}}^+ \cap \{M^j_{m_j+1:k}\} = \emptyset,\ \forall k_j \geq k$. Thus, $\hat{\mathbf{x}}_{k|k}$ cannot be calculated. □

Lastly, robot i can discard *all* information (own and other robots' measurements) up to time-step k at time-step $max\{k_j\}$, i.e., after calculating $\hat{\mathbf{x}}^{max\{k_j\}}_{k|k}$. Since we assume bidirectional communication between robots, if robot i has received information, up to time-step k from all other robots, this implies that robot i has communicated its own information about time-step k to all other robots too. Hence it can safely discard all the information and retain only the corresponding centralized estimates.

Expected time delay analysis

Let p^{ji}_m be the probability that edge e^{ji}_m exists in the communication graph. For simplicity, we assume that: (i) this probability remains constant over time (denoted by p^{ji}), and (ii) $p^{ji} = p^{ij}$. Therefore, the probability that edge e^{ji}_m does not exist is $(1-p^{ji})$. The expected time delay, $E_i(time\ delay)$, for robot i, $\forall i \in R$ is given by:

$$E_i(time\ delay) = \sum_{t=0}^{\infty} t \times p_i(time\ delay = t), \tag{5.9}$$

where $p_i(time\ delay = t)$ is the probability that the centralized estimates for time-step k can be calculated no earlier than time-step $k+t$.

Consider the event where the time delay in obtaining the centralized estimates is less than or equal to t, i.e., $time\ delay \leq t$. For this event to occur, robot j, $\forall\ j\ \in\ R,\ j \neq i$,

should communicate with robot i at least once up to time-step $k+t$, i.e., at least one edge e_m^{ji}, $k \leq m \leq k+t$ should exist in the communication graph for every robot j. The probability that there exists at least one edge between robot j and robot i from time-step k to time-step $k+t$ is given by:

$$p_i(\exists\ e_m^{ji},\ k \leq m \leq k+t) = 1-(1-p^{ji})^{t+1}. \tag{5.10}$$

Therefore the probability that there exists at least one edge between robot i and *every* other robot in the team from time-step k to $k+t$, i.e., $p_i(time\ delay \leq t)$ is given by:

$$\begin{aligned} &p_i(\exists\ e_{m_j}^{ji}, k \leq m_j \leq k+t, \forall j \in R, j \neq i) \\ &\qquad = \prod_{\substack{j=1 \\ j\neq i}}^{N} (1-(1-p^{ji})^{t+1}). \end{aligned} \tag{5.11}$$

Employing an analogous expression for $p_i(time\ delay \leq\ (t-1))$, we have:

$$\begin{aligned} &p_i(time\ delay = t) \\ &= p_i(time\ delay \leq t) - p_i(time\ delay \leq (t-1)) \\ &= \prod_{\substack{j=1 \\ j\neq i}}^{N} (1-(1-p^{ji})^{t+1}) - \prod_{\substack{j=1 \\ j\neq i}}^{N} (1-(1-p^{ji})^{t}). \end{aligned} \tag{5.12}$$

Substituting (5.12) in (5.9) yields:

$$\begin{aligned} &E_i(time\ delay) \\ &= \sum_{t=0}^{\infty} t \times \Big(\prod_{\substack{j=1 \\ j\neq i}}^{N} (1-(1-p^{ji})^{t+1}) - \prod_{\substack{j=1 \\ j\neq i}}^{N} (1-(1-p^{ji})^{t}) \Big). \end{aligned} \tag{5.13}$$

Thus, by modeling p^{ji} based on the network topology over time, the above formula can be used to accurately model the expected time delay.

In addition to the closed form expression in (5.12), we present two recursive expressions, one over the number of time steps and the other over the number of robots in the team. Now from (5.12) and (5.11), we obtain:

$$\sum_{s=0}^{t-1} p_i(time\ delay = s) = \prod_{\substack{j=1 \\ j\neq i}}^{N} (1-(1-p^{ji})^{t}). \tag{5.14}$$

Therefore, the recursive form of (5.12) over the number of time steps is given by:

$$p_i(time\ delay = t) = \prod_{\substack{j=1 \\ j\neq i}}^{N}(1-(1-p^{ji})^{t+1}) - \sum_{s=0}^{t-1} p_i(time\ delay = s). \tag{5.15}$$

where $p_i(time\ delay = 0) = \prod_{\substack{j=1 \\ j\neq i}}^{N} p^{ji}$.

We now consider recursion over the number of robots in the team. The event that $time\ delay = t$ for robot i, for a team of N robots can be expressed as the union of two sub-events:

E1: the $time\ delay = t$ for robot i, for a team of $N-1$ robots and at least one edge, e_m^{Ni} exists between robot N and robot i at time-step m, where $k \leq m \leq k+t$. Note that since $time\ delay = t$ for a team of $N-1$ robots, even if an edge exists between robot N and robot i at any time step before $k+t$, robot i cannot calculate the centralized estimates until time-step $k+t$. The probability that this event happens is given by $p_i(time\ delay = t, N-1\ robots)(1-(1-p^{Ni})^{t+1})$, where $p_i(time\ delay = t, N-1\ robots)$ denotes the probability that $time\ delay = t$ for robot i, for a team of $N-1$ robots.

E2: the edge e_{k+t}^{Ni} exists, i.e., edge from robot N to robot i exists at time-step $k+t$ and there exists at least one edge between robot i and all the remaining $N-2$ robots from time-step k to time-step $k+t-1$. Note that since robot i cannot communicate with robot N up to time-step $k+t$, the centralized estimates can be obtained only at time-step $k+t$. The probability of this event is given by $((1-p^{Ni})^t p^{Ni}) \prod_{\substack{j=1 \\ j\neq i}}^{N-1}(1-(1-p^{ji})^t)$.

Thus from (E1) and (E2) we obtain:

$$\begin{aligned} &p_i(time\ delay = t, N\ robots) \\ &= p_i(time\ delay = t, N-1\ robots)(1-(1-p^{Ni})^{t+1}) \\ &\quad + ((1-p^{Ni})^t p^{Ni}) \prod_{\substack{j=1 \\ j\neq i}}^{N-1}(1-(1-p^{ji})^t) \end{aligned} \tag{5.16}$$

where $p_i(time\ delay = t, 2\ robots) = (1-p^{ji})^t p^{ji}$, $j \neq i$.

We now present a numerical example for calculating the expected time delay using the formula in (5.13). For simplicity we consider $p^{ji} = p, \forall j \in R, j \neq i$. Therefore, (5.13) can be written as:

$$E_i(time\ delay) = \sum_{t=0}^{\infty} t \times \Big((1 - q^{t+1})^{N-1} - (1 - q^{t})^{N-1} \Big), \tag{5.17}$$

where $q = 1 - p$. Now, using Taylor's series expansion around $q^{t+1} = 0$, the term $(1 - q^{t+1})^{N-1}$ in (5.17) can be expressed as:

$$(1 - q^{t+1})^{N-1} = \sum_{a=0}^{N-1} \Big(\Big(\prod_{b=1}^{a} (N - b) \Big) (-1)^a (q^a)^{t+1} / a! \Big) \tag{5.18}$$

Similarly,

$$(1 - q^{t})^{N-1} = \sum_{a=0}^{N-1} \Big(\Big(\prod_{b=1}^{a} (N - b) \Big) (-1)^a (q^a)^{t} / a! \Big). \tag{5.19}$$

Therefore,

$$\begin{aligned} &(1 - q^{t+1})^{N-1} - (1 - q^{t})^{N-1} \\ &= \sum_{a=1}^{N-1} \Big(\Big(\prod_{b=1}^{a} (N - b) \Big) (-1)^{a-1} \big((q^a)^t - (q^a)^{t+1} \big) / a! \Big). \end{aligned} \tag{5.20}$$

Substituting (5.20) into (5.17) we obtain:

$$\begin{aligned} &E_i(time\ delay) \\ &= \sum_{t=1}^{\infty} t \times \Big[\sum_{a=1}^{N-1} \Big(\Big(\prod_{b=1}^{a} (N - b) \Big) (-1)^{a-1} \big((q^a)^t - (q^a)^{t+1} \big) / a! \Big) \Big] \\ &= \sum_{t=1}^{\infty} t \times \Big[\sum_{a=1}^{N-1} \Big(\Big(\prod_{b=1}^{a} (N - b) \Big) (-1)^{a-1} (1 - (q^a)) (q^a)^t / a! \Big) \Big] \\ &= \sum_{a=1}^{N-1} \Big(\Big(\prod_{b=1}^{a} (N - b) \Big) (-1)^{a-1} (1 - (q^a)) \sum_{t=1}^{\infty} t (q^a)^t / a! \Big). \end{aligned} \tag{5.21}$$

Substituting the formula for the mathematical series, $\sum_{t=1}^{\infty} t(q^a)^t = q^a/(1 - q^a)^2$, in (5.21) we get:

$$\begin{aligned} &E_i(time\ delay) \\ &= \sum_{a=1}^{N-1} \Big(\Big(\prod_{b=1}^{a} (N - b) \Big) (-1)^{a-1} q^a / \big((1 - q^a) a! \big) \Big). \end{aligned} \tag{5.22}$$

Thus using the above formula, we can obtain the expected value of the time delay for any number of robots.

5.3.2 Scheme 2: Information Transfer from q oldest time steps

Recall that in Scheme 1, information is transferred row-wise, i.e., each robot communicates the entire row corresponding to itself in its information table (see Fig. 5.4). Note, however that, for small robot teams communicating infrequently, the amount of information available in each row is usually significantly larger compared to the data stored in each column. In such cases, and in order to reduce the time delay in obtaining centralized estimates, information should be communicated column-wise, i.e., each robot should communicate all its locally available information, starting with the oldest time step first (i.e., the first column of its information table). The information-transfer scheme that we now propose is based on this general framework, where the parameter q defines the number of columns (time steps) that are communicated. When $q = 1$, all information from only the oldest time step is broadcast to neighbors. As q increases, information from more time steps is communicated, thus increasing the communication requirement per link, but reducing the delay in obtaining centralized estimates. Finally, the case where $q = \infty$ corresponds to the information-transfer scheme of [43], where each robot transfers all its locally-available information[4] .

Description

In this scheme, depending on the value of q, each robot communicates all its locally-available information from the oldest q time steps only. Let us assume that robot i has the oldest information about time-step k. Furthermore, at time-step m, $k \leq m \leq (k+q-1)$, robot i has information for itself, i.e., M_m^i, and for a subset, R_m^i, of the other robots in the team, i.e., $\{M_m^\ell\}$, $\forall \ell \in R_m^i$. If robot i communicates with robot j at time-step $k_j \geq k$, such that:

(1) $k_j \geq k+q-1$, robot i will communicate $S_{k_j}^i = \{M_m^i\} \cup \{M_m^\ell\}$, $\forall \ell \in R_m^i$, $k \leq m \leq (k+q-1)$.

(2) $k_j < k+q-1$, robot i will communicate $S_{k_j}^i = \{M_{k_j}^i\} \cup \{M_m^i\} \cup \{M_m^\ell\}$, $\forall \ell \in R_m^i$, $k \leq m < k_j$.

[4] We use the notation $q = \infty$ to indicate that information about an arbitrarily large number of time steps might have to be communicated, depending upon the evolution of the communication graph. For example, if two robots, that have not communicated for a long time period meet, then q can take on a very large value.

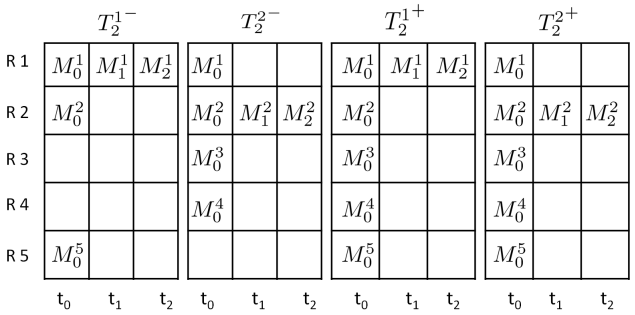


Figure 5.5: Scheme 2 with $q = 1$. Information tables for robots 1 and 2 at time-step 2, before and after communication.

Thus, after communication the information set for robot j at time-step k_j will contain:

$$T_{k_j}^{j\ +} = T_{k_j}^{j\ -} \cup \{S_{k_j}^r\}, \qquad \forall r \in C_{k_j}^j. \tag{5.23}$$

Next we consider the special case when $q = 1$. Here, each robot communicates all its locally-available measurement information from the *oldest* time step only. Assume that robot i has the oldest information about time-step k, for itself, i.e., M_k^i, and for a subset, R_k^i, of the other robots in the team, i.e., $\left(\bigcup_{\ell \in R_k^i} M_k^\ell\right)$. If robot i communicates with robot j at time-step $k_j \geq k$, robot i will transfer $S_{k_j}^i = M_k^i \cup \left(\bigcup_{\ell \in R_k^i} M_k^\ell\right)$. Thus, after communication the information set for robot j at time-step k_j will contain:

$$T_{k_j}^{j\ +} = T_{k_j}^{j\ -} \cup \left(\bigcup_{r \in C_{k_j}^j} S_{k_j}^r \right). \tag{5.24}$$

For the communication graph in Fig. 5.3, Fig. 5.5 shows the information tables for T_2^{1-} and T_2^{2-}, obtained using Scheme 2 with $q = 1$. When robots 1 and 2 communicate at time-step 2, robot 1 communicates $S_2^1 = M_0^5$ to robot 2 (it need not communicate robot 2's data to robot 2 itself), while robot 2 communicates $S_2^2 = M_0^3 \cup M_0^4$ to robot 1. Now since both robots have all information up to time-step 0, they can each calculate $\hat{\mathbf{x}}_{0|0}^2$. Finally, we note that the oldest information available to both robots after communication is the union of their individual oldest information before communication.

We now consider the case when $q = \infty$. While this scheme was originally proposed in [43], in this section we present additional communication complexity and expected time delay analysis. Furthermore, we also discuss a book-keeping technique that can be used by the robots to reduce communication overhead. In this scheme, each robot communicates all its locally-available measurement information, i.e., its own information and also the information that it received from other robots in the team. Assuming that robots i and j last communicated at time-step k, when they next communicate at time-step $k+p$, $p \geq 1$, each robot ℓ, $\ell \in \{i, j\}$, will communicate the set $S^{\ell}_{(k+p)}$ where:

$$S^{\ell}_{(k+p)} = M^{\ell}_{k+1:k+p} \cup \left(\bigcup_{k_r=k}^{k+p-1} \left(\bigcup_{r \in C^{\ell}_{k_r}} S^{r}_{k_r} \right) \right), \tag{5.25}$$

where $\left(\bigcup_{k_r=k}^{k+p-1} \left(\bigcup_{r \in C^{\ell}_{k_r}} S^{r}_{k_r} \right) \right)$ represents all the information that robot ℓ received from other robots, from time-step k up to time-step $k+p-1$. Therefore, after communication, the information set $T^{i^+}_{(k+p)}$ for each robot i, $i \in R$ will contain:

$$T^{i^+}_{(k+p)} = T^{i^-}_{(k+p)} \cup \left(\bigcup_{j \in C^{i}_{(k+p)}} S^{j}_{(k+p)} \right). \tag{5.26}$$

Using Scheme 2 with $q = \infty$, for the same communication graph as in Fig. 5.3, generates $T_2^{1^-}$ and $T_2^{2^-}$ as shown in Fig. 5.6. Now when robots 1 and 2 communicate at time-step 2, robot 1 communicates $S_2^1 = M_{1:2}^1 \cup M_{0:1}^5$ to robot 2, while robot 2 communicates $S_2^2 = M_{1:2}^2 \cup M_{0:1}^3 \cup M_0^4$ to robot 1. Similar to Scheme 2 with $q = 1$, both robots will be able to calculate $\hat{\mathbf{x}}^2_{0|0}$, but note that the information available about future time steps is different in both schemes. In particular, contrary to the case of $q = 1$, for $q = \infty$, $T_2^{1^+} = T_2^{2^+} = T_2^{1^-} \cup T_2^{2^-}$.

Since each robot communicates information about other robots too, depending on the evolution of the communication graph, there might exist an overlap between the information that is locally-available to two communicating robots. Thus by using an efficient book-keeping routine, the communication of redundant information can be avoided. In our proposed book-keeping approach, robot i first communicates a list of tuples $L_i = \{(\ell, k^i_\ell)\}$, where ℓ is the id of the robot whose measurement information is available to robot i, and k^i_ℓ is the latest time step for which this information is available.

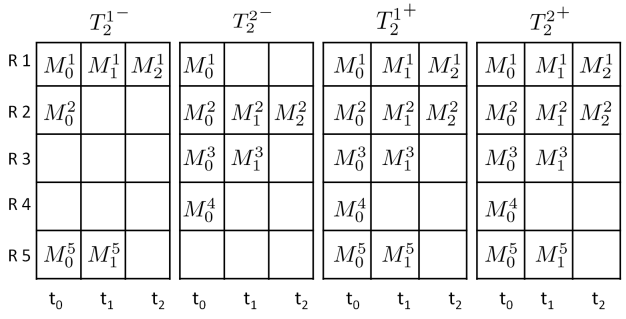


Figure 5.6: Scheme 2 with $q = \infty$. Information tables for robots 1 and 2 at time-step 2, before and after communication.

Once robot j receives this list from robot i, robot j calculates the set of robot ids, $V_j = \{v | (v \in L_i,\ v \in L_j,\ k_v^j > k_v^i) \vee (v \in L_j, v \notin L_i)\}$, of robots whose information has to be communicated to robot i and communicates only:

D1: $\{\mathbf{u}_K^v, \mathbf{z}_K^{v,r}\}$, where $v \in L_j, v \in L_i,\ \forall r \in R, r \neq v, d_K^{v,r} \leq d_{max}^v$ where $K = k_v^i + 1, \ldots, k_v^j$.

D2: $\{\mathbf{u}_K^v, \mathbf{z}_K^{v,r}\}$, where $v \in L_j,\ v \notin L_i,\ \forall r \in R, r \neq v, d_K^{v,r} \leq d_{max}^v$ where $K = m, \ldots, k_v^j$ and m is the earliest time step for which data about robot v is available.

Communication Complexity Analysis

When two robots meet, each robot has to communicate information about its q oldest time steps. For a team of N robots, the amount of information that has to be communicated by robot i, corresponding to a single time-step k can be at most $\{M_k^i\} \cup \{M_k^\ell\}$, $\forall \ell \in R, \ell \neq i$. Thus if information for q time steps has to be communicated, the communication complexity can be at most $O(|\{M_{k:k+q-1}^i\}| + |\{M_{k:k+q-1}^\ell\}|)$, $\forall \ell \in R, \ell \neq i$. Thus we see that the communication complexity of this information transfer scheme scales linearly with q and as q increases, the time delay in obtaining the centralized estimates decreases. Furthermore, this general framework generates a family of information transfer schemes and depending upon the communication resources available to the robot team, an appropriate scheme can be selected.

When $q = 1$, since each robot communicates only its oldest-available information, when robots i and j communicate, the communication cost/amount of information that

has to be transferred per communication link is $O(|M_k^i| + \sum_{\ell \in R_k^i} |M_k^\ell|) = O(|\mathbf{u}_k^i| + |\mathbf{z}_k^{i,p}| + \sum_{\ell \in R_k^i} (|\mathbf{u}_k^\ell| + |\mathbf{z}_k^{\ell,s}|)$, where p and s denote the robots that were observed by robots i and ℓ respectively, at time-step k. Importantly for this approach, the communication cost depends upon the outdegree of the measurement graph and the number of robots in the team, but it is *independent* of the frequency of inter-robot communication, i.e., irrespective of the time that has elapsed since robots i and j last communicated, when robot i next communicates with robot j, information for *only* a single time step will be communicated. This makes this scheme suitable for use in severe communication-resource constrained applications, but delays the calculation of centralized estimates.

For $q = \infty$, when robots i and j communicate after p time steps, robot i first communicates its list L_i and then depending on the list L_j that it obtains from robot j, it communicates the non-redundant measurement information corresponding to $\{L_i \backslash L_j\}$. Therefore, the total information that has to be communicated by robot i to robot j when they meet after p time steps is $|L_i|$ plus all the information mentioned in the previous section (sets $D1$ and $D2$ listed in book-keeping). While this scheme has the highest communication complexity per link, as compared to Scheme 1 and Scheme 2 with $q = 1$, it has the lowest time delay in obtaining the centralized estimates. Therefore, this scheme is applicable to scenarios where there is no constraint on the available communication bandwidth.

Communication graph connectivity analysis

We now present the necessary and sufficient condition for obtaining centralized estimates for Scheme 2.

Lemma 2. *Robot i can compute $\hat{\mathbf{x}}_{k|k}^{max\{k_j\}}$ if and only if there exists $j_k \rightarrow i_{k_j}$, $\forall j \in R$, $j \neq i$, where $k_j \geq k$. The notation $j_k \rightarrow i_{k_j}$ denotes a path in the communication graph from node j_k to node i_{k_j}, such that information about robot j up to time-step k, i.e., $M_{0:k}^j$, is available to robot i,* no earlier than *at time-step $k_j \geq k$.*

Proof. Assume $j_k \rightarrow i_{k_j}$, $\forall j \in R$, $j \neq i$, where $k_j \geq k$. Let $m_j < k$ be the time step up to which robot j's information, $M_{0:m_j}^j$, is already available to robot i through previous communication. If robot i never received any information from robot j before, we assume $m_j = -1$. Let robot ℓ be the robot for which $\ell_k \rightarrow i_{max\{k_j\}}$. Therefore, at

the previous time-step $(max\{k_j\}-1)$:

$$T^i_{max\{k_j\}-1}{}^{+} \supseteq \{M^j_{0:k_j}\} \supseteq \{M^j_{0:k}\},\ \forall j \in R,\ j \neq \ell$$
$$T^i_{max\{k_j\}-1}{}^{+} \cap \{M^\ell_{m_\ell+1:k}\} = \emptyset, \tag{5.27}$$

i.e., at time-step $(max\{k_j\}-1)$, robot i will have information *up to time-step* k, for all robots in the team, *except* robot ℓ. Therefore, robot i cannot calculate $\hat{\mathbf{x}}^{(max\{k_j\}-1)}_{k|k}$. But at time-step $max\{k_j\}$, robot ℓ's information up to time-step k will become available to robot i, i.e., $T^i_{max\{k_j\}}{}^{+} \supseteq \{M^j_{0:k}\},\ \forall j \in R$. Thus, robot i can compute $\hat{\mathbf{x}}^{max\{k_j\}}_{k|k}$.

We prove the second part using proof by contradiction, i.e., we show that if there exists a robot j such that the communication path $j_k \rightarrow i_{k_j}$ does not exist at any time-step $k_j \geq k$, then $\hat{\mathbf{x}}_{k|k}$ cannot be calculated. Since robot j last communicated with robot i at time-step m_j, $T^i_{k_j}{}^{+} \cap \{M^j_{m_j+1:k}\} = \emptyset,\ \forall k_j \geq k$. Thus, $\hat{\mathbf{x}}_{k|k}$ cannot be calculated[5] . □

While the statement of the above theorem is identical for all schemes generated with different values of q (i.e., they all require the existence of an appropriate path in the communication graph), the actual communication path in the graph differs based on the selected value of q. Let us assume that robots i and j communicate at time-step k. When $q = \infty$, each robot transfers all its locally-available information, and therefore the information communicated by robot i, at time-step k, *will include* M^i_k. But when $q = 1$, since only the oldest information is transmitted, the information communicated by robot i will include M^i_k, *only* if k is the oldest time step for which measurement information is available to robot i. For example, in Fig. 5.6, for $q = \infty$, when robots 1 and 2 communicate at time-step 2, robot 1 communicates information about time-step 2 too. On the other hand for $q = 1$ (see Fig. 5.5), at time-step 2 robot 1 communicates information only about the oldest time step, i.e., time-step 0. Robot 1 will communicate information about time-step 2 only after information about time-steps 0 and 1 has been discarded. Thus even though both cases require the existence of a path in the communication graph between robot j at time-step k and robot i at time-step k_j, these communication paths may not be the same.

Regardless of the choice of q, in Scheme 2 robot i can discard all information about time-step k, only after: (i) it has computed the centralized estimates for time-step k, and

[5] The derivation for Scheme 2 with $q = \infty$ has been presented in [43].

(ii) it has ensured that these centralized estimates (in the form of raw measurements) have been communicated to all other robots in the team. Robot i can ensure this by communicating this information to all robots personally, or by communicating with another robot in the team that has satisfied conditions (i) and (ii) above, whichever happens earlier[6] .

Expected time delay analysis

In Scheme 1, since each robot communicates only its own information, $E_i(time\ delay)$, $\forall i \in R$, depends only upon the $(N-1)$ possible communication links per time step, from other robots in the team to robot i. Thus the edges $e_k^{\ell r}$, $\forall \ell, r \in R \backslash i$, for any time-step k, do not affect $E_i(time\ delay)$. By contrast, in Scheme 2, since each robot can communicate information about all robots, $E_i(time\ delay)$ can be affected by any of the $N(N-1)$ possible edges per time step, depending upon the evolution of the communication graph. We now present the expected time delay analysis for a team of 3 robots using Scheme 2 with $q = \infty$. Future work will include generalization of this result to the case of N robots.

Assume that we want to calculate the centralized estimates for time-step k. Let $I_t^i = p_i(time\ delay > t)$ be the probability that the time delay in computing these centralized estimates is greater than t for robot i. Thus, $p_i(time\ delay \leq\ t) = 1 - I_t^i$ and $p_i(time\ delay =\ t) = -I_t^i + I_{t-1}^i$. We assume $p^{ij} = p, \forall i, j \in R$. Now in order to obtain an expression for $E_i(time\ delay)$ [see (5.9)], we need to calculate I_t^i. The event $(time\ delay > t)$ can be expressed as the union of two mutually-exclusive sub-events, (E1) and (E2), which in turn are expressed as the intersection of two independent sub-events each:

E1: (E1a) Neither of the other two robots in the team communicate with robot i at time-step $k+t$, i.e., edges e_{k+t}^{ji}, $\forall j \in R, j \neq i$, do not exist, and (E1b) the event $(time\ delay > t-1)$ is true for robot i. The probability of event (E1a) is $(1-p)^2$. Thus probability of event (E1) is given by $(1-p)^2 p_i(time\ delay > t-1)$.

[6] In the scheme proposed in [43], robot i discards the raw measurements at time-step k, as soon as it has calculated the centralized estimate for this time step and then communicates this centralized estimate to other robots. Note that irrespective of whether the raw measurements themselves or the centralized estimate computed from these measurements are communicated, the information content remains the same.

E2: (E2a) Either of the other two robots in the team (say robot j) communicates with robot i at time-step $k+t$, i.e., the edge e^{ji}_{k+t} exists at time-step $k+t$, and (E2b) the third robot in the team (say robot ℓ) does *not* communicate with either robot i or robot j from time-step k to time-step $k+t-1$, i.e., edges $e^{\ell i}_m$, $\ell \in R$, $\ell \neq i, j$, for $k \leq m \leq k+t-1$ do not exist. Now the probability of event (E2a) is given by $2(1-p)p$, while the probability of event (E2b) is given by $(1-p)^{2t}$. Therefore the probability of event (E2) is given by $2p(1-p)^{2t+1}$.

From the union of (E1) and (E2) we have:

$$I^i_t = p_i(time\ delay > t-1)(1-p)^2 + 2p(1-p)^{2t+1}. \tag{5.28}$$

Note that $p_i(time\ delay > 0) = (1-p^2)$ and $p_i(time\ delay = 0) = p^2$. Substituting (5.28) in (5.9) yields $E_i(time\ delay)$ as:

$$\begin{aligned} &\sum_{t=1}^{\infty} t \times \left(I^i_{t-1} - I^i_t\right) \\ &= \sum_{t=1}^{\infty} t \times \left(p_i(time\ delay > t-1)(2p-p^2) - 2p(1-p)^{2t+1}\right). \end{aligned} \tag{5.29}$$

5.4 Simulations

In this section, we compare the performances of the three information-transfer schemes discussed in this chapter. The objective of our simulation studies is twofold: (i) to compare the expected time delay in obtaining the centralized estimates for these schemes, and (ii) to study the trade-off between the accuracy of pose estimates for the robot team at time-step k, obtained using information available at time-step k (without including the delayed information), and the communication bandwidth requirements per link for these schemes.

To study the behavior of the expected time delay for these schemes, we carried out Monte-Carlo runs on a general mobile robot network of N robots over 1000 time steps, with $p^{ij} = 0.5, \forall i, j \in R$, for each scheme. Furthermore, by varying N, we studied the effect of the size of the robot team on the delay in obtaining the centralized estimates.

These results are presented in Table 5.1. First, we note that for Scheme 1 and Scheme 2 ($q = \infty$) for $N = 3$, the simulation results for the expected time delay corroborate our analytical results. Next, from Table 5.1 we see that the expected time delay is lowest for Scheme 2 ($q = \infty$), irrespective of the size of the robot team. This result is expected, because as compared to the other two approaches, this scheme has the highest rate of information transfer. However, this improvement in performance comes at the cost of increased communication complexity (see Sec. 5.3.2).

More importantly, in Scheme 2 ($q = \infty$), as the number of robots increases, the expected time delay goes on decreasing and finally becomes 1 time step. To corroborate this result, consider the $p_i(time\ delay\ >\ 1)$ for Scheme 2 ($q\ =\ \infty$) for a team of N robots:

$$\begin{aligned} & p_i(time\ delay > 1) \\ & = \sum_{r=1}^{N-1} \binom{N-1}{r} (1-p)^r p^{(N-1-r)} \left(1 - \left(1 - (1-p)^{(N-r)}\right)^r\right). \end{aligned} \tag{5.30}$$

Furthermore, $p_i(time\ delay = 1)$ can be expressed as:

$$\begin{aligned} & p_i(time\ delay \leq 1) - p_i(time\ delay = 0) \\ & = 1 - p_i(time\ delay > 1) - p^{N-1}. \end{aligned} \tag{5.31}$$

Table 5.2 shows the values of $p_i(time\ delay = 1)$ for different team sizes. From the table we see that as N increases, $p_i(time\ delay = 1)$ goes very quickly to 1. Hence the expected time delay also goes to 1, as seen in the simulation results.

The next lowest expected time delay is for Scheme 1, followed by Scheme 2 ($q = 1$). Moreover, the expected time delay for Scheme 2 ($q = 1$) is substantially larger than that for the other two approaches. This is due to the fact that as compared to the other two approaches, that do not have an upper bound on the time horizon over which information is transferred between robots, in this case information about *only* a single (oldest) time step is communicated. Unless this information has been discarded (i.e., *all* robots have computed the centralized estimates for this time step), information about the next time step cannot be communicated. As a result, the expected time delay in obtaining the centralized estimates goes on increasing with time.

In order to compare the accuracy of the robot pose estimates generated using the proposed information-transfer schemes, we consider a team of $N = 5$ robots moving

Table 5.1: Expected Time Delay Analysis

Team size	Sch. 1 (theor./sim.)	Sch. 2, $q = \infty$	Sch. 2, $q = 1$
3	1.6667/1.6488	1.2333/1.2222	66.0237
5	2.5048/2.4867	1.2876	66.7684
10	3.5813/3.5495	1.1673	69.9089
20	4.6183/4.5356	1.0231	70.6912
50	5.9621/5.8811	1	72.5475
100	6.9694/7.0152	1	74.2389

Table 5.2: $p_i(time\ delay = 1)$ for Scheme 2 ($q = \infty$)

Team size	$p_i(time\ delay = 1)$
3	0.5625
5	0.8789
10	0.9961
20	0.9999
50	1
100	1

in 2D with phase-shifted sinusoidal trajectories for 500 time steps (each time step has duration 0.05 sec). Each robot measures its linear, v, and rotational, ω, velocity, as well as its distance, d, and bearing, θ, to other robots in the team. The noise in these measurements is modeled as zero-mean, white Gaussian and has standard deviation $\sigma_v = 2\%v$, $\sigma_\omega = 1$ deg/sec for the linear and rotational velocity measurements, respectively, and $\sigma_d = 2\%d$ and $\sigma_\theta = 1$ deg for the corresponding distance and bearing measurements. Furthermore, as the robots move around, they randomly communicate with other robots in the team ($p^{ji} = p$). When two robots communicate, the information transferred depends upon the chosen information-transfer scheme.

The CL algorithm is implemented by each robot using an EKF. When a robot calculates pose estimates at the current time step, if the measurements for that time step are unavailable for some robots in the team, the last available proprioceptive measurements from those robots are used for the propagation step of the EKF.

We employ the RMS error criterion to test the accuracy of these three approaches.

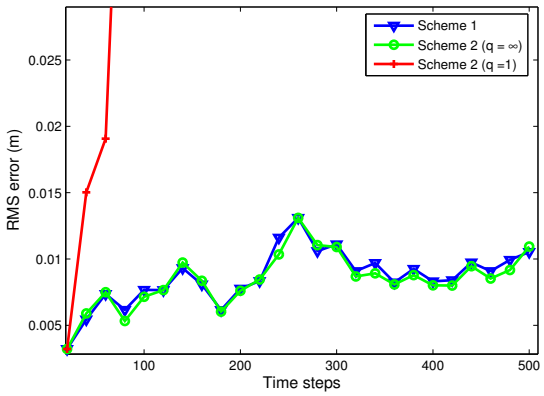


Figure 5.7: RMS error in the robots' position estimates.

Figs. 5.7 and 5.8 show the RMS error in the position and orientation estimates respectively, generated by robot 1 for the entire team. The RMS errors are plotted every 20 time steps for clarity. From the figures we see that for the chosen robot team of size $N = 5$, the performances of Scheme 2 ($q = \infty$) and Scheme 1 are almost indistinguishable from each other. Thus we can conclude that for small robot teams, using Scheme 1 instead of Scheme 2 ($q = \infty$), will save valuable communication bandwidth per link without any significant loss of accuracy. But as the number of robots increases, Scheme 2 will outperform the other two schemes. This is due to the fact that even though all information necessary for generating the centralized estimates, i.e., $\hat{\mathbf{x}}^k_{k|k}$, may not be available to robot i at time-step k, the set T^{i+}_k (i.e., locally-available information at time-step k) generated using Scheme 2 ($q = \infty$) is a superset of T^{i+}_k generated using the other two schemes. Lastly, as expected, the pose estimates generated using Scheme 2 ($q = 1$) are the least accurate. Since the communication bandwidth available per link for Scheme 2 ($q = 1$) is very restrictive, this translates into lower accuracy for the pose estimates. Thus depending upon the availability of communication resources and the accuracy requirements of the application, a suitable information-transfer scheme should be chosen.

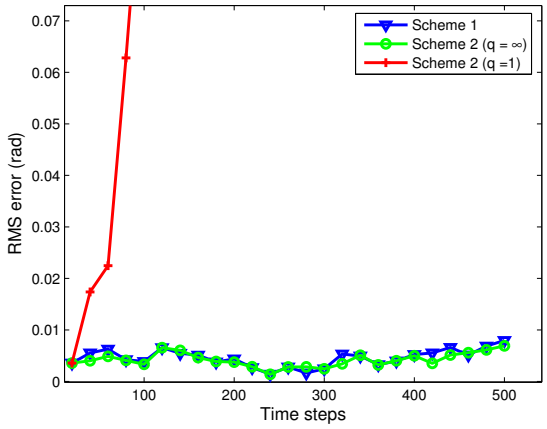


Figure 5.8: RMS error in the robots' orientation estimates.

5.5 Summary

In this chapter, we presented a general framework for inter-robot information-transfer schemes in multi-centralized CL under asynchronous communication. Depending upon the team's and the application's communication bandwidth availability, an appropriate information-transfer scheme can be selected to obtain pose estimates identical to the centralized estimates, but delayed. Moreover, to facilitate the choice of an appropriate information transfer scheme for a particular application, we have carried out detailed communication-complexity analysis, studied the necessary and sufficient conditions to generate centralized estimates, and performed detailed expected time delay analysis.

Chapter 6

MMSE-based Hybrid Estimation Framework for CL

In this chapter, we present a novel hybrid estimation framework (HEF) for multi-robot Cooperative Localization (CL) under time-varying, communication-bandwidth constraints. In the presence of severe communication constraints, robots can communicate only a quantized version of their analog measurements to the team. As opposed to existing estimators that can process either analog or quantized measurements, the HEF enables robots to process all available information, i.e., local analog measurements (recorded by its *own* sensors) and remote quantized measurements (collected and communicated by *other* robots), to obtain accurate robot pose estimates. Moreover, since the communication-bandwidth available for CL is *time-varying*, the HEF is resource-aware and can utilize additional bandwidth, whenever available, to further improve the estimation accuracy.

6.1 Introduction and Related Work

In wireless sensor (robot) network (WSN) applications, sensors typically estimate a quantity of interest, using noisy measurements from *all* sensor nodes. An example is multi-robot target tracking [113], where robots estimate a target's position, based on their noisy distance / bearing measurements to the target. Since the sensor nodes are spatially distributed, each node generally has to communicate its local information

to the team. Therefore, for WSNs deployed in (i) environments with inherent communication limitations or (ii) applications with power/battery restrictions, it becomes necessary to develop decentralized estimation algorithms that can trade communication bandwidth-availability for estimation accuracy. While bandwidth constraints exist in many WSN applications, this work focuses on the representative application of Cooperative Localization (CL).

CL incurs substantial communication overhead since each robot has to communicate its own local information, either its pose estimate and covariance, or its exteroceptive and proprioceptive measurements, with the team [28], [43]. However, factors such as: (i) inherent communication-bandwidth limitations in underwater applications, resulting from diffraction, attenuation, scattering, and fading, or (ii) constraints imposed on data transmission to increase the robots' operating time, may allow robots to communicate only a part of the required information. Moreover, the bandwidth available for CL is *time-varying* and depends upon the current resource-requirements of higher-level tasks and the battery-life of the robots. Therefore, the design of estimation algorithms for CL is dictated by these application-specific, communication-bandwidth considerations, and various approaches addressing these time-varying bandwidth constraints have been proposed in the literature.

Extended Kalman Filter (EKF)-based approaches where robots select and transmit all or a subset of their analog[1] measurements, depending upon the bandwidth-availability and subject to a suitable selection / optimality criterion, have been proposed in [101], [100], and [109]. Maximum A Posterior (MAP) estimator-based distributed approaches for CL, where the robots periodically exchange local information, are presented in [108], [34]. While the above algorithms are designed for applications where robots can communicate analog measurements, in this work we focus on CL under *severe* communication constraints (e.g., underwater) where each robot can transmit *only a few bits* per analog measurement. In such applications, every robot is forced to carry out *lossy quantization* of its measurements and the commonly-used estimation frameworks for real-valued measurements (e.g., EKF or MAP) have to be modified to accommodate

[1] Sensors sample a process and provide a measurement which is often represented in digital form using 32 or 64 bit floating-point number representation. We refer to such measurements as analog.

these quantized measurements. Estimation with quantized observations has been well-studied in the signal processing community for wireless sensor networks. While there exists a large body of work on parameter estimation (either deterministic [114, 115, 116] or random variable [117, 118, 119]), we will focus on approaches that were developed to estimate random processes, as is the case in CL. The Sign-of-Innovation Kalman filter (SOI-KF), for estimating stochastic, dynamic processes, has been proposed [44], where the measurement innovation[2] , instead of the actual analog measurement, is quantized to a single bit. The SOI-KF is derived for linear, Gaussian process and measurement models and by approximating the posterior probability density function (pdf) by a Gaussian[3] after each measurement update, its state / covariance update equations have structure very similar to that of the standard Kalman filter (KF) [98]. Moreover, even using a single bit, its performance is 67% of that of the standard KF. When $f \geq 1$ bits are available for quantization, the SOI-KF approach has been extended in [120, 39, 121] to the batch-quantized KF (BQKF) and the iteratively-quantized KF (IQKF), where with $f = 4$ bits, the performance is almost indistinguishable from that of the standard KF. An extension of this approach to batch MAP estimation, for single bit (QMAP) and multiple bits (iteratively quantized IQ-MAP), is presented in [122, 111, 45].

All the quantized innovation filters discussed above use the multi-centralized (MC) architecture for CL, that is robust to single-point failures. In multi-centralized CL (MC-CL), each robot broadcasts *all* its measurements and every robot locally processes measurements from the entire team to generate pose estimates for all the robots. However, the quantization rules used by the filters depend upon these pose estimates, via the measurement innovation, and thus all robots have to maintain *identical* filters to ensure estimation consistency. Hence, each robot is forced to use quantized versions of its *own* locally-available, analog measurements for estimation, thus discarding valuable information that can be used to improve localization accuracy.

To overcome this drawback, in this chapter we introduce a novel hybrid estimation

[2] Measurement innovation is the difference between the actual and the estimated (by the filter) measurement.

[3] Note that due to the nonlinearity of the quantization operation, the posterior pdf is not, in general, a Gaussian.

framework that enables each robot to incorporate its locally-available, analog measurements in the estimation process. To achieve this, each robot maintains two local estimators (see Fig. 6.1): (i) a quantized (Q) estimator that processes quantized measurements from all robots, including itself, and (ii) a hybrid (H) estimator that processes its own analog measurements along with the quantized measurements from other robots in the team. For the proposed hybrid estimation framework, we derive H-estimators that can handle the *general* case of time-varying communication-bandwidth availability, i.e., when robots in the team can communicate $f \geq 1$ bits, per analog measurement. Specifically, we develop H-estimators for two quantization scenarios: (i) Batch quantization: where the bandwidth availability is known beforehand, and (ii) Iterative quantization: where additional bandwidth becomes available on-the-fly. For both these scenarios, we derive Minimum Mean Squared Error (MMSE) H-estimators (H-BQKF and H-IQKF, respectively) that are capable of processing local analog measurements along with multiple bits per remote analog measurement. Thus, we demonstrate that the hybrid estimation framework can optimally utilize additional bandwidth, whenever available, in order to improve the estimation accuracy of CL. The performance and accuracy of the proposed H-estimators is extensively tested in both simulations and experiment.

6.2 Problem Formulation

The proposed hybrid MMSE estimators are designed for wireless sensor networks where (i) the process and measurement models are shared *a priori* by all sensor nodes, and (ii) each sensor node can communicate with the network at every time step. While we now proceed with the specific application of CL, we note that the proposed estimators are general and can be used for any static / mobile sensor network applications that satisfy the above assumptions.

For CL, the problem setup consists of a team of N robots performing multi-centralized CL (MC-CL) in 2D. In MC-CL, each robot broadcasts *all* its measurements and every robot locally processes measurements from the entire team to generate pose estimates for all the robots. The process model for the multi-robot team is given by the following linear, discrete-time, dynamic system:

$$\mathbf{x}_k = \mathbf{F}_{k-1}\mathbf{x}_{k-1} + \mathbf{G}_{k-1}\mathbf{w}_{k-1},\ \mathbf{x}_0 \sim \mathcal{N}(\mathbf{x}_{init}, \mathbf{P_0}) \tag{6.1}$$

where, $\mathbf{w}_k$ is the zero-mean, white, Gaussian, and uncorrelated system noise at time-step k with covariance $E\left[\mathbf{w}_k {\mathbf{w}_l}^T\right] = \delta_{kl}\mathbf{Q}_k$. Here, $\mathbf{x}_k = [{\mathbf{x}_k^1}^T, {\mathbf{x}_k^2}^T, \ldots, {\mathbf{x}_k^N}^T]^T$, is the joint-state of the team and $\mathbf{x}_k^i = [x_k^i, y_k^i, \phi_k^i]^T$, is the state (position and orientation) of robot i at time-step k. This formulation uses, as the process model, a statistical motion model (e.g., the constant-velocity model [42]), driven only by the system noise [see (6.1)]. This enables us to treat both proprioceptive and exteroceptive measurements identically and use them for updates in the filter.

We assume that robot i obtains M_k^i scalar, analog measurements (proprioceptive and exteroceptive) at time-step k. The measurement model for robot i, $i = 1, \ldots, N$, is

$$z_{km}^i = {\mathbf{h}_{km}^i}^T \mathbf{x}_k + v_{km}^i, \ m = 1, \ldots, M_k^i \tag{6.2}$$

where v_{km}^i is zero-mean, white, Gaussian, and uncorrelated measurement noise with $E[v_{km}^i v_{ln}^i] = \delta_{kl,mn}{\sigma_{km}^i}^2$ and $E[v_{km}^i v_{ln}^j] = 0, \ \forall j \neq i, \ j = 1, \ldots, N$.

The linear models (6.1)-(6.2) are used only to simplify the mathematical derivations. In real-world scenarios, the *linearized* system, obtained from the underlying *non-linear* process and measurement models, will be used. Moreover, a single vector-valued measurement can be decomposed into multiple scalar measurements using pre-whitening [98] and then processed using the above formulation. Lastly, in order to improve the clarity and simplify the notation of the material presented in this chapter, from now onwards, we assume that each robot i obtains only a single, scalar, analog measurement, z_k^i, at time-step k. The generalization to M_k^i measurements is straightforward.

6.2.1 Real vs. Quantized Measurements

In the absence of communication-bandwidth constraints, the Minimum Mean Squared Error (MMSE) estimate of the robots' poses at time-step k, $\hat{\mathbf{x}}_{k|k}$, given *all analog* measurements up to time-step k, $\mathbf{z}_{0:k}$, is calculated as

$$\hat{\mathbf{x}}_{k|k} = E[\mathbf{x}_k|\mathbf{z}_{0:k}] = \int_{\mathbb{R}^r} \mathbf{x}_k p[\mathbf{x}_k|\mathbf{z}_{0:k}] d\mathbf{x}_k, \quad \mathbf{x}_k \in \mathbb{R}^r. \tag{6.3}$$

Here, $r = 3N$ and $\mathbf{z}_{0:k} = \left[(\mathbf{z}_{0:k}^1)^T, \ldots, \ (\mathbf{z}_{0:k}^N)^T\right]^T$ where, $\mathbf{z}_{0:k}^i = [z_0^i, \ldots, z_k^i]^T$, $i = 1, \ldots, N$. Under the Gaussian noise assumption for the linear system (6.1)-(6.2), the prior pdf, $p[\mathbf{x}_k|\mathbf{z}_{0:k-1}]$, and the posterior pdf, $p[\mathbf{x}_k|\mathbf{z}_{0:k}]$, are completely characterized

by their mean and covariance and can be obtained using the standard KF (see Algorithm 4). However, note that the real-valued measurement, z_k^i, is required for the state update in the KF and has to be broadcasted by robot i to the entire team.

Algorithm 4 Kalman Filter (KF)

1: KF Propagation

$$\hat{\mathbf{x}}_{k|k-1} = \mathbf{F}_{k-1}\hat{\mathbf{x}}_{k-1|k-1} \triangleq \hat{\mathbf{x}}_{k|k,0}$$
$$\mathbf{P}_{k|k-1} = \mathbf{F}_{k-1}\mathbf{P}_{k-1|k-1}\mathbf{F}_{k-1}^T + \mathbf{G}_{k-1}\mathbf{Q}_{k-1}\mathbf{G}_{k-1}^T \triangleq \mathbf{P}_{k|k,0}$$

2: KF Measurement Update
for $i = 1$ to N **do**

$$\hat{\mathbf{x}}_{k|k,i} = \hat{\mathbf{x}}_{k|k,i-1} + \frac{\mathbf{P}_{k|k,i-1}\mathbf{h}_k^i(z_k^i - \mathbf{h}_k^{i^T}\hat{\mathbf{x}}_{k|k,i-1})}{\mathbf{h}_k^{i^T}\mathbf{P}_{k|k,i-1}\mathbf{h}_k^i + \sigma_k^{i^2}}$$
$$\mathbf{P}_{k|k,i} = \mathbf{P}_{k|k,i-1} - \frac{\mathbf{P}_{k|k,i-1}\mathbf{h}_k^i\mathbf{h}_k^{i^T}\mathbf{P}_{k|k,i-1}}{\mathbf{h}_k^{i^T}\mathbf{P}_{k|k,i-1}\mathbf{h}_k^i + \sigma_k^{i^2}}$$

end
$\hat{\mathbf{x}}_{k|k} := \hat{\mathbf{x}}_{k|k,N}, \mathbf{P}_{k|k} := \mathbf{P}_{k|k,N}$

On the contrary, consider the bandwidth-limited scenario where each robot can communicate only $f \geq 1$ bits per analog measurement. Therefore, robot i must quantize its analog measurement, $z_k^i \in \mathbb{R}$, to $b_k^i \in \mathcal{B}$, $\mathcal{B} := \{1, \ldots, 2^f\}$ using a quantization rule $\mathbf{q}[\cdot]$ of the form

$$b_k^i = \mathbf{q}[z_k^i], \text{ where } \mathbf{q} : \mathbb{R} \to \mathcal{B}. \tag{6.4}$$

The MMSE estimate of the robots' poses, $\hat{\mathbf{x}}_{k|k}^Q$, given quantized measurements, $\mathbf{b}_{0:k}$, from all robots up to time-step k is calculated as

$$\hat{\mathbf{x}}_{k|k}^Q = E[\mathbf{x}_k|\mathbf{b}_{0:k}] = \int_{\mathbb{R}^r} \mathbf{x}_k p[\mathbf{x}_k|\mathbf{b}_{0:k}]d\mathbf{x}_k, \quad \mathbf{x}_k \in \mathbb{R}^r \tag{6.5}$$

However, due to the nonlinear quantization operation in (6.4), the prior and posterior pdfs are no longer Gaussian, even for the linear process and measurement models of (6.1)-(6.2). Moreover, computing these pdfs is, in general, intractable.

The BQKF and IQKF [39] (see Proposition 1), that quantize the measurement innovation, $z_k^i - \mathbf{h}_k^{i^T}\hat{\mathbf{x}}_{k|k,i-1}^Q$, simplify this computation by *approximating* these pdfs

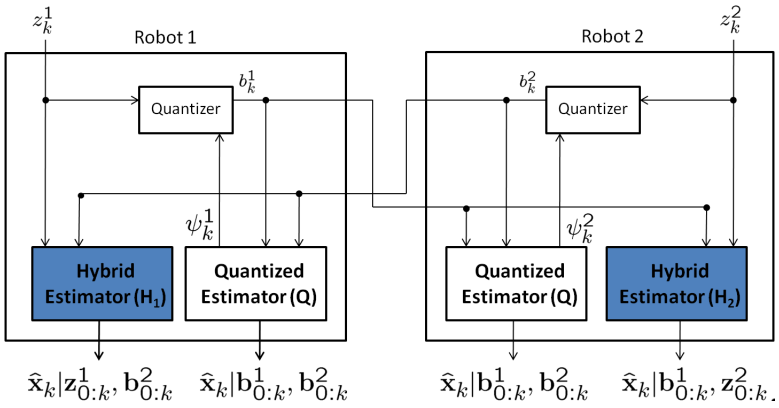


Figure 6.1: Hybrid Estimation framework. When $\psi_k^i = E[z_k^i|\mathbf{b}_{0:k-1}]$ and we use the quantization rule in (6.6), the H- and Q-estimators correspond to the H-BQKF and BQKF, respectively. When $\psi_k^i = E[z_k^i|\mathbf{b}_{0:k-1}, \mathbf{b}_k^{i(1:p-1)}]$ and we use the quantization rule in (6.12), the H- and Q-estimators correspond to the H-IQKF and IQKF, respectively, $i = 1, 2$.

as Gaussians (similar to the EKF[4]). The resulting estimators have structures very similar to that of the KF. However, a shortcoming of these filters is that even though each robot i has access to its own analog measurements, $\mathbf{z}_{0:k}^i$, the quantized-innovation filters force it to discard this information and process only the corresponding quantized measurements, $\mathbf{b}_{0:k}^i$. This is because the quantized measurement, b_k^i, depends upon the predicted measurement, $\mathbf{h}_k^{i^T} \hat{\mathbf{x}}_{k|k,i-1}^Q$, which has to be identical for all robots to ensure consistent estimation. This in turn implies that all robots have to process the same set of measurements, so that they can generate identical state estimates, $\hat{\mathbf{x}}_{k|k,i-1}^Q$. In Fig. 6.1, when the shaded Hybrid Estimators do not exist, we obtain the BQKF and IQKF.

6.3 Hybrid Estimation Framework

In this section, we present the hybrid estimation framework that enables each robot to process local analog and remote quantized measurements in order to improve the estimation accuracy of MC-CL. To achieve this, each robot i maintains two estimators (see

[4] It is important to note that this loss of Gaussianity in the EKF is due to the non-linearity of the process and measurement models as opposed to the non-linearity of the quantization step that we discuss here.

Fig. 6.1): (1) a quantized (Q) estimator that processes quantized measurements from all robots including itself, i.e., $\hat{\mathbf{x}}^{Q}_{k|k} = E[\mathbf{x}_k|\mathbf{b}^{q\neq i}_{0:k}, \mathbf{b}^{i}_{0:k}]$, and (2) a hybrid (H) estimator that processes its own real-valued measurements and quantized measurements from the other robots in the team, i.e., $\hat{\mathbf{x}}^{H_i}_{k|k} = E[\mathbf{x}_k|\mathbf{b}^{q\neq i}_{0:k}, \mathbf{z}^{i}_{0:k}]$, $q = 1, \ldots, N$. The estimates generated by the Q-estimator are identical for all robots since each robot processes identical measurements, $\mathbf{b}_{0:k} = \left\{\mathbf{b}^{q\neq i}_{0:k}, \mathbf{b}^{i}_{0:k}\right\}$. Therefore, they are used in the hybrid estimation framework for generating identical quantization thresholds for all robots, as will be shown in the next section. On the contrary, the estimates generated by the H-estimator are *different* for each robot, since each robot processes a different set of quantized and analog measurements.

In the next section we derive MMSE H-estimators that process: (i) local analog measurements, and (ii) multiple bits ($f \geq 1$), per analog measurement, communicated by other robots in the team. Specifically, we derive H-estimators for two quantization scenarios: (1) the H-BQKF for batch quantization, where the bandwidth availability (f bits per analog measurement) is known *a priori*, and (2) H-IQKF for iterative quantization, when additional bandwidth becomes available on-the-fly and the analog measurement is quantized iteratively, communicating additional bits for the same measurement as more bandwidth becomes available.

6.3.1 Batch Quantization

Encoding rule (Quantizer design)

Since robot j is pre-informed about the availability of $f \geq 1$ bits for communicating its analog measurement $z^{j}_{k} \in \mathbb{R}$, robot j partitions the observation space $\mathbb{R}$ into 2^f intervals. The interval $\mathcal{R}^{j}_{k}(n) := [\tau^{j}_{k}(n), \tau^{j}_{k}(n+1))$, where $\tau^{j}_{k}(n)$ are the quantization thresholds, $n \in \mathcal{B} := \{1, \ldots, 2^f\}$, $\tau^{j}_{k}(1) = -\infty$, $\tau^{j}_{k}(2^f + 1) = \infty$, and $\tau^{j}_{k}(n) < \tau^{j}_{k}(n+1)$. The quantization rule, which is based on the measurement innovation, has the form[5] [39]:

[5] We assume a round-robin scheduling algorithm where the quantized measurements are generated and processed sequentially, based on robot ids. Therefore, robot i generates and communicates its quantized measurement, b^{i}_{k}, before robot $(i+1)$. Moreover, all robots in the team process b^{i}_{k} to obtain $\hat{\mathbf{x}}^{Q}_{k|k,i}$, before robot $(i+1)$ generates its quantized measurement.

$$b_k^j = n, \text{ iff } (z_k^j - E[z_k^j|\mathbf{b}_{0:k-1}, \mathbf{b}_k^m]) \in [\tau_k^j(n), \tau_k^j(n+1)) \tag{6.6}$$

where, $\mathbf{b}_{0:k-1}$ denotes the quantized measurements from all robots up to time-step $k-1$, and $\mathbf{b}_k^m$ denotes the quantized measurements from robot m, $m = 1, \ldots, (j-1)$, at time-step k. From (6.2), the predicted measurement is $E[z_k^j|\mathbf{b}_{0:k-1}, \mathbf{b}_k^m] = \mathbf{h}_k^{j^T} \hat{\mathbf{x}}_{k|k,j-1}^Q$. Note that robot j uses the Q-estimator's predicted measurement, that is identical for all robots, for generating the quantized measurements. This allows other robots to correctly process / decode the quantized measurement. Robot j does *not* use the predicted measurement of its H-estimator, $E[z_k^j|\mathbf{b}_{0:k-1}^{q\neq j}, \mathbf{b}_k^m, \mathbf{z}_{0:k}^j]$, $q = 1, \ldots, N$, since it depends upon its local analog measurements, $\mathbf{z}_{0:k}^j$, which are unavailable to the other robots.

Decoding rule (Estimator design)

For the batch quantization rule from (6.6), we now derive the resulting MMSE Q- and H-estimators for the hybrid estimation framework. Note that the Q-estimator, *by definition*, is identical to the BQKF in [39]. It is presented here for completeness and to compare / contrast its structure with that of the H-estimator that we derive next.

Proposition 1. Q-estimator(BQKF)
Consider the linear model of (6.1)-(6.2) *and the quantization rule in* (6.6)*. If robot i assumes pdf $p\left[\mathbf{x}_{k-1}|\mathbf{b}_{0:k-1}\right] \sim \mathcal{N}\left(\hat{\mathbf{x}}_{k-1|k-1}^Q, \mathbf{P}_{k-1|k-1}^Q\right)$, then the state / covariance propagation equations are identical to the KF (see Algorithm 4). Define $\sigma_Q^2 = \mathbf{h}_k^{j^T} \mathbf{P}_{k|k,j-1}^Q \mathbf{h}_k^j + \sigma_k^{j^2}$ and the normalized thresholds $\Delta_Q(n) := \tau_k^j(n)/\sigma_Q$. If robot i assumes the pdf $p\left[\mathbf{x}_k|\mathbf{b}_{0:k-1}, \mathbf{b}_k^m\right] \sim \mathcal{N}\left(\hat{\mathbf{x}}_{k|k,j-1}^Q, \mathbf{P}_{k|k,j-1}^Q\right)$, $m = 1, \ldots, (j-1)$, then for the MMSE estimator for robot i, that processes the quantized measurement, b_k^j, $j \neq i$, from robot j, the state / covariance update equations are given by*

$$\hat{\mathbf{x}}_{k|k,j}^Q = \hat{\mathbf{x}}_{k|k,j-1}^Q + \alpha_Q(n) \frac{\mathbf{P}_{k|k,j-1}^Q \mathbf{h}_k^j}{\sqrt{\mathbf{h}_k^{j^T} \mathbf{P}_{k|k,j-1}^Q \mathbf{h}_k^j + \sigma_k^{j^2}}} \tag{6.7}$$

$$\mathbf{P}_{k|k,j}^Q = \mathbf{P}_{k|k,j-1}^Q - \beta_Q(n) \frac{\mathbf{P}_{k|k,j-1}^Q \mathbf{h}_k^j \mathbf{h}_k^{j^T} \mathbf{P}_{k|k,j-1}^Q}{\mathbf{h}_k^{j^T} \mathbf{P}_{k|k,j-1}^Q \mathbf{h}_k^j + \sigma_k^{j^2}} \tag{6.8}$$

where

$$\alpha_Q(n) = \frac{1}{\sqrt{2\pi}} \frac{exp\left[-\Delta_Q^2(n)/2\right] - exp\left[-\Delta_Q^2(n+1)/2\right]}{Q\left[\Delta_Q(n)\right] - Q\left[\Delta_Q(n+1)\right]}$$

$$\beta_Q(n) = \alpha_Q^2(n) - \frac{1}{\sqrt{2\pi}} \times \frac{\Delta_Q(n) exp\left[-\Delta_Q^2(n)/2\right] - \Delta_Q(n+1) exp\left[-\Delta_Q^2(n+1)/2\right]}{Q\left[\Delta_Q(n)\right] - Q\left[\Delta_Q(n+1)\right]}$$

and $0 < \beta_Q(n) < 1$.

Here, $\mathrm{Q}[x] = \int_x^\infty \frac{1}{\sqrt{2\pi}} \exp(-u^2/2) du$, is the Gaussian tail probability. The proof for Proposition 1 can be found in [39].

Proposition 2. H-estimator (H-BQKF)

Consider the linear model of (6.1)-(6.2) *and the quantization rule in* (6.6)*. If robot* i *assumes the pdf* $p\left[\mathbf{x}_{k-1} | \mathbf{b}_{0:k-1}^{q \neq i}, \mathbf{z}_{0:k-1}^{i}\right] \sim \mathcal{N}\left(\hat{\mathbf{x}}_{k-1|k-1}^{H_i}, \mathbf{P}_{k-1|k-1}^{H_i}\right)$*, then the state / covariance propagation equations are identical to the KF (see Algorithm 4). If robot* i*, assumes the pdf* $p\left[\mathbf{x}_k | \mathbf{b}_{0:k-1}^{q \neq i}, \mathbf{b}_k^{m \neq i}, \mathbf{z}_{0:k-1}^{i}\right] \sim \mathcal{N}\left(\hat{\mathbf{x}}_{k|k,j-1}^{H_i}, \mathbf{P}_{k|k,j-1}^{H_i}\right)$*, then the MMSE estimator for robot* i *processing its* own *analog measurement,* z_k^i*, is identical to the KF (see Algorithm 4).*

For the MMSE estimator processing the quantized measurement b_k^j *from robot* j*,* $j \neq i$*, the state / covariance update equations are given by*

$$\hat{\mathbf{x}}_{k|k,j}^{H_i} = \hat{\mathbf{x}}_{k|k,j-1}^{H_i} + \alpha_{H_i}(n) \frac{\mathbf{P}_{k|k,j-1}^{H_i} \mathbf{h}_k^j}{\sqrt{\mathbf{h}_k^{j^T} \mathbf{P}_{k|k,j-1}^{H_i} \mathbf{h}_k^j + \sigma_k^{j^2}}} \tag{6.9}$$

$$\mathbf{P}_{k|k,j}^{H_i} = \mathbf{P}_{k|k,j-1}^{H_i} - \beta_{H_i}(n) \frac{\mathbf{P}_{k|k,j-1}^{H_i} \mathbf{h}_k^j \mathbf{h}_k^{j^T} \mathbf{P}_{k|k,j-1}^{H_i}}{\mathbf{h}_k^{j^T} \mathbf{P}_{k|k,j-1}^{H_i} \mathbf{h}_k^j + \sigma_k^{j^2}} \tag{6.10}$$

where

$$\alpha_{H_i}(n) = \frac{1}{\sqrt{2\pi}} \frac{exp\left[-\Delta_{H_i}^2(n)/2\right] - exp\left[-\Delta_{H_i}^2(n+1)/2\right]}{Q\left[\Delta_{H_i}(n)\right] - Q\left[\Delta_{H_i}(n+1)\right]}$$

$$\beta_{H_i}(n) = \alpha_{H_i}^2(n) - \frac{1}{\sqrt{2\pi}} \times \frac{\Delta_{H_i}(n) exp\left[-\Delta_{H_i}^2(n)/2\right] - \Delta_{H_i}(n+1) exp\left[-\Delta_{H_i}^2(n+1)/2\right]}{Q\left[\Delta_{H_i}(n)\right] - Q\left[\Delta_{H_i}(n+1)\right]}$$

$$\Delta_{H_i}(n) = \frac{(\tau_k^j(n) - \mathbf{h}_k^{j^T}(\hat{\mathbf{x}}_{k|k,j-1}^{H_i} - \hat{\mathbf{x}}_{k|k,j-1}^{Q}))}{\sigma_{H_i}}$$

$$\sigma_{H_i}^2 = \mathbf{h}_k^{j^T} \mathbf{P}_{k|k,j-1}^{H_i} \mathbf{h}_k^j + \sigma_k^{j^2}$$

and $0 < \beta_{H_i}(n) < 1$.

Proof. The proof of Proposition 2 can be found in Appendix B.1. □

Quantization thresholds

We now discuss the selection of optimal quantization thresholds, $\tau_k^j(n)$, $n = 2, \ldots, 2^f$, for the batch quantized hybrid estimation framework. For the team to optimally and correctly process quantized measurements received from robot j, robot j's quantization thresholds should be known to the team. Therefore, these thresholds have to selected based on a metric that is common to the team. We choose robot j's thresholds so as to maximize the average reduction in the covariance of the BQKF [see (6.8)]. The Q-estimator, BQKF, is identical for all robots, and therefore, every robot in the team can locally calculate the quantization thresholds used by robot j. The threshold selection is formulated as an optimization problem of the form [39]:

$$\{\Delta_Q^*(n)\}_{n=2}^{2^f} := \arg \max_{\{\Delta_Q(n)\}_{n=2}^{2^f}} E[\beta_Q(n)|\mathbf{b}_{0:k-1}, \mathbf{b}_k^m] \tag{6.11}$$

where the expectation is with respect to the $Pr\{b_k^j|\mathbf{b}_{0:k-1}, \mathbf{b}_k^m\}$, where $Pr\{\cdot\}$ is the probability of the event of interest. Here, maximizing the average covariance reduction of the BQKF is equivalent to maximizing the expected value of $\beta_Q(n)$. Moreover, the optimization problem in (6.11) is equivalent to quantizing the measurement innovation, $z_k^j - \mathbf{h}_k^{j^T} \hat{\mathbf{x}}_{k|k,j-1}^Q$, with minimum MSE distortion [39]. The solution to (6.11) is the well-known Lloyd-Max quantizer and the corresponding values for the optimal quantization thresholds can be found in [123], [124].

Before proceeding, we make the following important observations about the proposed H-BQKF:

1. As seen from Proposition 2, even though robot i cannot communicate its analog measurement, z_k^i, to the team, the H-BQKF enables it to *optimally* process this measurement locally using the KF.

2. Define $\bar{\alpha}_Q(n) = \alpha_Q(n)\sqrt{\mathbf{h}_k^{j^T} \mathbf{P}_{k|k,j-1}^Q \mathbf{h}_k^j + \sigma_k^{j^2}}$ and $\bar{\alpha}_{H_i}(n) = \alpha_{H_i}(n)\sqrt{\mathbf{h}_k^{j^T} \mathbf{P}_{k|k,j-1}^{H_i} \mathbf{h}_k^j + \sigma_k^{j^2}}$. Using these terms in (6.7) and (6.9), we see that the structure of the state update equation for quantized measurements, in BQKF and H-BQKF, is very similar to that of the KF. Moreover, as expected, the

measurement innovation in the state update equation of the KF is approximated by $\bar{\alpha}_Q(n)$ and $\bar{\alpha}_{H_i}(n)$ in the BQKF and H-BQKF, respectively.

3. Consider the covariance update equations [see (6.8) and (6.10)] for processing quantized measurements in the BQKF and H-BQKF. Their structure is identical to that of KF, except for the factors $\beta_Q(n)$ and $\beta_{H_i}(n)$. Since $0 < \beta_Q(n), \beta_{H_i}(n) < 1$, the covariance reduction for these estimators will always be less than that of the KF since information is discarded during quantization.

4. While processing quantized measurements, the state / covariance update equations for the H-BQKF [see (6.9) and (6.10)] are a function of the difference between the predicted measurements, $\mathbf{h}_k^{j^T} \hat{\mathbf{x}}_{k|k,j-1}^{H_i}$ and $\mathbf{h}_k^{j^T} \hat{\mathbf{x}}_{k|k,j-1}^{Q}$, of the H-BQKF and BQKF respectively. Moreover, the covariance reduction in (6.10) increases as the absolute value of this difference decreases. This is because the quantized measurements encode information about BQKF's measurement innovation. If the difference between the predicted measurements of the H-BQKF and BQKF is large, the quantized measurement will convey very little information to the H-BQKF.

5. Lastly, note by choosing $f = 1$ and substituting the corresponding optimal thresholds $\tau_k^j(1) = -\infty$, $\tau_k^j(2) = 0$, and $\tau_k^j(3) = \infty$ in Propositions 1 and 2, we obtain the special case of the single bit Q- and H-estimators, the SOI-KF [44] and H-SOI-KF [125], respectively.

6.3.2 Iterative Quantization

Encoding rule (Quantizer design)

We now consider the quantization scenario where additional communication bandwidth becomes available to the robots on-the-fly. Therefore, robot j can now communicate extra bits, *one bit at a time*, for the same analog measurement z_k^j. Assume that robot j has communicated $(p-1)$ bits, $\mathbf{b}_k^{j(1:p-1)}$, $p \geq 1$, for the analog measurement z_k^j. Robot j generates the p-th bit, $b_k^{j(p)}$, using the following quantization rule

$$b_k^{j(p)} := \text{sign}[z_k^j - E[z_k^j | \mathbf{b}_{0:k-1}, \mathbf{b}_k^m, \mathbf{b}_k^{j(1:p-1)}]] \tag{6.12}$$

where, $\mathbf{b}_{0:k-1}$ denotes the quantized bits from all robots up to time-step $k-1$, and $\mathbf{b}_k^m$ denotes the quantized measurements from robot m, $m = 1, \ldots, (j-1)$, at time-step k. The expected measurement is given by

$$\begin{aligned} E[z_k^j|\mathbf{b}_{0:k-1}, \mathbf{b}_k^m, \mathbf{b}_k^{j(1:p-1)}] &= E[\mathbf{h}_k^{j^T}\mathbf{x}_k + v_k^j|\mathbf{b}_{0:k-1}, \mathbf{b}_k^m, \mathbf{b}_k^{j(1:p-1)}] \\ &= \mathbf{h}_k^{j^T} E[\mathbf{x}_k|\mathbf{b}_{0:k-1}, \mathbf{b}_k^m, \mathbf{b}_k^{j(1:p-1)}] + E[v_k^j|\mathbf{b}_{0:k-1}, \mathbf{b}_k^m, \mathbf{b}_k^{j(1:p-1)}] \\ &= \mathbf{h}_k^{j^T}\hat{\mathbf{x}}_{k|k,j}^{Q(p-1)} + E[v_k^j|\mathbf{b}_{0:k-1}, \mathbf{b}_k^m, \mathbf{b}_k^{j(1:p-1)}]. \end{aligned} \tag{6.13}$$

Importantly, note that in the above equation, the term $E[v_k^j|\mathbf{b}_{0:k-1}, \mathbf{b}_k^m, \mathbf{b}_k^{j(1:p-1)}] \neq 0$, unless $p = 1$. If $p > 1$, the measurement noise, v_k^j, is no longer independent of the previous bits, $\mathbf{b}_k^{j(1:p-1)}$, since they were generated using the noisy analog measurement, z_k^j, itself. Therefore, we need the MMSE estimates of the noise term so as to correctly generate / decode the bits. These estimates can be obtained by augmenting the state, $\mathbf{x}_k$, with the noise term, v_k^j, and considering the augmented state vector $\breve{\mathbf{x}}_k = [\mathbf{x}_k^T, v_k^j]^T$ and a modified $\breve{\mathbf{h}}_k^j = [\mathbf{h}_k^j, 1]^T$. With these changes, the measurement model from (6.2) can be rewritten as [39]:

$$z_k^j = \breve{\mathbf{h}}_k^{j^T}\breve{\mathbf{x}}_k. \tag{6.14}$$

Similarly, the process model in (6.1) can also be expressed using the augmented state as follows

$$\breve{\mathbf{x}}_k = \breve{\mathbf{F}}_{k-1}\breve{\mathbf{x}}_{k-1} + \breve{\mathbf{G}}_{k-1}\breve{\mathbf{w}}_{k-1} \tag{6.15}$$

$$\text{where } \breve{\mathbf{F}}_{k-1} := \begin{bmatrix} \mathbf{F}_{k-1} & \mathbf{0} \\ \mathbf{0}^T & 0 \end{bmatrix},\ \breve{\mathbf{G}}_{k-1} := \begin{bmatrix} \mathbf{G}_{k-1} & \mathbf{0} \\ \mathbf{0}^T & 1 \end{bmatrix},$$

$$\breve{\mathbf{w}}_{k-1} := [\mathbf{w}_{k-1}^T, v_k^j]^T,$$

$$\breve{\mathbf{Q}}_{k-1} := E[\breve{\mathbf{w}}_{k-1}\breve{\mathbf{w}}_{k-1}^T] = \begin{bmatrix} \mathbf{Q}_{k-1} & \mathbf{0} \\ \mathbf{0}^T & \sigma_k^{j^2} \end{bmatrix}.$$

Thus, every time robot i generates its own iteratively quantized measurements or processes iteratively quantized bits communicated by other robots, it has to augment its own state vector with the corresponding measurement noise so that the noise statistics can be correctly estimated. Then the quantization process in (6.12) becomes identical to that of the sign-of-innovation quantization rule from [44].

Decoding rule (Estimator design)

For the iterative quantization rule in (6.12), we now derive the resulting Q- and H-estimators for the hybrid estimation framework. Note that the Q-estimator, *by definition*, is identical to the IQKF presented in [39] and is not presented here. The H-estimator, H-IQKF, is obtained as follows.

Proposition 3. H-estimator (H-IQKF)
Consider the linear model of (6.15)-(6.14) *and the quantization rule in* (6.12)*. If robot i assumes the pdf* $p\left[\breve{\mathbf{x}}_{k-1}|\mathbf{b}_{0:k-1}^{q\neq i},\mathbf{z}_{0:k-1}^{i}\right]\sim\mathcal{N}\left(\hat{\breve{\mathbf{x}}}_{k-1|k-1}^{H_i},\breve{\mathbf{P}}_{k-1|k-1}^{H_i}\right)$*, the state / covariance propagation equations are given by*

$$\hat{\breve{\mathbf{x}}}_{k|k-1}^{H_i}=\breve{\mathbf{F}}_{k-1}\hat{\breve{\mathbf{x}}}_{k-1|k-1}^{H_i} \tag{6.16}$$

$$\breve{\mathbf{P}}_{k|k-1}^{H_i}=\breve{\mathbf{F}}_{k-1}\breve{\mathbf{P}}_{k-1|k-1}^{H_i}\breve{\mathbf{F}}_{k-1}^{T}+\breve{\mathbf{G}}_{k-1}\breve{\mathbf{Q}}_{k-1}\breve{\mathbf{G}}_{k-1}^{T}. \tag{6.17}$$

If robot i assumes the pdf $p\left[\breve{\mathbf{x}}_{k}|\mathbf{b}_{0:k-1}^{q\neq i},\mathbf{z}_{0:k-1}^{i},\mathbf{b}_{k}^{m\neq i},\mathbf{b}_{k}^{j(1:p-1)}\right]\sim\mathcal{N}\left(\hat{\breve{\mathbf{x}}}_{k|k,j}^{H_i(p-1)},\breve{\mathbf{P}}_{k|k,j}^{H_i(p-1)}\right)$*, then the MMSE estimator for robot i processing its* own *analog measurement,* z_k^i*, is identical to the Kalman filter (see Algorithm 4).*

For the MMSE estimator for robot i, that processes the quantized measurement, $b_k^{j(p)}$*,* $j\neq i$*, from robot j, the state / covariance update equations are given by*

$$\hat{\breve{\mathbf{x}}}_{k|k,j}^{H_i(p)}=\hat{\breve{\mathbf{x}}}_{k|k,j}^{H_i(p-1)}+\alpha\frac{\breve{\mathbf{P}}_{k|k,j}^{H_i(p-1)}\breve{\mathbf{h}}_k^j}{\sqrt{\breve{\mathbf{h}}_k^{j^T}\breve{\mathbf{P}}_{k|k,j}^{H_i(p-1)}\breve{\mathbf{h}}_k^j}}b_k^{j(p)} \tag{6.18}$$

$$\breve{\mathbf{P}}_{k|k,j}^{H_i(p)}=\breve{\mathbf{P}}_{k|k,j}^{H_i(p-1)}-\beta\frac{\breve{\mathbf{P}}_{k|k,j}^{H_i(p-1)}\breve{\mathbf{h}}_k^j\breve{\mathbf{h}}_k^{j^T}\breve{\mathbf{P}}_{k|k,j}^{H_i(p-1)}}{\breve{\mathbf{h}}_k^{j^T}\breve{\mathbf{P}}_{k|k,j}^{H_i(p-1)}\breve{\mathbf{h}}_k^j} \tag{6.19}$$

where,

$$\alpha=\frac{exp[-\Delta^2/2]}{\sqrt{2\pi}Q[-b_k^{j(p)}\Delta]},\quad \beta=\alpha^2+b_k^{j(p)}\frac{\Delta exp[-\Delta^2/2]}{\sqrt{2\pi}Q[-b_k^{j(p)}\Delta]}$$

$$\Delta=\frac{\breve{\mathbf{h}}_k^{j^T}\left(\hat{\breve{\mathbf{x}}}_{k|k,j}^{H_i(p-1)}-\hat{\breve{\mathbf{x}}}_{k|k,j}^{Q(p-1)}\right)}{\sqrt{\breve{\mathbf{h}}_k^{j^T}\breve{\mathbf{P}}_{k|k,j}^{H_i(p-1)}\breve{\mathbf{h}}_k^j}}$$

Proof. The proof of Proposition 3 can be found in Appendix B.2. □

Also, note that the average covariance reduction for the H-IQKF can be calculated as follows:

$$
\begin{aligned}
&E_{b_k^{j(p)}}[\beta(b_k^{j(p)})|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{z}_{0:k-1}^{i}, \mathbf{b}_k^{m\neq i}, \mathbf{b}_k^{j(1:p-1)}] \\
&\quad = \beta(b_k^{j(p)}=-1)\Pr\{b_k^{j(p)}=-1|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{z}_{0:k-1}^{i}, \mathbf{b}_k^{m\neq i}, \mathbf{b}_k^{j(1:p-1)}\} \\
&\qquad + \beta(b_k^{j(p)}=1)\Pr\{b_k^{j(p)}=1|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{z}_{0:k-1}^{i}, \mathbf{b}_k^{m\neq i}, \mathbf{b}_k^{j(1:p-1)}\} \\
&\quad = \beta(b_k^{j(p)}=-1)\mathrm{Q}[\Delta] + \beta(b_k^{j(p)}=1)\mathrm{Q}[-\Delta] \\
&\quad = \frac{\exp(-\Delta^2)}{2\pi \mathrm{Q}[\Delta]\mathrm{Q}[-\Delta]}
\end{aligned} \tag{6.20}
$$

In the above equation, the expectation is with respect to the iteratively quantized bit $b_k^{j(p)}$. The notation $\beta(b_k^{j(p)})$, indicates that β is a function of the bit $b_k^{j(p)}$. Note that the first $r = 3N$ elements of $\hat{\breve{\mathbf{x}}}_{k|k,j}^{Q(p)}$ and $\hat{\breve{\mathbf{x}}}_{k|k,j}^{H_i(p)}$ correspond to the robots' state estimates, and the top $r \times r$ sub-matrices of $\breve{\mathbf{P}}_{k|k,j}^{Q(p)}$ and $\breve{\mathbf{P}}_{k|k,j}^{H_i(p)}$ correspond to their covariance, respectively. Once all bits corresponding to z_k^j have been processed, the robots can revert to the original state vector, $\mathbf{x}_k$. When processing bits from a new analog measurement, the robots will again augment the state with the corresponding measurement noise and the above procedure will be repeated.

First, from Proposition 3, we see that even though robot i cannot communicate its analog measurement, z_k^i, to the team, the H-IQKF enables it to *optimally* process this measurement locally using the KF. Next, note that structures of the IQKF and H-IQKF, are strikingly similar to that of the single-bit SOI-KF [44] and H-SOIKF [125], respectively, where the analog measurement is quantized to a single bit. Specifically, as expected, when $p = 1$, the IQKF and H-IQKF are identical to SOI-KF and H-SOIKF, respectively. Moreover, the structure of the IQKF and H-IQKF is similar to that of the KF, and as expected, the covariance reduction of these quantized-innovation filters is smaller than that of the KF. Lastly, similar to the H-BQKF, when the H-IQKF processes quantized measurements, the state / covariance update equations are a function of the difference between the predicted measurements, $\breve{\mathbf{h}}_k^{j^T}\hat{\breve{\mathbf{x}}}_{k|k,j}^{H_i(p-1)}$ and $\breve{\mathbf{h}}_k^{j^T}\hat{\breve{\mathbf{x}}}_{k|k,j}^{Q(p-1)}$, of the H-IQKF and IQKF, respectively.

6.4 Simulations and Experiment

6.4.1 Simulation Results

The simulation set-up consists of a team of two robots navigating in 2D while performing MC-CL. The continuous-time dynamics for each robot are given by the constant velocity motion model [42]:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{G}_c \begin{bmatrix} w_v \\ w_\omega \end{bmatrix} \tag{6.21}$$

where $\mathbf{x} = [x,\ y,\ \phi,\ v,\ \omega]^T$, $\mathbf{f}(\mathbf{x}) = [v\cos\phi,\ v\sin\phi,\ \omega,\ 0,\ 0]^T$ and $\mathbf{G}_c = [\mathbf{0}_{2\times 3},\ \mathbf{I}_{2\times 2}]^T$. The standard deviation of the continuous-time noise in the linear, v, and rotational, ω, velocity is chosen to be $\sigma_v = 0.6325$ m/s.$\sqrt{\text{Hz}}$ and $\sigma_\omega = 0.4967$ rad/s.$\sqrt{\text{Hz}}$ respectively. Each robot obtains measurements for its linear, v_m, and rotational, ω_m, velocity, as well as its distance, d_m, and bearing, θ_m, to the other robot. The noise in these measurements is modeled as zero-mean, white Gaussian with standard deviation $\sigma_{v_m} = 0.07$ m/s, $\sigma_{\omega_m} = 0.28$ rad/s for the linear and rotational velocity measurements, respectively, and $\sigma_{d_m} = 0.05$ m, $\sigma_{\theta_m} = 0.09$ rad for the corresponding distance and bearing measurements.

In this section, we compare the performance of the proposed H-estimators (local analog and remote quantized), H-BQKF and H-IQKF with: (1) the Q-estimators (local and remote quantized measurements), BQKF and IQKF, using $1-4$ bits per analog measurement, and (2) the standard EKF that uses analog measurements from all robots and hence is our benchmark.

Figures 6.4.1 and 6.4.1 show the root mean squared error (RMSE) in the position and orientation estimates for these estimators, averaged over the 2 robots and 100 Monte Carlo trials. We have included the results for $n = \{1, 2, 3, 4\}$ bits. Since the estimates generated by the H-estimator are different for each robot, the RMSE for the H-estimators, H-BQKF and H-IQKF, are also averaged over estimators maintained by each robot. Table 6.1 presents the results for position and orientation RMSE, for $n = 1, 2, 4$ bits, averaged over the duration of the simulation run. Moreover, since the 1-bit iterative- and batch-quantized estimators are identical, Table 6.1 omits the results for the 1-bit iteratively quantized filters.

From Figures 2, 3, and Table 6.1, we observe that the estimates generated by the

proposed H-estimators, H-BQKF and H-IQKF, are more accurate that their Q-estimator counterparts, BQKF and IQKF, irrespective of the number of quantization bits (n = {1,2,3,4}) considered. Specifically, the 1-bit hybrid filters are 20% more accurate than the 1-bit quantized filters, while the 2-bit hybrid filters show a performance improvement of 13% over their quantized counterparts. Overall, the error in the estimates decreases as we increase the number of quantization bits and by communicating as few as 4 bits per analog measurement, both the H- and Q-estimators are able to achieve accuracy very close to that of the analog EKF. Also, for a fixed number of bits, the performance of both the batch and iteratively quantized estimators is comparable. Thus, we conclude that by including their local analog measurements in the estimation process, without any additional communication overhead, the robots are able to substantially improve the estimation accuracy of CL.

Table 6.1: Simulation results for $N = 2$ robots

	Pos. RMSE (m)	Orient. RMSE (rad)
BQKF (1 bit):	1.0743	0.1689
H-BQKF (1 bit):	0.8584	0.1361
BQKF (2 bit):	0.7203	0.1171
H-BQKF (2 bit):	0.6199	0.1018
BQKF (4 bit):	0.5515	0.0910
H-BQKF (4 bit):	0.5337	0.0885
IQKF (2 bit):	0.6932	0.1120
H-IQKF (2 bit):	0.6105	0.0997
IQKF (4 bit):	0.6176	0.1007
H-IQKF (4 bit):	0.5673	0.0934
EKF (analog):	0.5151	0.0858

6.4.2 Experimental Results

Experimental validation was carried out using a team of four Pioneer-I robots moving in a rectangular arena of 4 m $\times$ 2.5 m for approximately 16 minutes. An overhead camera is used to obtain the robots' poses in a global coordinate frame (ground truth).

The robots move with a constant velocity of 0.1 m/s while avoiding collisions with boundaries of the arena and other robots in the team. The robots obtain linear and

rotational velocity (odometry) measurements, and relative distance and bearing measurements at a frequency of 1 Hz. The noise standard deviations of the odometry measurements for the heterogeneous robot team vary from 0.0078 rad/s to 0.02 rad/s for rotational velocity, and from 0.0032 m/s to 0.0059 m/s for linear velocity. The relative distance and bearing measurements between the robots are generated synthetically using data from the overhead camera and adding Gaussian noise with standard deviation $\sigma_d = 0.05$ m for distance and $\sigma_\theta = 2$ deg for relative bearing.

Table 6.2 presents the position and orientation RMSE and the NEES for each robot, averaged over all time-steps. For the H-estimators, these quantities are also averaged across all robot's H-estimators. The experiment corroborates our simulation results. From the RMSE data in the table, we conclude that the n-bit, $n = \{1, 2, 3\}$, H-estimators (H-BQKF and H-IQKF) outperform the corresponding Q-estimators (BQKF and IQKF). Thus the H-estimators, by enabling robots to include their local analog measurements in the estimation process, significantly improve the estimation accuracy of CL. Specifically, the improvement in performance of the H-estimators over the Q-estimators is more pronounced for the $n = 1, 2$ bit scenario, while with $n = 3$ bits, the performance of both the Q- and H-estimators is very close to that of the standard analog EKF. For this particular experiment, we observe that in some cases, the RMSE for the 3-bit Q- and H-estimators is slightly lower than the EKF. However, these results are reported for a single experimental run, and in general, we would expect the analog EKF to outperform all the Q- and H-estimators. The table also reports the average NEES for each robot's pose (x-y position and orientation). The ideal value for the NEES should be 3 and from the table we see that both the Q- and H-estimators have consistent performance.

6.5 Summary

In this chapter, we presented hybrid Minimum Mean Squared Error estimators for wireless sensor networks with *time-varying* communication-bandwidth constraints, focusing on the particular application of multi-robot Cooperative Localization. When sensor nodes (e.g., robots) communicate only a quantized version of their analog measurements

to the team, our proposed hybrid filters enable robots to process all available information, i.e., local analog measurements (recorded by its *own* sensors) as well as remote quantized measurements (collected and communicated by *other* sensors). Moreover, these filters are *resource-aware* and can utilize additional bandwidth, whenever available, to maximize estimation accuracy. Specifically, in this work, we presented two filters, the Hybrid Batch-Quantized Kalman filter (H-BQKF) and the Hybrid Iteratively-Quantized Kalman filter (H-IQKF), that can process local analog measurements along with remote measurements quantized to *any number of bits*. We tested our proposed filters in simulations and experimentally, and demonstrated that they achieve performance comparable to the standard Kalman filter.

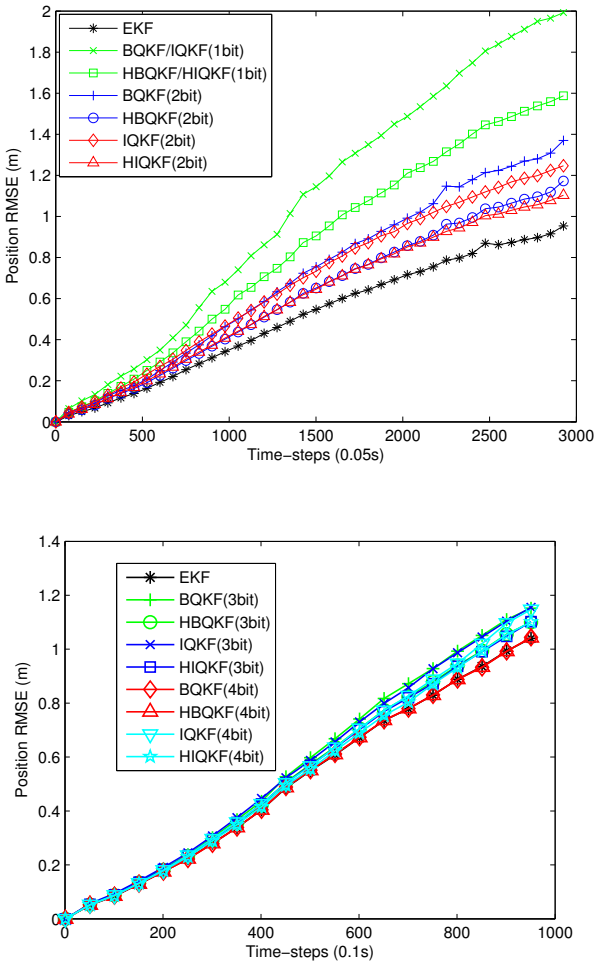


Figure 6.2: Comparison of position RMSE for EKF, and $1-4$ bit H-BQKF, H-IQKF, BQKF and IQKF.

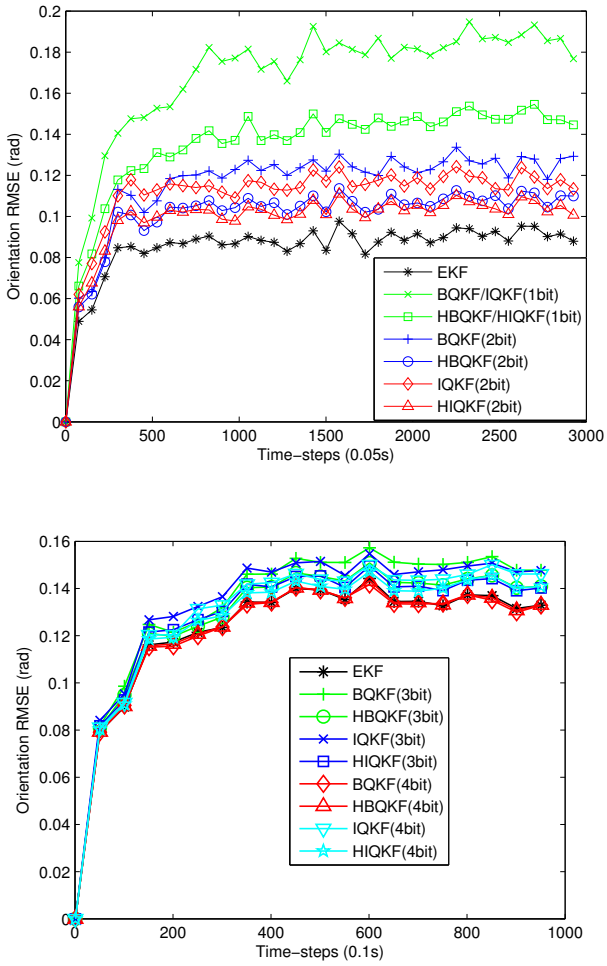


Figure 6.3: Comparison of orientation RMSE for EKF, and $1-4$ bit H-BQKF, H-IQKF, BQKF and IQKF.

Table 6.2: Experimental results for the hybrid estimation framework

RMS Position error (m)				
	Robot 1	Robot 2	Robot 3	Robot 4
BQKF (1 bit):	1.3643	1.1973	1.2849	1.1794
H-BQKF (1 bit):	1.0254	0.9358	0.9716	0.9714
BQKF (2 bit):	0.8654	0.7405	0.8383	0.8752
H-BQKF (2 bit):	0.7516	0.6652	0.7254	0.7542
BQKF (3 bit):	0.5330	0.4824	0.5168	0.4742
H-BQKF (3 bit):	0.5821	0.5553	0.5665	0.5809
IQKF (2 bit):	0.5990	0.5961	0.5758	0.7062
H-IQKF (2 bit):	0.6023	0.5879	0.5870	0.6637
IQKF (3 bit):	0.6255	0.6187	0.6291	0.7028
H-IQKF (3 bit):	0.5770	0.5547	0.5650	0.5974
EKF (analog):	0.5651	0.5385	0.5239	0.5719
RMS Orientation error (rad)				
	Robot 1	Robot 2	Robot 3	Robot 4
BQKF (1 bit):	0.7793	0.7810	0.7777	0.7820
H-BQKF (1 bit):	0.6331	0.6337	0.6289	0.6345
BQKF (2 bit):	0.6305	0.6314	0.6294	0.6296
H-BQKF (2 bit):	0.5392	0.5402	0.5386	0.5385
BQKF (3 bit):	0.3182	0.3174	0.3171	0.3176
H-BQKF (3 bit):	0.4064	0.4062	0.4059	0.4058
IQKF (2 bit):	0.4530	0.4549	0.4540	0.4543
H-IQKF (2 bit):	0.4300	0.4316	0.4306	0.4314
IQKF (3 bit):	0.4480	0.4468	0.4461	0.4465
H-IQKF (3 bit):	0.3812	0.3802	0.3795	0.3796
EKF (analog):	0.4055	0.4064	0.4059	0.4060
NEES for robot pose				
	Robot 1	Robot 2	Robot 3	Robot 4
BQKF (1 bit):	3.6846	4.3333	3.6520	3.9089
H-BQKF (1 bit):	3.2373	3.8466	3.4646	3.7814
BQKF (2 bit):	4.4646	4.6979	4.5131	4.6764
H-BQKF (2 bit):	4.2031	4.3621	4.2589	4.3843
BQKF (3 bit):	4.1651	4.1963	3.9965	4.2839
H-BQKF (3 bit):	4.4942	4.5439	4.3342	4.6524
IQKF (1 bit):	3.6846	4.3333	3.6520	3.9089
H-IQKF (1 bit):	3.2373	3.8466	3.4646	3.7814
IQKF (2 bit):	4.2233	4.1830	4.4816	4.6072
H-IQKF (2 bit):	4.0503	4.0098	4.2102	4.4643
IQKF (3 bit):	3.6569	3.6445	3.5527	3.9546
H-IQKF (3 bit):	3.3999	3.4098	3.3166	3.6531
EKF (analog):	4.7408	4.7551	4.6850	5.0168

Chapter 7

MAP-based Hybrid Estimation Framework for CL

In this chapter, we extend the hybrid estimation framework for CL, presented in the previous chapter, to the Maximum A Posteriori (MAP) estimator. For the nonlinear process and measurement models prevalent in robotics and the nonlinearities arising from measurement quantization, the proposed hybrid MAP estimator improves the accuracy of the robots' and landmarks' estimates over the *entire* trajectory by acting as a *smoother* and reducing linearization errors. Moreover, the hybrid MAP estimators are resource-aware, i.e., they can process local analog measurements along with remote measurements quantized to any number of bits.

7.1 Introduction and Related Work

For sensor localization in GPS-denied environments, mobile WSN (e.g., mobile robots) combine noisy observations of motion (e.g., velocity) and relative position (e.g., distance and bearing to each other), from *all* sensors, to accurately estimate their joint-state, i.e., their positions and velocities. Since the sensors are spatially distributed, transmitting these *analog* observations incurs substantial communication overhead. In this work, we focus on WSN localization in the presence of severe communication-bandwidth constraints, where each sensor can transmit *only a few bits* per analog observation. Thus, every sensor carries out *lossy quantization* of its observations and the commonly-used

estimation frameworks for analog observations (e.g., the Kalman Filter (KF) or the Maximum A Posteriori (MAP) estimator) have to be modified to accommodate these quantized observations.

While there exists a large body of work on parameter estimation (either deterministic [114, 115, 116] or random variable [117, 118, 119]) for WSN, we consider approaches that were developed to estimate random processes, as is the case in sensor localization. The Sign-of-Innovation Kalman filter, for estimating stochastic, dynamic processes, has been proposed [44], where the measurement innovation[1] , instead of the actual analog measurement[2] , is quantized to a single bit. When $f \geq 1$ bits are available, this approach has been extended in [39] to the batch and iteratively quantized KF, where with $f = 4$ bits, the performance is almost indistinguishable from that of the standard KF. These estimators are derived for linear, Gaussian motion and measurement models and they approximate the posterior probability density function by a Gaussian[3] after each measurement update, to reduce computational complexity. For nonlinear models, in order to mitigate the effect of linearization and Gaussian approximations carried out above, extensions of this approach to the MAP estimator (that acts as a smoother), for single bit (QMAP) and multiple bits (batch and iteratively quantized BQMAP and IQMAP, respectively), are presented in [45, 122, 111].

The quantized innovation estimators discussed above use the multi-centralized (MC) architecture (robust to single point failures) where each sensor broadcasts *all* its observations and every sensor locally processes observations from the entire network to generate joint-state estimates. However, the quantization rules used by these estimators depend upon the computed state estimates, via the measurement innovation, and thus all sensors have to maintain *identical* estimators to ensure estimation consistency. Hence, each sensor is forced to use quantized versions of its *own* locally-available, analog observations for estimation, thus discarding valuable information that can be used to improve localization accuracy.

To overcome this drawback, we have introduced a hybrid estimation framework

[1] Measurement innovation is the difference between the actual and the estimated (by the estimator) measurement.

[2] We use measurement and observation interchangeably.

[3] Note that due to the nonlinearity of the quantization operation, the posterior pdf is not, in general, a Gaussian.

in Chapter 6, for Minimum Mean Squared Error (MMSE) estimation (filtering), that enables each sensor to incorporate its locally-available, analog observations in the estimation process. Specifically, each sensor maintains two local estimators (see Fig. 7.1): (i) a quantized (Q) estimator that processes quantized observations from all sensors, including itself, and (ii) a hybrid (H) estimator that processes its own analog observations along with the quantized observations from other sensors. In this work, we extend the hybrid estimation framework to MAP estimation and derive a (H) estimator, called H-BQMAP, that by using its own analog observations can outperform the existing BQMAP, that uses only quantized observations.

7.2 Problem Formulation

Consider a mobile WSN consisting of N sensor nodes. The motion (process) model for the WSN is given by the following linear, discrete-time, dynamic system driven by system noise:

$$\mathbf{x}_k = \mathbf{F}_{k-1}\mathbf{x}_{k-1} + \mathbf{G}_{k-1}\mathbf{w}_{k-1}, p(\mathbf{x}_0) \sim \mathcal{N}(\mathbf{x}(0), \mathbf{P}_0) \tag{7.1}$$

where, $\mathbf{w}_k$ is the zero-mean, white, Gaussian, and uncorrelated system noise at time-step k with covariance given by $E\left[\mathbf{w}_k {\mathbf{w}_l}^T\right] = \delta_{kl}\mathbf{Q}_k$. Here, $\mathbf{x}_k = [{\mathbf{x}_k^1}^T, {\mathbf{x}_k^2}^T, \ldots, {\mathbf{x}_k^N}^T]^T$, is the joint-state of the sensor network and $\mathbf{x}_k^i$ denotes the state[4] of each individual sensor i at time-step k, $k = 0, \ldots, K$. $p(\mathbf{x}_0)$ is the prior on the sensors' initial states.

Sensor i obtains M_k^i scalar, analog measurements at time-step k. Sensor i's measurement model, $i = 1, \ldots, N$, is:

$$z_{km}^i = {\mathbf{h}_{km}^i}^T \mathbf{x}_k + v_{km}^i, \ m = 1, \ldots, M_k^i \tag{7.2}$$

where v_{km}^i is zero-mean, white, Gaussian, and uncorrelated measurement noise with $E[v_{km}^i v_{ln}^i] = \delta_{kl,mn}{\sigma_{km}^i}^2$ and $E[v_{km}^i v_{ln}^j] = 0, \ \forall j \neq i, \ j = 1, \ldots, N$. The noise terms in the process and measurement models are independent. We assume that: (i) the process and measurement models are shared *a priori* by all sensors before deployment, and (ii) each sensor can communicate with the team at every time step. The linear models (7.1)-(7.2)

[4] For example, for the sensor localization task, the joint-state consists of the positions and velocities of all the sensors.

are used to simplify the mathematical derivations. In real-world scenarios, the *linearized* system, obtained from the underlying *non-linear* models, will be used. Moreover, a single vector-valued measurement can be decomposed into multiple scalar measurements using pre-whitening [98] and then processed using the above formulation. Lastly, in order to simplify the notation in this chapter, from now onwards, we assume that each sensor i obtains only a single, scalar, analog measurement, z_k^i, at time-step k. The generalization to M_k^i measurements is straightforward.

7.2.1 Real vs. Quantized Measurements

In the absence of communication-bandwidth constraints, the MAP estimate, $\hat{\mathbf{x}}_{0:K}$, of *all* sensors' states from time-step 0 to K, given *all analog* measurements up to time-step K, $\mathbf{z}_{0:K}$, is calculated as:

$$\begin{aligned}\hat{\mathbf{x}}_{0:K} &= \arg\max p(\mathbf{x}_{0:K}|\mathbf{z}_{0:K}) \\ &= \arg\max \frac{1}{p(\mathbf{z}_{0:K})} p(\mathbf{z}_{0:K}|\mathbf{x}_{0:K}) p(\mathbf{x}_{0:K}) \\ &= \arg\max \frac{1}{p(\mathbf{z}_{0:K})} \prod_{k=0}^{K} \prod_{i=1}^{N} p(z_k^i|\mathbf{x}_k) \prod_{k=0}^{K-1} p(\mathbf{x}_{k+1}|\mathbf{x}_k) \cdot p(\mathbf{x}_0)\end{aligned} \tag{7.3}$$

Here, $\mathbf{z}_{0:K} = [(\mathbf{z}_{0:K}^1)^T, \ldots, (\mathbf{z}_{0:K}^N)^T]^T$ where, $\mathbf{z}_{0:K}^i = [z_0^i, \ldots, z_K^i]^T$, $i = 1, \ldots, N$. Under the Gaussian noise assumption for the linear system (7.1)-(7.2), the conditional pdfs in (7.3) and hence the posterior pdf $p(\mathbf{x}_{0:K}|\mathbf{z}_{0:K})$ are Gaussian. Therefore, the optimization problem in (7.3) can be formulated as a Weighted Least Squares and solved using standard methods [64] such as normal equations, QR decomposition, etc. However, note that the real-valued measurements, $\mathbf{z}_{0:K}$, from *all* sensors, are required for solving the optimization problem in (7.3) and each sensor i has to broadcast its measurements, $\mathbf{z}_{0:K}^i$, to the WSN.

On the contrary, in WSNs with severe power and communication bandwidth limitations, each sensor can communicate only $f \geq 1$ bits per analog measurement. Therefore, sensor i must quantize its analog measurement, $z_k^i \in \mathbb{R}$, to $b_k^i \in \mathcal{B}$, $\mathcal{B} := \{1, \ldots, 2^f\}$ using a quantization rule $\mathbf{q}[\cdot]$ of the form:

$$b_k^i = \mathbf{q}[z_k^i], \text{ where } \mathbf{q} : \mathbb{R} \rightarrow \mathcal{B} \tag{7.4}$$

Thus, the MAP estimate of the sensors' states, $\hat{\mathbf{x}}^Q_{0:K}$, given quantized measurements, $\mathbf{b}_{0:K}$, from all sensors, up to time-step K is calculated as:

$$\begin{aligned}\hat{\mathbf{x}}^Q_{0:K} &= \arg\max p(\mathbf{x}_{0:K}|\mathbf{b}_{0:K}) \\ &= \arg\max \frac{1}{p(\mathbf{b}_{0:K})} \prod_{k=0}^{K} \prod_{i=1}^{N} p(b^i_k|\mathbf{x}_k) \prod_{k=0}^{K-1} p(\mathbf{x}_{k+1}|\mathbf{x}_k) \cdot p(\mathbf{x}_0)\end{aligned} \tag{7.5}$$

Importantly, note that due to the nonlinear quantization operation in (7.4), the conditional pdf $p(b^i_k|\mathbf{x}_k)$ and hence the resulting posterior pdf are no longer Gaussian even for the linear process and measurement models.

In the BQMAP estimator [122, 111, 45], that generates these bits by quantizing the measurement innovation, $z^i_k - \mathbf{h}^{i^T}_k \hat{\mathbf{x}}^Q_{\tau k}$, it has been shown that the resulting pdf $p(\mathbf{x}_{0:K}|\mathbf{b}_{0:K})$ in (7.5) is log-concave in $\mathbf{x}_{0:K}$. Hence, we can find a unique, globally optimum solution for the BQMAP estimator. Here, the state estimate $\hat{\mathbf{x}}^Q_{\tau k}$ is chosen to be the latest available MAP estimate (using quantized measurements only) for $\mathbf{x}_k$. However, a shortcoming of this approach is that even though each sensor i has access to its own analog measurements, $\mathbf{z}^i_{0:K}$, the quantized-innovation BQMAP estimator forces it to discard this information and process only the corresponding quantized measurements, $\mathbf{b}^i_{0:K}$. This is because the quantized measurement, b^i_k, depends upon the estimated measurement, $\mathbf{h}^{i^T}_k \hat{\mathbf{x}}^Q_{\tau k}$, which has to be identical for all sensors to ensure consistent estimation. Therefore, all sensors have to process the same set of measurements, so that they can generate identical state estimates, $\hat{\mathbf{x}}^Q_{\tau k}$. In Fig. 7.1, when the shaded Hybrid Estimators do not exist, we obtain the BQMAP.

7.3 Hybrid Estimation Framework

In Chapter 6, we introduced a hybrid estimation framework for *filtering* that enables each sensor to obtain MMSE state estimates (under Gaussian assumption) by processing local analog and remote quantized measurements. In this work, we focus on MAP estimation since for the nonlinear process and measurement models in real-world systems, the MAP estimator acts as a smoother and mitigates linearization errors, hence improving estimation accuracy. To achieve this, we propose that each sensor i maintains two estimators (see Fig. 7.1): (1) a quantized (Q) estimator that processes quantized measurements

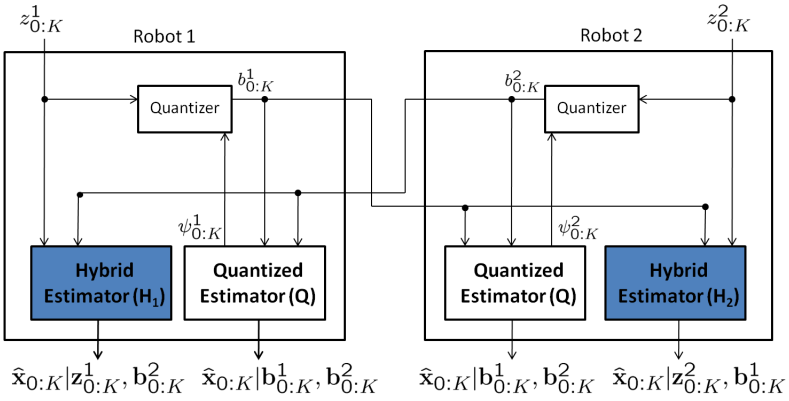


Figure 7.1: Hybrid Estimation framework. Here, $\psi^i_k = \mathbf{h}^{i^T}_k \hat{\mathbf{x}}^Q_{\tau k}$, $k = 0, \ldots, K$, $i = 1, 2$, and using the quantization rule in (7.6), the H- and Q-estimators correspond to the H-BQMAP and BQMAP, respectively.

from all sensors including itself, i.e., $\hat{\mathbf{x}}^Q_{0:K} = \arg\max p(\mathbf{x}_{0:K}|\mathbf{b}^{q\neq i}_{0:K}, \mathbf{b}^i_{0:K})$, and (2) a hybrid (H) estimator that processes its own real-valued measurements and quantized measurements from the other sensors in the team, i.e., $\hat{\mathbf{x}}^{H_i}_{0:K} = \arg\max p(\mathbf{x}_{0:K}|\mathbf{b}^{q\neq i}_{0:K}, \mathbf{z}^i_{0:K})$, $q = 1, \ldots, N$. The estimates generated by the Q-estimator are identical for all sensors since each sensor processes identical measurements, $\mathbf{b}_{0:K} = \left\{\mathbf{b}^{q\neq i}_{0:K}, \mathbf{b}^i_{0:K}\right\}$. Therefore, they are used in the hybrid estimation framework for generating identical quantization thresholds for all sensors, as will be shown in the next section. On the contrary, the estimates generated by the H-estimator are *different* for each sensor, since each sensor processes a different set of quantized and analog measurements. In the next section, we will derive the Q-and H-estimators, BQMAP and H-BQMAP, respectively, for the quantization scenario where the bandwidth availability (f bits per analog measurement) is known *a priori*.

7.3.1 Quantization Rule

Since sensor j is pre-informed about the availability of $f \geq 1$ bits for communicating its analog measurement $z^j_k \in \mathbb{R}$, sensor j partitions the observation space $\mathbb{R}$ into 2^f intervals. The interval $\mathcal{R}^j_k(n) := [\tau^j_k(n), \tau^j_k(n+1))$, where $\tau^j_k(n)$ are the quantization thresholds, $n \in \mathcal{B} := \{1, \ldots, 2^f\}$, $\tau^j_k(1) = -\infty$, $\tau^j_k(2^f + 1) = \infty$, and $\tau^j_k(n) < \tau^j_k(n+1)$.

The quantization rule, based on the measurement innovation, has the form[5] [39]:

$$b_k^j = n, \text{ iff } \tau_k^j(n) < z_k^j - \mathbf{h}_k^{j^T} \hat{\mathbf{x}}_{\tau k}^Q \leq \tau_k^j(n+1) \tag{7.6}$$

where, $\hat{\mathbf{x}}_{\tau k}^Q$ is the BQMAP's latest available estimate for the state $\mathbf{x}_k$. The thresholds $\tau_k^j(n)$ are defined as:

$$\tau_k^j(n) = \Delta(n)\sqrt{(\mathbf{h}_k^{j^T}\mathbf{P}_k\mathbf{h}_k^j + \sigma_k^{j^2})} \tag{7.7}$$

where $\mathbf{P}_k$ is the covariance of the estimate $\hat{\mathbf{x}}_{\tau k}^Q$ and $\Delta(n)$ corresponds to the Lloyd-Max quantization thresholds [123, 124]. This is equivalent to quantizing the measurement innovation, $z_k^j - \mathbf{h}_k^{j^T}\hat{\mathbf{x}}_{\tau k}^Q$, with minimum MSE distortion [39]. Note that sensor j uses its BQMAP's estimate in (7.6) that is identical for all sensors. This enables all sensors to correctly process / decode the quantized measurement, since this estimate is needed in the design of the H-BQMAP and BQMAP [see (7.8), (7.9), and [111]]. Sensor j *cannot* use its H-BQMAP's estimate since it depends upon its local analog measurements, $\mathbf{z}_{0:K}^j$, which are unavailable to the rest of the sensors.

7.3.2 BQMAP and H-BQMAP Estimators

For the batch quantization rule from (7.6), we now derive the resulting (Q) and (H) MAP estimators for the hybrid estimation framework. Note that the Q-estimator, *by definition*, is identical to the BQMAP in [45] and is not described here. Before proceeding, we first calculate the conditional probability $p(b_k^q = n|\mathbf{x}_k)$ as follows:

$$\begin{aligned} p(b_k^q = n|\mathbf{x}_k) &= \Pr\{\tau_k^q(n) < z_k^q - \mathbf{h}_k^{q^T}\hat{\mathbf{x}}_{\tau k}^Q \leq \tau_k^q(n+1)|\mathbf{x}_k\} \\ &= \left(\mathrm{Q}\left[\frac{\tau_k^q(n) - \mathbf{h}_k^{q^T}(\mathbf{x}_k - \hat{\mathbf{x}}_{\tau k}^Q)}{\sigma_k^q}\right] - \mathrm{Q}\left[\frac{\tau_k^q(n+1) - \mathbf{h}_k^{q^T}(\mathbf{x}_k - \hat{\mathbf{x}}_{\tau k}^Q)}{\sigma_k^q}\right]\right) \end{aligned} \tag{7.8}$$

where, $\mathrm{Q}[x] = \int_x^\infty \frac{1}{\sqrt{2\pi}}\exp(-u^2/2)du$, is the Gaussian tail probability and we use the fact that the conditional pdf $p(z_k^q|\mathbf{x}_k) \sim \mathcal{N}(\mathbf{h}_k^{qT}\mathbf{x}_k, \sigma_k^{q^2})$. We now present the derivation for sensor i's H-estimator, the H-BQMAP. The MAP estimate, $\hat{\mathbf{x}}_{0:K}^{H_i}$, computed by

[5] We assume a round-robin scheduling algorithm where the quantized measurements are generated and processed sequentially, based on sensor ids. Therefore, sensor i generates and communicates its quantized measurement, b_k^i, before sensor $(i+1)$. Moreover, all sensors in the team process b_k^i to obtain $\hat{\mathbf{x}}_{0:k,i}^Q$, before sensor $(i+1)$ generates its quantized measurement.

sensor i's H-estimator using (i) its own analog measurements, $\mathbf{z}^i_{0:K}$, and (ii) quantized bits, $\mathbf{b}^{q\neq i}_{0:K}$, $q = 1, \ldots, N$, received from other sensors in the team is given by:

$$\begin{aligned}\hat{\mathbf{x}}^{H_i}_{0:K} &= \arg\max p(\mathbf{x}_{0:K}|\mathbf{b}^{q\neq i}_{0:K}, \mathbf{z}^i_{0:K}) \\ &= \arg\max \left(\frac{1}{p(\mathbf{b}^{q\neq i}_{0:K}, \mathbf{z}^i_{0:K})} \prod_{k=0}^{K} p(z^i_k|\mathbf{x}_k) \prod_{k=0}^{K} \prod_{q=1, q\neq i}^{N} p(b^q_k|\mathbf{x}_k) \prod_{k=0}^{K-1} p(\mathbf{x}_{k+1}|\mathbf{x}_k) \cdot p(\mathbf{x}_0) \right) \quad (7.9)\end{aligned}$$

where we obtain (7.9) using Bayes' rule. For the linear model of (7.1)-(7.2), $p(\mathbf{x}_{k+1}|\mathbf{x}_k) \sim \mathcal{N}(\mathbf{F}_k\mathbf{x}_k, \mathbf{G}_k\mathbf{Q}_k\mathbf{G}^T_k)$ and $p(z^i_k|\mathbf{x}_k) \sim \mathcal{N}({\mathbf{h}^i_k}^T\mathbf{x}_k, \sigma^{i2}_k)$. Therefore, the above equation can be written as:

$$\begin{aligned}\hat{\mathbf{x}}^{H_i}_{0:K} &= \arg\max p(\mathbf{x}_{0:K}|\mathbf{b}^{q\neq i}_{0:K}, \mathbf{z}^i_{0:K}) \\ &= \arg\max \left(\prod_{k=0}^{K} \mathcal{N}({\mathbf{h}^i_k}^T\mathbf{x}_k, \sigma^{i2}_k) \prod_{k=0}^{K} \prod_{q=1, q\neq i}^{N} p(b^q_k|\mathbf{x}_k) \prod_{k=0}^{K-1} \mathcal{N}(\mathbf{F}_k\mathbf{x}_k, \mathbf{G}_k\mathbf{Q}_k\mathbf{G}^T_k)\, \mathcal{N}(\mathbf{x}(0), \mathbf{P}_0) \right) \\ &= \arg\min \left(\sum_{k=0}^{K} \frac{1}{2}||z^i_k - {\mathbf{h}^i_k}^T\mathbf{x}_k||^2_{\sigma^{i-2}_k} - \sum_{k=0}^{K} \sum_{q=1, q\neq i}^{N} \log p(b^q_k|\mathbf{x}_k) \right. \\ &\qquad \left. + \sum_{k=0}^{K-1} \frac{1}{2}||\mathbf{x}_{k+1} - \mathbf{F}_k\mathbf{x}_k||^2_{(\mathbf{G}_k\mathbf{Q}_k\mathbf{G}^T_k)^{-1}} + \frac{1}{2}||\mathbf{x}_0 - \mathbf{x}(0)||^2_{\mathbf{P}_0^{-1}} \right) \quad (7.10)\end{aligned}$$

where $p(b^q_k|\mathbf{x}_k)$ is as defined in (7.8). The $p(\mathbf{b}^{q\neq i}_{0:K}, \mathbf{z}^i_{0:K})$ is a normalizing constant that is neglected during optimization. Next, we prove a very important property for the posterior pdf $p(\mathbf{x}_{0:K}|\mathbf{b}^{q\neq i}_{0:K}, \mathbf{z}^i_{0:K})$.

Lemma 3. *The posterior pdf of the H-BQMAP in* (7.9) *is log-concave.*

Proof. Consider the innovation term $r^q_k = z^q_k - {\mathbf{h}^q_k}^T\hat{\mathbf{x}}^Q_{\tau k}$, where $p(r^q_k|\mathbf{x}_k) \sim \mathcal{N}({\mathbf{h}^q_k}^T(\mathbf{x}_k - \hat{\mathbf{x}}^Q_{\tau k}), \sigma^{q2}_k)$. This conditional pdf is log-concave in both arguments. Therefore, $p(b^q_k|\mathbf{x}_k) = \int_{\tau^q_k(n)}^{\tau^q_k(n+1)} p(r^q_k|\mathbf{x}_k)dr^q_k$ is also log-concave since it involves the integral over a convex set of a log-concave pdf [126]. Thus, both the Gaussian pdfs and (7.8) are log-concave. Also, log-concavity is closed under multiplication and preserved under linear transformation of the argument [127, 122]. □

We conclude that, due to the log-concavity of this posterior pdf, the optimization problem in (7.10) is nonlinear but convex. Hence, it is guaranteed to converge to the global optimum and can be solved using efficient convex optimization techniques [127] such as Newton's method, the interior point methods, etc. Moreover, its solution, i.e., the MAP estimate, is unique and the global optimum.

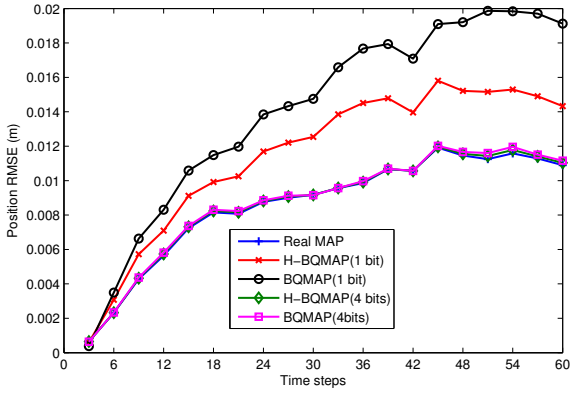
7.4 Simulation Results

The simulation set up consists of two sensors deployed in 1D. The motion model for these sensors is given by a constant velocity statistical model [42]. Each sensor obtains noisy measurements for its own velocity, v_m, and distance, d_m, to the other sensor, with the noise modeled as zero-mean, white Gaussian with std. deviation $\sigma_{v_m} = 0.01$ m/s and $\sigma_{d_m} = 0.037$ m, respectively. We compare the performance of the proposed H-estimator, H-BQMAP, using $f = \{1, 4\}$ bits per analog measurement, with: (1) the Q-estimator (local and remote quantized measurements), BQMAP, and (2) the real-valued MAP that uses analog measurements from all sensors and hence is our benchmark. Fig. 7.2 shows the root mean squared error (RMSE) in the position and velocity estimates for these estimators, averaged over the 2 sensors and 10 Monte Carlo trials. Since the estimates generated by the H-BQMAP are different for each sensor, the RMSE for H-BQMAP is also averaged over estimators maintained by each sensor. As evident from Fig. 7.2, the estimates generated by the H-BQMAP are more accurate that the BQMAP, irrespective of the number of quantization bits considered. This is expected since the H-BQMAP includes local analog measurements in the estimation process. Specifically, when $f = 1$, the H-BQMAP is significantly more accurate than the BQMAP. Overall, the error in the estimates[6] decreases as we increase the number of quantization bits (results for $f = \{2, 3\}$ bits not shown for clarity) and by communicating as few as 4 bits per analog measurement, both the H-BQMAP and BQMAP are able to achieve accuracy very close to that of the real MAP estimator.

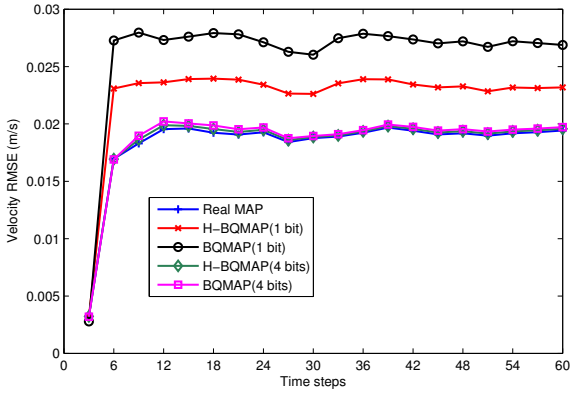
7.5 Summary

In this chapter, we presented an extension of the hybrid estimation framework to MAP estimators. We conducted preliminary simulation tests that demonstrated that the hybrid MAP estimators outperform the quantized MAP estimators. As part of our future work, we plan to design iterative quantized hybrid MAP estimators that can optimally utilize additional bandwidth that becomes available on-the-fly.

[6] Note that this gain in estimation accuracy comes at the cost of increased processing, as each sensor has to maintain both H- and Q-MAP estimators (see Fig. 7.1).



(a) Comparison of position RMSE.



(b) Comparison of velocity RMSE.

Figure 7.2: Comparison of position and velocity RMSE for the real-valued, quantized and hybrid MAP estimators.

Chapter 8

Concluding Remarks

8.1 Summary of contributions

In the preceding chapters, we developed resource-aware estimation algorithms for mobile robot localization, focusing on the key problems of SLAM and CL. The robots' processing and communication resources are limited, and thus it becomes necessary to develop estimators that maximize localization accuracy, subject to these resource-constraints. The main contributions of this work can be summarized as follows:

- **Mobile robot localization under processing constraints**
  In Chapters 2 and 3, we studied the problem of single-robot SLAM and presented resource-aware solutions using the two most commonly-used estimation frameworks; the MMSE estimator and the batch MAP estimator. Specifically, in Chapter 2, we developed the approximate and consistent MMSE-based Power-SLAM estimator, which in contrast to the quadratic computational complexity of the standard EKF-based SLAM, has processing requirements only *linear* in the number of features in the map. Moreover, as opposed to existing *ad hoc* approximate approaches, the Power-SLAM estimator exploits the structure of the EKF's covariance update equation to systematically minimize the information loss over multiple time steps, resulting in accurate and consistent estimation. Lastly, a key advantage of Power-SLAM is that it is an *anytime* algorithm whose processing requirements can be adjusted online to match the available resources, while maximizing the achieved estimation accuracy. Next, in Chapter 3, we developed

the constrained keyframe-based localization and mapping (C-KLAM) algorithm, an approximate batch MAP-based estimator for SLAM. In C-KLAM, instead of building a computationally-expensive dense map of the environment, we estimate only the key robot poses and the associated key landmark-map over the entire robot trajectory. As a result, we not only maintain the sparsity of the Hessian, necessary for generating fast solutions, but also enable efficient loop closures, necessary for ensuring accurate and consistent long-term navigation. Moreover, instead of discarding measurement information from non-key poses and landmarks, this information is used in C-KLAM (through marginalization) to generate constraints between the key poses, without affecting the sparse structure of the Hessian matrix. Thus, the C-KLAM approach not only achieves substantial speed-up, but it also provides estimation accuracy comparable to that of the batch MAP-based SLAM, which uses all available measurements.

For CL under processing constraints, we developed a consistent and fully-distributed version of the batch MAP-based CL estimator in Chapter 4. In particular, we introduced a novel data-distribution scheme that utilizes the computational and storage resources of all robots in the team in order to speed up the processing of the CL algorithm. Moreover, we presented a resource-aware extension of this approach, that generates an approximate solution with reduced processing and communication requirements. Lastly, in order to facilitate the selection of key design parameters (that trade estimation accuracy for resource utilization) in the above resource-constrained estimation algorithms for SLAM and CL, we carried out detailed computational and communication (for CL) complexity analysis.

- **Mobile robot localization under communication constraints**
  The second part of this thesis work focused on developing estimation algorithms for CL under asynchronous communication and bandwidth constraints. In Chapter 5, we developed a general framework of information-transfer schemes that enable centralized-equivalent and consistent MC-CL in the presence of asynchronous communication constraints. The proposed information-transfer schemes were estimator-independent and generated delayed centralized-equivalent pose estimates, where

the time-delay increased with decreasing bandwidth-availability per communication link. Moreover, in order to facilitate application-specific information-transfer-scheme selection, we presented: (i) analytical results for the expected time-delay, (ii) necessary and sufficient conditions on the communication-graph connectivity, and (iii) detailed communication complexity analysis. Lastly, in Chapters 6 and 7, we studied the problem of MC-CL under severe communication-bandwidth constraints, when robots can communicate only a quantized version (few bits) of their analog measurements to the team. In contrast to existing estimators that can process either analog or quantized measurements, we developed novel MMSE and MAP-based hybrid estimators that enable robots to process all available quantized and analog measurement information, resulting in improved estimation accuracy. Moreover, we also presented resource-aware extensions for the proposed hybrid estimators that handle time-varying communication-bandwidth availability, when robots communicate $n \geq 1$ bits per analog measurement.

In conclusion, we believe that the resource-aware robot localization algorithms developed in this work, coupled with the advances in processing, sensing, and communication technologies, will significantly improve the localization performance of robots employed in various application domains and result in a proliferation of novel location-based applications.

8.2 Future research directions

Our work on resource-aware localization algorithms opens a number of interesting avenues for future research. Nowadays, a large number of commonly-available mobile hand-held devices, such as cell phones and PDAs, are equipped with sensing (e.g., cameras, IMUs, and compass), processing (e.g., system on a chip), and networking (e.g., bluetooth and Wi-Fi) capabilities. Therefore, in order to harness these resources for novel location-based mobile sensor applications, it becomes necessary to develop accurate and consistent localization algorithms that can operate with limited resources over longer periods of time. To this end, our ongoing and future research efforts will focus on porting and evaluating the performance of our Power-SLAM and C-KLAM algorithms on mobile hand-held devices. However, this transition from high-quality research-grade

devices used in our experiments, to commercial mobile devices characterized by cheap and low-quality sensing and processing modalities, presents several research challenges. For example, as opposed to research-quality global shutter cameras that have been used in our experiments, cell phones are equipped with rolling shutter cameras. Therefore, the rolling shutter time also has to be estimated [128] in order to generate consistent and accurate results. Similarly, the IMUs used in cell phones suffer from larger bias noise and necessitate the estimation of additional parameters such as the sensors' axis misalignment and the unknown scale factors [129]. Moreover, the measurement frequencies of different sensing modalities are not fixed and hence the time synchronization between sensors also has to be estimated over time [130]. Lastly, the limited processing resources and stringent battery-power constraints contribute to the difficulty of the localization task. Therefore, our future work will focus on systematic development and testing of efficient solutions that handle these additional estimation parameters in order to ensure reliable and long-term navigation using mobile hand-held devices.

Another interesting research problem, specifically for the C-KLAM introduced in Chapter 3, is that of accurate selection and re-detection (during loop-closure) of key landmarks in the environment. Our work on C-KLAM, until now, concentrated on developing an estimation framework that generates accurate and sparse maps of the environment, while the sensor-data selection component of the algorithm, that is, choosing key features and poses, was treated simplistically (selected periodically over time). However, in order to optimize C-KLAM's performance, it is of critical importance to investigate algorithms for efficiently and reliably determining key features in the environment. Thus, to ensure real-time localization and mapping over long time periods, we will focus on two main key feature-related research threads in our future work: (i) identification and fast extraction of unique feature descriptors (e.g., ORB [131], FREAK [132], and BRIEF [133]), and (ii) efficient feature-matching algorithms (e.g., vocabulary trees [84]) to establish loop-closures and improve localization accuracy.

Concerning our work on CL, obvious but important extensions are those to 3D and cooperative SLAM (C-SLAM). An increasing number of applications [134, 135] involve both ground and aerial robot teams operating in expansive 3D environments and therefore, it becomes necessary to estimate the full 6 d.o.f. of the robots' poses. Moreover, we will also focus on developing resource-aware distributed mapping algorithms and

information transfer schemes that are able to consistently fuse local maps, built by individual robots, in order to generate accurate global maps of the environment.

Lastly, our work on the hybrid estimation framework brings up a number of fundamental research questions. An important topic that needs to be investigated is the analytical performance characterization of the designed hybrid estimators. In particular, along with the expected performance in terms of their estimation accuracy, we will also quantify the expected loss in performance of the hybrid estimators as compared to their real-valued counterparts. Another avenue for future research will be the design of encoding rules capable of incorporating information from the more accurate H-estimator (as opposed to the current quantization rules in Chapters 6 and 7 that use information only from the Q-estimators), in order to further improve the performance of the hybrid estimators. Since both the Q- and H-estimators use information from the quantized bits, this research will entail the design of joint estimators which will be able to account for correlations that exist between the quantized and hybrid estimates. Lastly, we will extend the hybrid estimation framework to nonparametric methods. Until now, our work focused on developing the hybrid framework for parametric estimation problems, where there exist common priors or models that are known to all sensors. These priors / models are used by all sensors not only as thresholds for *generating* quantized measurements, but also for efficiently *processing* the received quantized bits. However, in many multi-sensor applications (e.g., kriging [136] and environmental monitoring in disaster areas), no such parametric form can be assumed for the underlying distribution and hence we need to develop and implement novel *data-driven quantization* schemes along with the corresponding hybrid estimators and / or predictors.

As we finish this work, we are excited to find mobile robot systems at the brink of widespread field deployment in real-world applications. We hope that our work helps bring mobile robotics one step closer to this goal.

References

[1] C. Parker, H. Zhang, and C. R. Kube. Blind bulldozing: Multiple robot nest construction. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2010–2015, Las Vegas, NV, Oct. 27–31 2003.

[2] B. Sellner, F.W. Heger, L. M. Hiatt, R. Simmons, and S. Singh. Coordinated multiagent teams and sliding autonomy for large-scale assembly. *Proceedings of the IEEE - Special Issue on Multi-Robot Systems*, 94(7):1425–1444, Jul. 2006.

[3] A. Broggi, M. Bertozzi, A. Fascioli, C. G. L. Bianco, and A. Piazzi. Visual perception of obstacles and vehicles for platooning. *IEEE Transactions on Intelligent Transportation Systems*, 1(3):164–176, Sep. 2000.

[4] S. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, J. Diebel, P. Fong, J. Gale, M. Halpenny, G. Hoffmann, K. Lau, C. Oakley, M. Palatucci, V. Pratt, P. Stang, S. Strohband, C. Dupont, L.-E. Jendrossek, C. Koelen, C. Markey, C. Rummel, J. van Niekerk, E. Jensen, P. Alessandrini, G. Bradski, B. Davies, S. Ettinger, A. Kaehler, A. Nefian, and P. Mahoney. Stanley, the robot that won the DARPA Grand Challenge. *Journal of Field Robotics*, 23(9):661–692, Jun. 2006.

[5] Y. Feng, Z. Zhu, and J. Xiao. Heterogeneous multi-robot localization in unknown 3D space. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4533–4538, Beijing, China, Oct. 9–15 2006.

[6] F. Capezio, F. Mastrogiovanni, A. Sgorbissa, and R. Zaccaria. Robot-assisted surveillance in large environments. *Journal of Computing and Information Technology*, 17(1):95–108, Mar. 2009.

[7] A. Dhariwal, G. Sukhatme, and A. A. G. Requicha. Bacterium-inspired robots for environmental monitoring. In *Proc. of the IEEE International Conference on Robotics and Automation*, pages 1436–1443, New Orleans, LA, Apr. 26–May 1 2004.

[8] J. A. Hesch and S. I. Roumeliotis. Design and analysis of a portable indoor localization aid for the visually impaired. *International Journal of Robotics Research*, 29(11):1400–1415, Sep. 2010.

[9] I. Ulrich and J. Borenstein. The GuideCane - Applying mobile robot technologies to assist the visually impaired. *IEEE Transactions on Systems, Man, and Cybernetics - Part A*, 31(2):131–136, Mar. 2001.

[10] A. D. Tews, G. S. Sukhatme, and M. J. Mataric. A multi-robot approach to stealthy navigation in the presence of an observer. In *Proc. of the IEEE International Conference on Robotics and Automation*, pages 2379–2385, New Orleans, LA, Apr. 26–May 1 2004.

[11] J. Casper and R. R. Murphy. Human-robot interactions during the robot-assisted urban search and rescue response at the World Trade Center. *IEEE Transactions on Systems, Man, and Cybernetics - Part B*, 33(2):367–385, Jun. 2003.

[12] T. L. Huntsberger, A. T.-Ollennu, H. Aghazarian, P. S. Schenker, P. Pirjanian, and H. D. Nayar. Distributed control of multi-robot systems engaged in tightly coupled tasks. *Autonomous Robots*, 17(1):79–92, 2004.

[13] J. K. Erickson. Living the dream - An overview of the Mars exploration project. *IEEE Robotics and Automation Magazine*, 13(2):12–18, Jun. 2006.

[14] J.S. Jennings, G. Whelan, and W.F. Evans. Cooperative search and rescue with a team of mobile robots. In *Proc. of 8th International Conference on Advanced Robotics*, pages 193–200, Monterey, CA, Jul. 7–9 1997.

[15] C. Harris and M. Stephens. A combined corner and edge detector. In *Proc. of the 4th Alvey Vision Conference*, pages 147–151, Manchester, UK, Aug. 31–Sep. 2 1988.

[16] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, Nov. 2004.

[17] S. I. Roumeliotis and G. A. Bekey. SEGMENTS: A layered, dual-Kalman filter algorithm for indoor feature extraction. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 454–461, Takamatsu, Japan, Oct. 30–Nov. 5 2000.

[18] J. Weingarten and R. Siegwart. EKF-based 3D SLAM for structured environment reconstruction. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3834–3839, Edmonton, Canada, Aug. 2–6 2005.

[19] R. Smith and P. Cheeseman. On the representation and estimation of spatial uncertainty. *International Journal of Robotics Research*, 5(4):56–68, Dec. 1986.

[20] P. Moutarlier and R. Chatila. Stochastic multisensory data fusion for mobile robot location and environment modeling. In *Proc. of the Fifth International Symposium of Robotics Research*, pages 85–94, Tokyo, Japan, Aug. 28–31 1989.

[21] T. Bailey and H. Durrant-Whyte. Simultaneous localization and mapping: Part II. *IEEE Robotics Automation Magazine*, 13(3):108–117, Sep. 2006.

[22] H. Durrant-Whyte and T. Bailey. Simultaneous localization and mapping: Part I. *IEEE Robotics Automation Magazine*, 13(2):99–110, Jun. 2006.

[23] A. I. Mourikis and S. I. Roumeliotis. Analytical characterization of the accuracy of SLAM without absolute orientation measurements. In *Proc. of the Robotics: Science and Systems*, pages 215–222, Philadelphia, PA, Aug. 16–19 2006.

[24] H. Sugiyama, T. Tsujioka, and M. Murata. Collaborative movement of rescue robots for reliable and effective networking in disaster area. In *Proc. of the International Conference on Collaborative Computing: Networking, Applications and Worksharing*, San Jose, CA, Dec. 19–21 2005.

[25] R. W. Beard, T. W. McLain, D. B. Nelson, D. Kingston, and D. Johanson. Decentralized cooperative aerial surveillance using fixed-wing miniature UAVs. *Proceedings of the IEEE*, 94(7):130–1324, Jul. 2006.

[26] B. Kesner, J. S. Plante, P. J. Boston, T. Fabian, and S. Dubowsky. Mobility and power feasibility of a microbot team system for extraterrestrial cave exploration. In *Proc. of the IEEE International Conference on Robotics and Automation*, pages 4893–4898, Rome, Italy, Apr. 10–14 2007.

[27] I. M. Rekleitis, G. Dudek, and E. Milios. Multi-robot collaboration for robust exploration. In *Proc. of IEEE International Conference on Robotics and Automation*, pages 3164–3169, San Francisco, CA, Apr. 24–28 2000.

[28] S. I. Roumeliotis and G. A. Bekey. Distributed multirobot localization. *IEEE Transactions on Robotics and Automation*, 18(5):781–795, Oct. 2002.

[29] A. I. Mourikis and S. I. Roumeliotis. Performance analysis of multirobot cooperative localization. *IEEE Transactions on Robotics*, 22(4):666–681, Aug. 2006.

[30] A. I. Mourikis and S. I. Roumeliotis. Predicting the accuracy of cooperative simultaneous localization and mapping (C-SLAM). *International Journal of Robotics Research*, 25(12):1273–1286, Dec. 2006.

[31] S. M. Kay. *Fundamentals of Statistical Signal Processing: Estimation Theory.* Prentice Hall, Upper Saddle River, 1993.

[32] A. Howard, M. J. Matarić, and G. S. Sukhatme. Localization for mobile robot teams using maximum likelihood estimation. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 434–439, EPFL, Switzerland, Sep. 30–Oct. 5 2002.

[33] F. Dellaert and M. Kaess. Square root SAM: Simultaneous Localization and Mapping via square root information smoothing. *International Journal of Robotics Research*, 25(12):1181–1203, Dec. 2006.

[34] E. D. Nerurkar, S. I. Roumeliotis, and A. Martinelli. Distributed Maximum A Posteriori estimation for multi-robot cooperative localization. In *Proc. of the IEEE International Conference on Robotics and Automation*, pages 1402–1409, Kobe, Japan, May 12–17 2009.

[35] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. FastSLAM: A factored solution to the simultaneous localization and mapping problem. In *Proc. of the AAAI National Conference on Artificial Intelligence*, pages 593–598, Edmonton, Canada, Jul. 28–Aug. 1 2002.

[36] D. Fox, W. Burgard, H. Kruppa, and S. Thrun. A probabilistic approach to collaborative multi-robot localization. *Autonomous Robots*, 8(3):325–344, Jun. 2000.

[37] S. I. Roumeliotis. *Robust mobile robot localization: From single-robot uncertainties to multi-robot interdependencies*. PhD thesis, University of Southern California, Los Angeles, CA, 2000.

[38] F. Dellaert, F. Alegre, and E. B. Martinson. Intrinsic localization and mapping with 2 applications: Diffusion mapping and marco polo localization. In *Proc. of the IEEE International Conference on Robotics and Automation*, pages 2344–2349, Taipei, Taiwan, Sep. 14–19 2003.

[39] E. J. Msechu, S. I. Roumeliotis, A. Ribeiro, and G. B. Giannakis. Decentralized quantized kalman filtering with scalable communication cost. *IEEE Transactions on Signal Processing*, 56(8):3727–3741, Aug. 2008.

[40] Velodyne LIDAR. Available: `http://velodynelidar.com/lidar/lidar.aspx`.

[41] Microsoft Kinect. Available: `http://www.microsoft.com/en-us/kinectforwindows/`.

[42] Y. Bar-Shalom and X.-R. Li. *Estimation and Tracking: Principles, Techniques and Software*. Norwood, MA: Artech House, Inc., 1993.

[43] K. Y. K. Leung, T. D. Barfoot, and H. H. T. Liu. Decentralized localization of sparsely-communicating robot networks: A centralized-equivalent approach. *IEEE Transactions on Robotics*, 26(1):62–77, Feb. 2010.

[44] A. Ribeiro, G. B. Giannakis, and S. I. Roumeliotis. SOI-KF: Distributed kalman filtering with low-cost communications using the sign of innovations. *IEEE Transactions on Signal Processing*, 54(12):4782–4795, Dec. 2006.

[45] N. Trawny. *Cooperative localization: On motion-induced initialization and joint state estimation under communication constraints*. PhD thesis, University of Minnesota, Minneapolis, MN, Aug. 2010.

[46] E. D. Nerurkar and S. I. Roumeliotis. Power-SLAM: A Linear-Complexity, Consistent Algorithm for SLAM. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 636–643, San Diego, CA, Oct. 29–Nov. 2 2007.

[47] E. D. Nerurkar and S. I. Roumeliotis. Power-SLAM: A linear-complexity, anytime algorithm for SLAM. *International Journal of Robotics Research*, 30(6):772–788, May 2011.

[48] M. A. Paskin. Thin Junction Tree Filters for Simultaneous Localization and Mapping. In *Proc. of the 18th International Joint Conference on Artificial Intelligence*, pages 1157–1164, Acapulco, Mexico, Aug. 9–15 2003.

[49] U. Frese, P. Larsson, and T. Duckett. A multilevel relaxation algorithm for simultaneous localization and mapping. *IEEE Transactions on Robotics*, 21(2):196–207, Apr. 2005.

[50] U. Frese. Treemap: An O(log n) algorithm for indoor simultaneous localization and mapping. *Autonomous Robots*, 21(2):103–122, Sep. 2006.

[51] A. Davison. *Mobile Robot Navigation using Active Vision*. PhD thesis, Oxford University, Department of Engineering Science, Feb. 1998.

[52] J. Knight, A. Davison, and I. Reid. Towards constant time SLAM using postponement. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 405–413, Maui, HI, Oct. 29–Nov. 3 2001.

[53] S. Williams, G. Dissanayake, and H. Durrant-Whyte. An efficient approach to the simultaneous localisation and mapping problem. In *Proc. of the IEEE International Conference on Robotics and Automation*, pages 406–411, Washington, DC, May 11–15 2002.

[54] L. M. Paz, J. D. Tardos, and J. Neira. Divide and Conquer: EKF SLAM in O(n). *IEEE Transactions on Robotics*, 24(5):1107–1120, Oct. 2008.

[55] G. Dissanayake, H. Durrant-Whyte, and T. Bailey. A computationally efficient solution to the simultaneous localisation and map building (SLAM) problem. In *Proc. of the IEEE International Conference on Robotics and Automation*, pages 1009–1014, San Francisco, CA, Apr. 24–28 2000.

[56] H. Durrant-Whyte, G. Dissanayake, and P. W. Gibbens. Towards deployment of large-scale simultaneous localization and map building (SLAM) systems. Technical report, Australian Centre for Field Robotics, University of Sydney, 2000.

[57] J. Leonard and H. J. S. Feder. A computationally efficient method for large-scale concurrent mapping and localization. In *Proc. of the 9th International Symposium on Robotics Research*, pages 169–176, Snowbird, Utah, Oct. 9–12 1999.

[58] M. Csorba and H. Durrant-Whyte. New approach to map building using relative position estimates. In *Proc. of SPIE: Navigation and Control Technologies for Unmanned Systems II*, pages 115–125, Orlando, FL, Apr. 21 1997.

[59] P. Newman. *On the Structure and Solution of the Simultaneous Localization and Map Building Problem.* PhD thesis, University of Sydney, 1999.

[60] J. E. Guivant and E. M. Nebot. Optimization of the simultaneous localization and map-building algorithm for real-time implementation. *IEEE Transactions on Robotics and Automation*, 17(3):242–257, Jun. 2001.

[61] J. K. Uhlmann, S. J. Julier, and M. Csorba. Nondivergent simultaneous map building and localization using covariance intersection. In *Proc. of SPIE: Navigation and Control Technologies for Unmanned Systems II*, pages 2–11, Orlando, FL, Apr. 21 1997.

[62] S. J. Julier and J. K. Uhlmann. Simultaneous localisation and map building using split covariance intersection. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1257–1262, Maui, HI, Oct. 29–Nov.3 2001.

[63] S. J. Julier. A sparse weight Kalman filter approach to simultaneous localisation and map building. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1251–1256, Maui, HI, Oct. 29–Nov.3 2001.

[64] G. H. Golub and C. F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, 1996.

[65] G. Dissanayake, P. Newman, S. Clark, H. Durrant-Whyte, and M. Csorba. A solution to the simultaneous localization and map building (SLAM) problem. *IEEE Transactions on Robotics and Automation*, 17(3):229–241, Jun. 2001.

[66] G. P. Huang, A. I. Mourikis, and S. I. Roumeliotis. Observability-based rules for designing consistent EKF SLAM estimators. *International Journal of Robotics Research*, 29(5):502–528, Apr. 2010.

[67] Z. Zhang. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, Nov. 2000.

[68] J. Heikkila and O. Silven. A four-step camera calibration procedure with implicit image correction. In *Proc. of the Conference on Computer Vision and Pattern Recognition*, pages 1106–1112, San Jaun, Puerto Rico, Jun. 17–19 1997.

[69] R. Kummerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard. g2o: A general framework for graph optimization. In *Proc. of the IEEE International Conference on Robotics and Automation*, pages 3607–3613, Shanghai, China, May 9–13 2011.

[70] K. Konolige, G. Grisetti, R. Kummerle, W. Burgard, B. Limketkai, and R. Vincent. Efficient Sparse Pose Adjustment for 2D mapping. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 22–29, Taipei, Taiwan, Oct. 18–22 2010.

[71] M. Kaess, A. Ranganathan, and F. Dellaert. iSAM: Incremental Smoothing and Mapping. *IEEE Transactions on Robotics*, 24(6):1365–1378, Dec. 2008.

[72] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. Leonard, and F. Dellaert. iSAM2: Incremental smoothing and mapping using the Bayes tree. *International Journal of Robotics Research*, 21(2):217–236, Feb. 2012.

[73] J. Folkesson and H. Christensen. Graphical SLAM–A self-correcting map. In *Proc. of the IEEE International Conference on Robotics and Automation*, pages 383–390, New Orleans, LA, Apr. 26–May 1 2004.

[74] A. I. Mourikis, N. Trawny, S. I. Roumeliotis, A. Johnson, A. Ansar, and L. Matthies. Vision-aided inertial navigation for spacecraft entry, descent, and landing. *IEEE Transactions on Robotics*, 25(2):264–280, Apr. 2009.

[75] G. Sibley, L. Matthies, and G. Sukhatme. Sliding window filter with application to planetary landing. *Journal of Field Robotics*, 27(5):587–608, Sep./Oct. 2010.

[76] G. Klein and D. Murray. Parallel Tracking And Mapping for small AR workspaces. In *Proc. of the IEEE and ACM International Symposium on Mixed and Augmented Reality*, pages 225–234, Nara, Japan, Nov. 13–16 2007.

[77] K. Konolige and M. Agrawal. FrameSLAM: From bundle adjustment to real-time visual mapping. *IEEE Transactions on Robotics*, 24(5):1066–1077, Oct. 2008.

[78] H. Strasdat, J. M. M. Montiel, and A. J. Davison. Scale drift-aware large scale monocular SLAM. In *Proc. of Robotics: Science and Systems*, Zaragoza, Spain, Jun. 27–30 2010.

[79] H. Strasdat, A. J. Davison, J. M. M. Montiel, and K. Konolige. Double window optimisation for constant time visual SLAM. In *Proc. of the IEEE International Conference on Computer Vision*, pages 2352–2359, Barcelona, Spain, Nov. 6–13 2011.

[80] K. Konolige, J. Bowman, J. D. Chen, P. Mihelich, M. Calonder, V. Lepetit, and P. Fua. View-based maps. *International Journal of Robotics Research*, 29(29):941–957, Jul. 2010.

[81] R. M. Eustice, H. Singh, and J. J. Leonard. Exactly sparse delayed-state filters for view-based SLAM. *IEEE Transactions on Robotics*, 22(6):1100–1114, Dec. 2006.

[82] H. Johannsson, M. Kaess, M. Fallon, and J. Leonard. Temporally Scalable Visual SLAM using a Reduced Pose Graph. In *Proc. of the IEEE International Conference on Robotics and Automation*, pages 54–61, Karlsruhe, Germany, May 6–10 2013.

[83] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon. Bundle adjustment – a modern synthesis. *Lecture Notes in Computer Science*, 1883:298–372, Jan. 2000.

[84] D. Nister and H. Stewenius. Scalable recognition with a vocabulary tree. In *Proc. of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2161–2168, New York, NY, Jun. 17–22 2006.

[85] D. Fox, W. Burgard, H. Kruppa, and S. Thrun. Collaborative multi-robot localization. In *Proc. of the 23rd Annual German Conference on Artificial Intelligence*, pages 255–266, Bonn, Germany, Sep. 13–15 1999.

[86] R. Kurazume, S. Nagata, and S. Hirose. Cooperative positioning with multiple robots. In *Proc. of the IEEE International Conference on Robotics and Automation*, pages 1250–1257, San Diego, CA, May 8–13 1994.

[87] R. Grabowski, L. E. Navarro-Serment, C. J. J. Paredis, and P. K. Khosla. Heterogeneous teams of modular robots for mapping and exploration. *Autonomous Robots*, 8(3):293–308, Jun. 2000.

[88] R. Kurazume, S. Hirose, S. Nagata, and N. Sashida. Study on cooperative positioning system (basic principle and measurement experiment). In *Proc. of the IEEE International Conference on Robotics and Automation*, pages 1421–1426, Minneapolis, MN, Apr. 22–28 1996.

[89] R. Kurazume and S. Hirose. Study on cooperative positioning system: Optimum moving strategies for CPS-III. In *Proc. of the IEEE International Conference on Robotics and Automation*, pages 2896–2903, Leuven, Belgium, May 16–20 1998.

[90] R. Kurazume and S. Hirose. An experimental study of a cooperative positioning system. *Autonomous Robots*, 8(1):43–52, Jan. 2000.

[91] A. Martinelli, F. Pont, and R. Siegwart. Multi-robot localization using relative observations. In *Proc. of the IEEE International Conference on Robotics and Automation*, pages 2797–2802, Barcelona, Spain, Apr. 18–22 2005.

[92] A. Ansar and C. Sanderson. A distributed algorithm for cooperative navigation among multiple mobile robots. *Advanced Robotics*, 12(4):335–349, Dec. 1997.

[93] K. X. Zhou and S. I. Roumeliotis. A sparsity-aware QR decomposition algorithm for efficient cooperative localization. In *Proc. of the IEEE International Conference on Robotics and Automation*, pages 799–806, St. Paul, MN, May 14–18 2012.

[94] D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific, 1999.

[95] R. Grabowski and P. Khosla. Localization techniques for a team of small robots. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1067–1072, Maui, HI, Oct. 29–Nov. 3 2001.

[96] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, 1992.

[97] T. A. Davis. *Direct Methods for Sparse Linear Systems (Fundamentals of Algorithms 2)*. Society for Industrial and Applied Mathematics, 2006.

[98] P. S. Maybeck. *Stochastic Models, Estimation, and Control*, volume 1. Academic Press, New York, 1979.

[99] T. Bailey, M. Bryson, H. Mu, J. Vial, L. McCalman, and H. Durrant-Whyte. Decentralised cooperative localisation for heterogeneous teams of mobile robots. In *Proc. of the IEEE International Conference on Robotics and Automation*, pages 2859–2865, Shanghai, China, May 9–13 2011.

[100] V. Caglioti, A. Citterio, and A. Fossati. Cooperative, distributed localization in multi-robot systems: A minimum-entropy approach. In *Proc. of IEEE Workshop on Distributed Intelligent Systems: Collective Intelligence and Its Applications*, pages 25–30, Prague, Czech Republic, Jun. 15–16 2006.

[101] A. I. Mourikis and S. I. Roumeliotis. Optimal sensor scheduling for resource-constrained localization of mobile robot formations. *IEEE Transactions on Robotics*, 22(5):917–931, Oct. 2006.

[102] L. Chen, P. O. Arambel, and R. K. Mehra. Estimation under unknown correlation: Covariance intersection revisited. *IEEE Transactions on Automatic Control*, 47(11):1879–1882, Nov. 2002.

[103] P. O. Arambel, C. Rago, and R. K. Mehra. Covariance intersection algorithm for distributed spacecraft state estimation. In *Proc. of the American Control Conference*, pages 4398–4403, Arlington, VA, Jun. 25–27 2001.

[104] L. C. Carrillo-Arce, E. D. Nerurkar, J. L. Gordillo, and S. I. Roumeliotis. Decentralized multi-robot cooperative localization using covariance intersection. In *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1412–1417, Toyko, Japan, Nov. 3–7 2013.

[105] S. Panzieri, F. Pascucci, and R. Setola. Multirobot localization using interlaced extended kalman filter. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2816–2821, Beijing, China, Oct. 9–15 2006.

[106] N. Karam, F. Chausse, R. Aufrere, and R. Chapuis. Localization of a group of communicating vehicles by state exchange. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 519–524, Beijing, China, Oct. 9–15 2006.

[107] A. Martinelli. Improving the precision on multi robot localization by using a series of filters hierarchically distributed. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1053–1058, San Diego, CA, Oct. 9–Nov. 2 2007.

[108] A. Howard, M. J. Matarić, and G. S. Sukhatme. Localization for mobile robot teams: A distributed MLE approach. In *Experimental Robotics VIII*, volume 5, pages 146–155. 2003.

[109] E. D. Nerurkar and S. I. Roumeliotis. Asynchronous multi-centralized cooperative localization. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4352–4359, Taipei, Taiwan, Oct. 18–22 2010.

[110] D. P. Bertsekas and J. N. Tsitsiklis. *Parallel and distributed computation: Numerical methods*. Prentice-Hall Inc., 1989.

[111] N. Trawny, S. I. Roumeliotis, and G. B. Giannakis. Cooperative multi-robot localization under communication constraints. In *Proc. of the IEEE International Conference on Robotics and Automation*, pages 4394–4400, Kobe, Japan, May 12–17 2009.

[112] A. Bahr. *Cooperative Localization for Autonomous Underwater Vehicles*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, Feb. 2009.

[113] L. E. Parker. Distributed algorithms for multi-robot observation of multiple moving targets. *Autonomous Robots*, 12:231–255, May 2002.

[114] A. Ribeiro and G. B. Giannakis. Bandwidth-constrained distributed estimation for wireless sensor networks-part I: Gaussian case. *IEEE Transactions on Signal Processing*, 54(3):1131–1143, Mar. 2006.

[115] Z. Q. Luo. An isotropic universal decentralized estimation scheme for a bandwidth constrained ad hoc sensor network. *IEEE Journal on Selected Areas in Communications*, 23(4):735–744, Apr. 2005.

[116] H. C. Papadopoulos, G. W. Wornell, and A. V. Oppenheim. Sequential signal encoding from noisy measurements using quantizers with dynamic bias control. *IEEE Transactions on Information Theory*, 47(3):978–1002, Mar. 2001.

[117] F. A. Shah, A. Ribeiro, and G. B. Giannakis. Bandwidth-constrained MAP estimation for wireless sensor networks. In *Proc. of 39th Asilomar Conference on Signals, Systems and Computers*, pages 215–219, Pacific Grove, CA, Oct. 28–Nov. 1 2005.

[118] W. M. Lam and A. R. Reibman. Design of quantizers for decentralized estimation systems. *IEEE Transactions on Communications*, 41(11):1602–1605, Nov. 1993.

[119] J. A. Gubner. Distributed estimation and quantization. *IEEE Transactions on Information Theory*, 39(4):1456–1459, Jul. 1993.

[120] E. J. Msechu, S. I. Roumeliotis, A. Ribeiro, and G. B. Giannakis. Distributed iteratively quantized kalman filtering for wireless sensor networks. In *Proc. of the 41st Asilomar Conference on Signals, Systems and Computers*, pages 646–650, Pacific Grove, CA, Nov. 4–7 2007.

[121] E. J. Msechu, A. Ribeiro, S. I. Roumeliotis, and G. B. Giannakis. Distributed kalman filtering based on quantized innovations. In *Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 3293–3296, Las Vegas, NV, Mar. 31–Apr. 4 2008.

[122] N. Trawny and S. I. Roumeliotis. Cooperative localization under communication constraints. Technical report, Dept. of Comp. Sci. & Eng., University of Minnesota, 2008. [Online] Available: `http://www-users.cs.umn.edu/~trawny/Publications/Trawny_TR_Quant.pdf`.

[123] S. Lloyd. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137, Mar. 1982.

[124] J. Max. Quantizing for minimum distortion. *IRE Transactions on Information Theory*, 6(1):7–12, Mar. 1960.

[125] E. D. Nerurkar, K. X. Zhou, and S. I. Roumeliotis. A hybrid estimation framework for cooperative localization under communication constraints. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 502–509, San Francisco, CA, Sep. 25–30 2011.

[126] A. Prekopa. Logarithmic concave measures and related topics. In *Stochastic Programming*, pages 63–82. Academic Press, 1980.

[127] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, New York, NY, 2004.

[128] M. Li, B. H. Kim, and A. I. Mourikis. Real-time motion tracking on a cellphone using inertial sensing and a rolling-shutter camera. In *Proc. of the IEEE International Conference on Robotics and Automation*, pages 4697–4704, Karlsruhe, Germany, May 6–10 2013.

[129] A. B. Chatfield. *Fundamentals of high accuracy inertial navigation*. Progress in astronautics and aeronautics. American Institute of Aeronautics and Astronautics, Inc., Reston, VA, 1997.

[130] M. Li and A. I. Mourikis. 3-D motion estimation and online temporal calibration for camera-IMU systems. In *Proc. of the IEEE International Conference on Robotics and Automation*, pages 5689–5696, Karlsruhe, Germany, May 6–10 2013.

[131] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. ORB: An efficient alternative to SIFT or SURF. In *Proc. of the IEEE International Conference on Computer Vision*, pages 2564–2571, Barcelona, Spain, Nov 6–13 2011.

[132] P. Vandergheynst A. Alahi, R. Ortiz. FREAK: Fast Retina Keypoint. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 510–517, Rhode Island, Providence, Jun. 16–21 2012.

[133] M. Calonder, V. Lepetit, C. Strecha, and P. Fua. BRIEF: Binary Robust Independent Elementary Features. In *Proc. of the 11th European Conference on Computer Vision*, pages 778–792, Crete, Greece, Sep. 5–11 2010.

[134] B. Grocholsky, J. Keller, V. Kumar, and G. Pappas. Cooperative air and ground surveillance. *IEEE Robotics and Automation Magazine*, 13(3):16–25, Sep. 2006.

[135] V. Kumar and N. Michael. Opportunities and challenges with autonomous micro aerial vehicles. *International Journal of Robotics Research*, 31(11):1279–1291, Sep. 2012.

[136] N. Cressie. The origins of kriging. *Mathematical Geology*, 22:239–252, 1990. 10.1007/BF00889887.

[137] A. Papoulis. *Probability, random variables, and stochastic processes*. McGraw-Hill, New York, third edition, 1991.

Appendix A

Appendices for Chapter 4

A.1 Derivation of $\mathbf{D}_1^*$ and $\mathbf{c}_1^*$

From (4.38), (4.39) and Figs. 4.4, 4.5 we see that $\mathbf{D}_1^* = (\mathbf{A}_{new}^T \mathbf{A}_{new})_{(1:N)\times(1:N)}$ and $\mathbf{c}_1^* = (\mathbf{A}_{new}^T \mathbf{b}_{new})_{(1:N)\times 1}$. Therefore,

$$\begin{aligned}
\mathbf{D}_1^* &= (\mathbf{A}_{new}^T \mathbf{A}_{new})_{(1:N)\times(1:N)} \\
&= \mathbf{E}_{trunc}^T \mathbf{B}^{-T} \big\{ (\mathbf{P}_0)^{-1} + \overline{\mathbf{F}}_0{}^T \overline{\mathbf{F}}_0 + \overline{\mathbf{H}}_0{}^T \overline{\mathbf{H}}_0 \big\} \mathbf{B}^{-1} \mathbf{E}_{trunc} \\
&\quad + (\mathbf{Q}_0)^{-1} + \big\{ \overline{\mathbf{F}}_1{}^T \overline{\mathbf{F}}_1 + \overline{\mathbf{H}}_1{}^T \overline{\mathbf{H}}_1 \big\} \\
&\quad + \big((\mathbf{Q}_0)^{-T/2} \overline{\mathbf{F}}_0 \big) \mathbf{B}^{-1} \mathbf{E}_{trunc} + \mathbf{E}_{trunc}^T \mathbf{B}^{-T} \big(\overline{\mathbf{F}}_0{}^T (\mathbf{Q}_0)^{-1/2} \big).
\end{aligned}$$

From (4.27) and (4.28) we obtain:

$$\begin{aligned}
\mathbf{D}_1^* &= \mathbf{E}_{trunc}^T \mathbf{B}^{-T} \mathbf{B} \mathbf{B}^{-1} \mathbf{E}_{trunc} + \mathbf{D}_{1C} + \mathbf{D}_{1M} \\
&\quad - \mathbf{E}_{trunc}^T \mathbf{B}^{-1} \mathbf{E}_{trunc} - \mathbf{E}_{trunc}^T \mathbf{B}^{-T} \mathbf{E}_{trunc}.
\end{aligned}$$

Since $\mathbf{B}$ is a symmetric matrix we obtain:

$$\begin{aligned}
\mathbf{D}_1^* &= \mathbf{D}_{1M} + \mathbf{D}_{1C} - \mathbf{E}_{trunc}^T \mathbf{B}^{-1} \mathbf{E}_{trunc} \\
&= \mathbf{D}_{1M} + \mathbf{D}_{mod},
\end{aligned}$$

where $\mathbf{D}_{mod} = \mathbf{D}_{1C} - \mathbf{E}_{trunc}^T \mathbf{B}^{-1} \mathbf{E}_{trunc}$. Similarly deriving for $\mathbf{c}_1^*$,

$$
\begin{aligned}
\mathbf{c}_1^* &= (\mathbf{A}_{new}^T \mathbf{b}_{new})_{(1:N)\times 1} \\
&= \mathbf{E}_{trunc}^T \mathbf{B}^{-T} \big\{ (\mathbf{P}_0)^{-1} + \overline{\mathbf{F}}_0^T \overline{\mathbf{F}}_0 + \overline{\mathbf{H}}_0^T \overline{\mathbf{H}}_0 \big\} \mathbf{B}^{-1} \mathbf{c}_0 \\
&+ \mathbf{E}_{trunc}^T \mathbf{B}^{-T} \big\{ -\overline{\mathbf{F}}_0^T (\mathbf{Q}_0)^{-1/2} (\hat{\mathbf{x}}_{1M} - \hat{\mathbf{x}}_{0M} - \mathbf{f}(\hat{\mathbf{x}}_{0M}, \mathbf{u}_{m_0}) \delta t) \\
&+ (\mathbf{P}_0)^{-1} (\hat{\mathbf{x}}_{0M} - \mathbf{x}_{init}) - \overline{\mathbf{H}}_0^T (\mathbf{R}_0)^{-1/2} (\mathbf{z}_0 - \mathbf{h}(\hat{\mathbf{x}}_{0M})) \big\} \\
&- \mathbf{E}_{trunc}^T \mathbf{B}^{-T} \big(\overline{\mathbf{F}}_0^T (\mathbf{Q}_0)^{-1/2} \big) (\hat{\mathbf{x}}_1 - \hat{\mathbf{x}}_{1M}) \\
&+ \big((\mathbf{Q}_0)^{-T/2} \overline{\mathbf{F}}_0 \big) \mathbf{B}^{-1} \mathbf{c}_0 - (\mathbf{Q}_0)^{-1} (\hat{\mathbf{x}}_{1M} - \hat{\mathbf{x}}_{0M} - \mathbf{f}(\hat{\mathbf{x}}_{0M}, \mathbf{u}_{m_0}) \delta t) \\
&- (\mathbf{Q}_0)^{-1} (\hat{\mathbf{x}}_1 - \hat{\mathbf{x}}_{1M}) + \overline{\mathbf{H}}_1^T (\mathbf{R}_1)^{-1/2} (\mathbf{z}_1 - \mathbf{h}(\hat{\mathbf{x}}_1)) \\
&+ \overline{\mathbf{F}}_1^T (\mathbf{Q}_1)^{-1/2} (\hat{\mathbf{x}}_2 - \hat{\mathbf{x}}_1 - \mathbf{f}(\hat{\mathbf{x}}_1, \mathbf{u}_{m_1}) \delta t).
\end{aligned}
$$

From (4.27), (4.28), (4.29), (4.30), and (4.31) we get:

$$
\begin{aligned}
\mathbf{c}_1^* &= \mathbf{E}_{trunc}^T \mathbf{B}^{-T} \mathbf{B} \mathbf{B}^{-1} \mathbf{c}_0 - \mathbf{E}_{trunc}^T \mathbf{B}^{-T} \mathbf{c}_0 \\
&\quad + \mathbf{E}_{trunc}^T \mathbf{B}^{-T} \mathbf{E}_{trunc} (\hat{\mathbf{x}}_1 - \hat{\mathbf{x}}_{1M}) \\
&\quad - \mathbf{E}_{trunc}^T \mathbf{B}^{-1} \mathbf{c}_0 + \mathbf{c}_{1C} - \mathbf{D}_{1C} (\hat{\mathbf{x}}_1 - \hat{\mathbf{x}}_{1M}) + \mathbf{c}_{1M} \\
&= \mathbf{c}_{1M} - \big\{ \mathbf{D}_{1C} - \mathbf{E}_{trunc}^T \mathbf{B}^{-T} \mathbf{E}_{trunc} \big\} (\hat{\mathbf{x}}_1 - \hat{\mathbf{x}}_{1M}) \\
&\quad + \mathbf{c}_{1C} - \mathbf{E}_{trunc}^T \mathbf{B}^{-T} \mathbf{c}_0.
\end{aligned}
$$

Simplifying the above equation and using (4.40) we obtain:

$$
\mathbf{c}_1^* = \mathbf{c}_{1M} - \mathbf{D}_{mod} (\hat{\mathbf{x}}_1 - \hat{\mathbf{x}}_{1M}) + \mathbf{c}_{mod}
$$

where $\mathbf{c}_{mod} = \mathbf{c}_{1C} - \mathbf{E}_{trunc}^T \mathbf{B}^{-1} \mathbf{c}_0$.

A.2 Derivation of $\mathbf{D}_p^*$ and $\mathbf{c}_p^*$

Consider a team of N robots over $r+1$ time steps with a complete measurement graph at every time step. We marginalize robots' poses from the first p time steps, i.e., we marginalize time-steps 0 to $p-1$ where $r+1 > p$. Fig. A.1 shows the structure of $\mathbf{H}$ and $\mathbf{e}$ for the first $p+1$ time steps. The $N \times N$ diagonal block and the $N \times N$ off-diagonal block (which has a sparse diagonal structure) of $\mathbf{H}$, corresponding to time-step k are denoted by $\mathbf{B}_k$ and $\mathbf{E}_t^k$, respectively, for $k = 0, \ldots, (p-1)$. Comparing with Fig. 4.3, it can be seen that $\mathbf{B}_0 = \mathbf{B}$, $\mathbf{B}_1 = \mathbf{D}_1$ and $\mathbf{E}_t^0 = \mathbf{E}_{trunc}$. The $N \times N$ diagonal block of $\mathbf{H}$, corresponding to time-step p is denoted by $\mathbf{D}_p$. Similarly, the $N \times 1$ sub-vector of $\mathbf{e}$,

corresponding to time-step k, is denoted by $\mathbf{c}_k$, for $k = 0, \ldots, p$. The corresponding cost function linearized about the latest robots' pose estimates at the time of marginalization, $\hat{\mathbf{x}}_{kM}$, where $k = 0, \ldots, r$, is given by:

$$\begin{aligned}\eta = &\sum_{k=0}^{r-1} \big\{ ||(\mathbf{Q}_k)^{-1/2}\delta\mathbf{x}_{k+1} - \overline{\mathbf{F}}_k\delta\mathbf{x}_k \\ &\qquad + (\mathbf{Q}_k)^{-1/2}(\hat{\mathbf{x}}_{(k+1)_M} - \hat{\mathbf{x}}_{kM} - \mathbf{f}(\hat{\mathbf{x}}_{kM}, \mathbf{u}_{m_k})\delta t)||_2^2 \big\} \\ &+ \sum_{k=0}^{r} || - \overline{\mathbf{H}}_k\delta\mathbf{x}_k + (\mathbf{R}_k)^{-1/2}(\mathbf{z}_k - \mathbf{h}(\hat{\mathbf{x}}_{kM}))||_2^2 \\ &+ ||(\mathbf{P}_0)^{-1/2}\delta\mathbf{x}_0 + (\mathbf{P}_0)^{-1/2}(\hat{\mathbf{x}}_{0M} - \mathbf{x}_{init})||_2^2. \end{aligned} \tag{A.1}$$

In order to marginalize the first p time steps, we differentiate the cost function in (A.1) w.r.t $\delta\mathbf{x}_0, \ldots, \delta\mathbf{x}_{p-1}$ to obtain:

$$\begin{aligned}\frac{\partial\eta}{\partial\delta\mathbf{x}_k} = &-2\overline{\mathbf{F}}_k{}^T\big\{(\mathbf{Q}_k)^{-1/2}\delta\mathbf{x}_{k+1} - \overline{\mathbf{F}}_k\delta\mathbf{x}_k \\ &\qquad + (\mathbf{Q}_k)^{-1/2}(\hat{\mathbf{x}}_{(k+1)_M} - \hat{\mathbf{x}}_{kM} - \mathbf{f}(\hat{\mathbf{x}}_{kM}, \mathbf{u}_{m_k})\delta t)\big\} \\ &- 2\overline{\mathbf{H}}_k^T\big(- \overline{\mathbf{H}}_k\delta\mathbf{x}_k + (\mathbf{R}_k)^{-1/2}(\mathbf{z}_k - \mathbf{h}(\hat{\mathbf{x}}_{kM}))\big) \\ &+ 2(\mathbf{Q}_{(k-1)})^{-T/2}\big\{(\mathbf{Q}_{(k-1)})^{-1/2}\delta\mathbf{x}_k - \overline{\mathbf{F}}_{(k-1)}\delta\mathbf{x}_{(k-1)} \\ &+ (\mathbf{Q}_{(k-1)})^{-1/2}(\hat{\mathbf{x}}_{kM} - \hat{\mathbf{x}}_{(k-1)_M} - \mathbf{f}(\hat{\mathbf{x}}_{(k-1)_M}, \mathbf{u}_{m_{(k-1)}})\delta t)\big\} \end{aligned} \tag{A.2}$$

where $k = 1, \ldots, (p-1)$. For $k = 0$, the derivation is same as that in Section 4.5.3 [see (4.33)]. Simplifying and equating (A.2) to zero leads to:

$$\begin{aligned}&\big(\overline{\mathbf{F}}_k{}^T\overline{\mathbf{F}}_k + \overline{\mathbf{H}}_k{}^T\overline{\mathbf{H}}_k + (\mathbf{Q}_{(k-1)})^{-1}\big)\delta\mathbf{x}_k \\ &- \big(\overline{\mathbf{F}}_k{}^T(\mathbf{Q}_k)^{-1/2}\big)\delta\mathbf{x}_{(k+1)} - \big((\mathbf{Q}_{(k-1)})^{-T/2}\overline{\mathbf{F}}_{(k-1)}\big)\delta\mathbf{x}_{(k-1)} \\ &- \overline{\mathbf{F}}_k{}^T(\mathbf{Q}_k)^{-1/2}(\hat{\mathbf{x}}_{(k+1)_M} - \hat{\mathbf{x}}_{kM} - \mathbf{f}(\hat{\mathbf{x}}_{kM}, \mathbf{u}_{m_k})\delta t) \\ &- \overline{\mathbf{H}}_k{}^T(\mathbf{R}_k)^{-1/2}(\mathbf{z}_k - \mathbf{h}(\hat{\mathbf{x}}_{kM})) \\ &+ (\mathbf{Q}_{(k-1)})^{-1}(\hat{\mathbf{x}}_{kM} - \hat{\mathbf{x}}_{(k-1)_M} - \mathbf{f}(\hat{\mathbf{x}}_{(k-1)_M}, \mathbf{u}_{m_{(k-1)}})\delta t) = \mathbf{0}. \end{aligned} \tag{A.3}$$

With reference to Fig. A.1, we see that the term $\big(\overline{\mathbf{F}}_k{}^T\overline{\mathbf{F}}_k + \overline{\mathbf{H}}_k{}^T\overline{\mathbf{H}}_k + (\mathbf{Q}_{(k-1)})^{-1}\big)$ corresponds to $\mathbf{B}_k$. From (4.28), we see that $-\big((\mathbf{Q}_{(k-1)})^{-T/2}\overline{\mathbf{F}}_{(k-1)}\big)$ and $-\big(\overline{\mathbf{F}}_k{}^T(\mathbf{Q}_{(k)})^{-1/2}\big)$ correspond to $(\mathbf{E}_t^{(k-1)})^T$ and $\mathbf{E}_t^k$ respectively. Finally from (4.30), (4.31), we see that $\big[- \overline{\mathbf{F}}_k{}^T(\mathbf{Q}_k)^{-1/2}(\hat{\mathbf{x}}_{(k+1)_M} - \hat{\mathbf{x}}_{kM} - \mathbf{f}(\hat{\mathbf{x}}_{kM}, \mathbf{u}_{m_k})\delta t) - \overline{\mathbf{H}}_k{}^T(\mathbf{R}_i)^{-1/2}(\mathbf{z}_k - \mathbf{h}(\hat{\mathbf{x}}_{kM})) +$

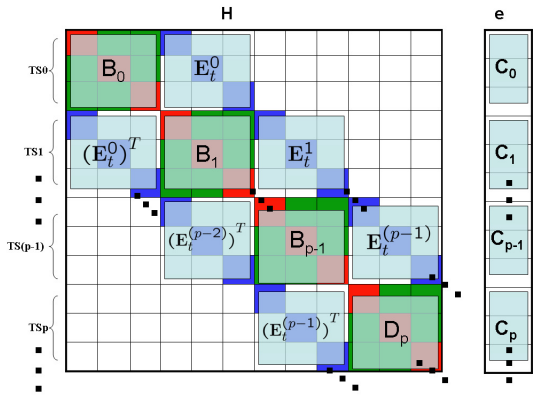


Figure A.1: Structure of $\mathbf{H}$ and $\mathbf{e}$ for the first $p+1$ time steps.

$(\mathbf{Q}_{(k-1)})^{-1}(\hat{\mathbf{x}}_{kM} - \hat{\mathbf{x}}_{(k-1)_M} - \mathbf{f}(\hat{\mathbf{x}}_{(k-1)_M}, \mathbf{u}_{m_{(k-1)}})\delta t)\big]$ corresponds to $-\mathbf{c}_k$. Therefore, from (A.3) and (4.33) we get:

$$\delta\mathbf{x}_k = (\mathbf{B}_k^*)^{-1}\mathbf{c}_k^* - (\mathbf{B}_k^*)^{-1}\mathbf{E}_t^k\delta\mathbf{x}_{(k+1)}, \tag{A.4}$$

where,

$$\mathbf{B}_k^* = \mathbf{B}_k - (\mathbf{E}_t^{(k-1)})^T(\mathbf{B}_{(k-1)}^*)^{-1}\mathbf{E}_t^{(k-1)} \tag{A.5}$$

$$\mathbf{c}_k^* = \mathbf{c}_k - (\mathbf{E}_t^{(k-1)})^T(\mathbf{B}_{(k-1)}^*)^{-1}\mathbf{c}_{(k-1)}^* \tag{A.6}$$

$$\mathbf{B}_0^* = \mathbf{B}_0, \mathbf{c}_0^* = \mathbf{c}_0. \tag{A.7}$$

In (A.4) we showed that $\delta\mathbf{x}_k$ can be expressed as a function of $\delta\mathbf{x}_{(k+1)}$. Thus, $\delta\mathbf{x}_k$ can also be expressed as a function of $\delta\mathbf{x}_p$ as follows:

$$\begin{aligned}\delta\mathbf{x}_k = (\mathbf{B}_k^*)^{-1}\mathbf{c}_k^* + (-1)^{(p-k)}\Big(\prod_{m=k}^{p-1}(\mathbf{B}_m^*)^{-1}\mathbf{E}_t^m\Big)\delta\mathbf{x}_p \\ + \sum_{l=k+1}^{p-1}\Big\{(-1)^{(l-k)}\Big(\prod_{m=k}^{l-1}(\mathbf{B}_m^*)^{-1}\mathbf{E}_t^m\Big)(\mathbf{B}_l^*)^{-1}\mathbf{c}_l^*\Big\}.\end{aligned} \tag{A.8}$$

Let us now consider the term $\big((\mathbf{Q}_k)^{-1/2}\delta\mathbf{x}_{k+1} - \overline{\mathbf{F}}_k\delta\mathbf{x}_k\big)$ for $k = 0, \ldots, (p-2)$ in (A.1). Using (A.4) and (A.8), it can be re-written as:

$$
\begin{aligned}
&(\mathbf{Q}_k)^{-1/2}\delta\mathbf{x}_{k+1} - \overline{\mathbf{F}}_k\delta\mathbf{x}_k \\
&= \big((\mathbf{Q}_k)^{-1/2} + \overline{\mathbf{F}}_k(\mathbf{B}_k^*)^{-1}\mathbf{E}_t^k\big)\delta\mathbf{x}_{k+1} - \overline{\mathbf{F}}_k(\mathbf{B}_k^*)^{-1}\mathbf{c}_k^* \\
&= \big((\mathbf{Q}_k)^{-1/2} + \overline{\mathbf{F}}_k(\mathbf{B}_k^*)^{-1}\mathbf{E}_t^k\big)(-1)^{(p-k-1)}\big(\prod_{m=k+1}^{p-1}(\mathbf{B}_m^*)^{-1}\mathbf{E}_t^m\big)\delta\mathbf{x}_p \\
&\quad + \big((\mathbf{Q}_k)^{-1/2} + \overline{\mathbf{F}}_k(\mathbf{B}_k^*)^{-1}\mathbf{E}_t^k\big)\Bigg((\mathbf{B}_{(k+1)}^*)^{-1}\mathbf{c}_{(k+1)}^* \\
&\qquad + \sum_{l=k+2}^{p-1}\big\{(-1)^{(l-k-1)}\big(\prod_{m=k+1}^{l-1}(\mathbf{B}_m^*)^{-1}\mathbf{E}_t^m\big)(\mathbf{B}_l^*)^{-1}\mathbf{c}_l^*\big\}\Bigg) \\
&\quad - \overline{\mathbf{F}}_k(\mathbf{B}_k^*)^{-1}\mathbf{c}_k^* \\
&\triangleq \mathbf{L}_k\delta\mathbf{x}_p + \mathbf{r}_k.
\end{aligned} \tag{A.9}
$$

Similarly, the term $-\overline{\mathbf{H}}_k{}^T\delta\mathbf{x}_k$ in (A.1) for $k = 0, \ldots, (p-1)$ can be re-written as:

$$
\begin{aligned}
&-\overline{\mathbf{H}}_k{}^T\delta\mathbf{x}_k \\
&= -\overline{\mathbf{H}}_k{}^T(-1)^{(p-k)}\big(\prod_{m=k}^{p-1}(\mathbf{B}_m^*)^{-1}\mathbf{E}_t^m\big)\delta\mathbf{x}_p - \overline{\mathbf{H}}_k{}^T\Bigg((\mathbf{B}_k^*)^{-1}\mathbf{c}_k^* \\
&\quad + \sum_{l=k+1}^{p-1}\big\{(-1)^{(l-k)}\big(\prod_{m=k}^{l-1}(\mathbf{B}_m^*)^{-1}\mathbf{E}_t^m\big)(\mathbf{B}_l^*)^{-1}\mathbf{c}_l^*\Bigg) \\
&\triangleq \mathbf{S}_k\delta\mathbf{x}_p + \mathbf{t}_k.
\end{aligned} \tag{A.10}
$$

Finally the term $(\mathbf{P}_0)^{-1/2}\delta\mathbf{x}_0$ can also be re-written as:

$$
\begin{aligned}
&(\mathbf{P}_0)^{-1/2}\delta\mathbf{x}_0 \\
&= (\mathbf{P}_0)^{-1/2}(-1)^{(p)}\big(\prod_{m=0}^{p-1}(\mathbf{B}_m^*)^{-1}\mathbf{E}_t^m\big)\delta\mathbf{x}_p + (\mathbf{P}_0)^{-1/2}\Bigg((\mathbf{B}_0^*)^{-1}\mathbf{c}_0^* \\
&\quad + \sum_{l=1}^{p-1}\big\{(-1)^{(l)}\big(\prod_{m=0}^{l-1}(\mathbf{B}_m^*)^{-1}\mathbf{E}_t^m\big)(\mathbf{B}_l^*)^{-1}\mathbf{c}_l^*\big\}\Bigg) \\
&\triangleq \mathbf{O}\delta\mathbf{x}_p + \mathbf{p}.
\end{aligned} \tag{A.11}
$$

Substituting (A.9) and (A.10) into (A.1) and using the same idea of rearranging terms as in (4.37), we obtain the marginalized cost function as follows:

$$
\begin{aligned}
&\eta_{new} = \\
&\sum_{k=0}^{p-2} ||\mathbf{L}_k \delta\mathbf{x}_p + \mathbf{r}_k + (\mathbf{Q}_k)^{-1/2}(\hat{\mathbf{x}}_{(k+1)\,M} - \hat{\mathbf{x}}_{k\,M} - \mathbf{f}(\hat{\mathbf{x}}_{k\,M}, \mathbf{u}_{m_k})\delta t)||_2^2 \\
&+ ||\big((\mathbf{Q}_{(p-1)})^{-1/2} + \overline{\mathbf{F}}_{(p-1)}(\mathbf{B}^*_{(p-1)})^{-1}\mathbf{E}_t^{(p-1)}\big)\delta\mathbf{x}_p \\
&\quad - \overline{\mathbf{F}}_{(p-1)}(\mathbf{B}^*_{(p-1)})^{-1}\mathbf{c}^*_{(p-1)} + (\mathbf{Q}_{(p-1)})^{-1/2}(\hat{\mathbf{x}}_p - \hat{\mathbf{x}}_{p\,M}) \\
&\quad + (\mathbf{Q}_{(p-1)})^{-1/2}(\hat{\mathbf{x}}_{p\,M} - \hat{\mathbf{x}}_{(p-1)\,M} - \mathbf{f}(\hat{\mathbf{x}}_{(p-1)\,M}, \mathbf{u}_{m_{(p-1)}})\delta t)||_2^2 \\
&+ \sum_{k=p}^{r-1} \big\{||(\mathbf{Q}_k)^{-1/2}\delta\mathbf{x}_{k+1} - \overline{\mathbf{F}}_k\delta\mathbf{x}_k \\
&\qquad + (\mathbf{Q}_k)^{-1/2}(\hat{\mathbf{x}}_{(k+1)} - \hat{\mathbf{x}}_k - \mathbf{f}(\hat{\mathbf{x}}_k, \mathbf{u}_{m_k})\delta t)||_2^2\big\} \\
&+ \sum_{k=0}^{p-1} ||\mathbf{S}_k\delta\mathbf{x}_p + \mathbf{t}_k + (\mathbf{R}_k)^{-1/2}(\mathbf{z}_k - \mathbf{h}(\hat{\mathbf{x}}_{k\,M}))||_2^2 \\
&+ \sum_{k=p}^{r} ||-\overline{\mathbf{H}}_k\delta\mathbf{x}_k + (\mathbf{R}_k)^{-1/2}(\mathbf{z}_k - \mathbf{h}(\hat{\mathbf{x}}_k))||_2^2 \\
&+ ||\mathbf{O}\delta\mathbf{x}_p + \mathbf{p} + (\mathbf{P}_0)^{-1/2}(\hat{\mathbf{x}}_{0\,M} - \mathbf{x}_{init})||_2^2 \\
&= ||\mathbf{A}_{new}\delta\mathbf{x} - \mathbf{b}_{new}||_2^2. \qquad \text{(A.12)}
\end{aligned}
$$

Thus the new cost function no longer contains terms involving $\delta\mathbf{x}_0$ to $\delta\mathbf{x}_{(p-1)}$. The system of normal equations [see (4.21)] corresponding to the marginalized cost function is $(\mathbf{H}_{new} + \lambda\mathbf{I})\ [\delta{\mathbf{x}_p}^T, \ldots, \delta{\mathbf{x}_r}^T]^T = \mathbf{e}_{new}$. The correlations between time-steps $p-1$ and p introduce additional terms in $\mathbf{D}_p$ and $\mathbf{c}_p$ which are denoted by $\mathbf{D}^*_p$ and $\mathbf{c}^*_p$ respectively. These quantities are defined as follows:

$$\mathbf{D}^*_p = \big(\mathbf{A}^T_{new}\mathbf{A}_{new}\big)_{(1:N)\times(1:N)} = (\mathbf{A}^+_{new})^T\mathbf{A}^+_{new} \qquad \text{(A.13)}$$

$$\mathbf{c}^*_p = \big(\mathbf{A}^T_{new}\mathbf{b}_{new}\big)_{(1:N)\times 1} = (\mathbf{A}^+_{new})^T\mathbf{b}^+_{new} \qquad \text{(A.14)}$$

where,

$$\mathbf{A}^{+}_{new} = \begin{bmatrix} \mathbf{O} \\ \mathbf{L}_0 \\ \mathbf{S}_0 \\ \vdots \\ \mathbf{L}_{(p-2)} \\ \mathbf{S}_{(p-2)} \\ (\mathbf{Q}_{(p-1)})^{-1/2} + \overline{\mathbf{F}}_{(p-1)}(\mathbf{B}^{*}_{(p-1)})^{-1}\mathbf{E}^{(p-1)}_{t} \\ \mathbf{S}_{(p-1)} \\ -\overline{\mathbf{F}}_p \\ -\overline{\mathbf{H}}_p \end{bmatrix} \tag{A.15}$$

$$\mathbf{b}^{+}_{new} = \begin{bmatrix} -\mathbf{p} - (\mathbf{P}_0)^{-1/2}(\hat{\mathbf{x}}_{0\,M} - \mathbf{x}_{init}) \\ -\mathbf{r}_0 - (\mathbf{Q}_0)^{-1/2}(\hat{\mathbf{x}}_{1\,M} - \hat{\mathbf{x}}_{0\,M} - \mathbf{f}(\hat{\mathbf{x}}_{0\,M}, \mathbf{u}_{m_0})\delta t) \\ -\mathbf{t}_0 - (\mathbf{R}_0)^{-1/2}(\mathbf{z}_0 - \mathbf{h}(\hat{\mathbf{x}}_{0\,M})) \\ \vdots \\ \{ -\mathbf{r}_{(p-2)} - (\mathbf{Q}_{(p-2)})^{-1/2}(\hat{\mathbf{x}}_{(p-1)\,M} \\ -\hat{\mathbf{x}}_{(p-2)\,M} - \mathbf{f}(\hat{\mathbf{x}}_{(p-2)\,M}, \mathbf{u}_{m_{(p-2)}})\delta t)\} \\ -\mathbf{t}_{(p-2)} - (\mathbf{R}_{(p-2)})^{-1/2}(\mathbf{z}_{(p-2)} - \mathbf{h}(\hat{\mathbf{x}}_{(p-2)\,M})) \\ \{\overline{\mathbf{F}}_{(p-1)}(\mathbf{B}^{*}_{(p-1)})^{-1}\mathbf{c}^{*}_{(p-1)} - (\mathbf{Q}_{(p-1)})^{-1/2}(\hat{\mathbf{x}}_p - \hat{\mathbf{x}}_{p\,M}) \\ -(\mathbf{Q}_{(p-1)})^{-1/2}(\hat{\mathbf{x}}_{p\,M} - \hat{\mathbf{x}}_{(p-1)\,M} - \mathbf{f}(\hat{\mathbf{x}}_{(p-1)\,M}, \mathbf{u}_{m_{(p-1)}})\delta t)\} \\ -\mathbf{t}_{(p-1)} - (\mathbf{R}_{(p-1)})^{-1/2}(\mathbf{z}_{(p-1)} - \mathbf{h}(\hat{\mathbf{x}}_{(p-1)\,M})) \\ -(\mathbf{Q}_p)^{-1/2}(\hat{\mathbf{x}}_{(p+1)} - \hat{\mathbf{x}}_p - \mathbf{f}(\hat{\mathbf{x}}_p, \mathbf{u}_{m_p})\delta t) \\ -(\mathbf{R}_p)^{-1/2}(\mathbf{z}_p - \mathbf{h}(\hat{\mathbf{x}}_p)) \end{bmatrix} \tag{A.16}$$

We now present the derivation for $\mathbf{D}^{*}_{p}$ in detail. The derivation for $\mathbf{c}^{*}_{p}$ follows similarly.

$$\begin{aligned} \mathbf{D}^{*}_{p} &= \overline{\mathbf{H}}_p{}^{T}\overline{\mathbf{H}}_p + \overline{\mathbf{F}}_p{}^{T}\overline{\mathbf{F}}_p \\ &+ \sum_{k=0}^{p-1}\Bigg(\prod_{m=k}^{p-1}(\mathbf{B}^{*}_{m})^{-1}\mathbf{E}^{m}_{t}\Bigg)^{T}\big(\overline{\mathbf{H}}_k{}^{T}\overline{\mathbf{H}}_k\big)\Bigg(\prod_{m=k}^{p-1}(\mathbf{B}^{*}_{m})^{-1}\mathbf{E}^{m}_{t}\Bigg) \\ &+ (\mathbf{Q}_{(p-1)})^{-1} + (\mathbf{Q}_{(p-1)})^{-T/2}\overline{\mathbf{F}}_{(p-1)}(\mathbf{B}^{*}_{(p-1)})^{-1}\mathbf{E}^{(p-1)}_{t} \\ &+ (\mathbf{E}^{(p-1)}_{t})^{T}(\mathbf{B}^{*}_{(p-1)})^{-T}\overline{\mathbf{F}}^{T}_{(p-1)}(\mathbf{Q}_{(p-1)})^{-1/2} \\ &+ (\mathbf{E}^{(p-1)}_{t})^{T}(\mathbf{B}^{*}_{(p-1)})^{-T}\overline{\mathbf{F}}^{T}_{(p-1)}\overline{\mathbf{F}}_{(p-1)}(\mathbf{B}^{*}_{(p-1)})^{-1}\mathbf{E}^{(p-1)}_{t} \\ &+ \sum_{k=0}^{p-2}\Bigg(\prod_{m=k+1}^{p-1}(\mathbf{B}^{*}_{m})^{-1}\mathbf{E}^{m}_{t}\Bigg)^{T}\big((\mathbf{Q}_k)^{-1} \\ &\quad + (\mathbf{Q}_k)^{-T/2}\overline{\mathbf{F}}_k(\mathbf{B}^{*}_{k})^{-1}\mathbf{E}^{k}_{t} + (\mathbf{E}^{k}_{t})^{T}(\mathbf{B}^{*}_{k})^{-T}\overline{\mathbf{F}}^{T}_{k}(\mathbf{Q}_k)^{-1/2} \\ &\quad + (\mathbf{E}^{k}_{t})^{T}(\mathbf{B}^{*}_{k})^{-T}\overline{\mathbf{F}}^{T}_{k}\overline{\mathbf{F}}_k(\mathbf{B}^{*}_{k})^{-1}\mathbf{E}^{k}_{t}\Bigg)\Bigg(\prod_{m=k+1}^{p-1}(\mathbf{B}^{*}_{m})^{-1}\mathbf{E}^{m}_{t}\Bigg) \\ &+ \mathbf{O}^{T}\mathbf{O} \end{aligned} \tag{A.17}$$

But we know that $\mathbf{B}_i = \left(\overline{\mathbf{F}}_i{}^T\overline{\mathbf{F}}_i + \overline{\mathbf{H}}_i{}^T\overline{\mathbf{H}}_i + (\mathbf{Q}_{(i-1)})^{-1}\right)$ and from (4.28) and (A.5) we obtain:

$$\begin{aligned}\mathbf{D}_p^* = \mathbf{B}_p &- 2(\mathbf{E}_t^{(p-1)})^T(\mathbf{B}_{(p-1)}^*)^{-1}\mathbf{E}_t^{(p-1)} + (\mathbf{E}_t^{(p-1)})^T(\mathbf{B}_{(p-1)}^*)^{-T}\times \\ &\{\overline{\mathbf{H}}_{(p-1)}{}^T\overline{\mathbf{H}}_{(p-1)} + \overline{\mathbf{F}}_{(p-1)}{}^T\overline{\mathbf{F}}_{(p-1)}\}(\mathbf{B}_{(p-1)}^*)^{-1}\mathbf{E}_t^{(p-1)} \\ &+\sum_{k=0}^{p-2}\Big(\prod_{m=k+1}^{p-1}(\mathbf{B}_m^*)^{-1}\mathbf{E}_t^m\Big)^T\big((\mathbf{Q}_k)^{-1} \\ &-2(\mathbf{E}_t^k)^T(\mathbf{B}_k^*)^{-1}\mathbf{E}_t^k + (\mathbf{E}_t^k)^T(\mathbf{B}_k^*)^{-T}\{\overline{\mathbf{F}}_k^T\overline{\mathbf{F}}_k + \overline{\mathbf{H}}_k^T\overline{\mathbf{H}}_k\}\times \\ &(\mathbf{B}_k^*)^{-1}\mathbf{E}_t^k\Big)\Big(\prod_{m=k+1}^{p-1}(\mathbf{B}_m^*)^{-1}\mathbf{E}_t^m\Big) + \mathbf{O}^T\mathbf{O}. \end{aligned} \tag{A.18}$$

But we know that $\mathbf{B}_0^* = \mathbf{B}_0$ and we know the value of $\mathbf{B}_0$ from (4.27). Substituting the values of $\mathbf{B}_i^*$ recursively in the above equations yields the desired form of $\mathbf{D}_p^*$:

$$\begin{aligned}\mathbf{D}_p^* &= \mathbf{D}_p - (\mathbf{E}_t^{(p-1)})^T(\mathbf{B}_{(p-1)}^*)^{-1}(\mathbf{E}_t^{(p-1)}) \\ &= \mathbf{D}_{pM} + \mathbf{D}_{pC} - (\mathbf{E}_t^{(p-1)})^T(\mathbf{B}_{(p-1)}^*)^{-1}(\mathbf{E}_t^{(p-1)}) \\ &= \mathbf{D}_{pM} + \mathbf{D}_{mod}, \end{aligned} \tag{A.19}$$

where $\mathbf{D}_{mod} = \mathbf{D}_{pC} - (\mathbf{E}_t^{(p-1)})^T(\mathbf{B}_{(p-1)}^*)^{-1}(\mathbf{E}_t^{(p-1)})$.

Similarly, we can obtain $\mathbf{c}_p^*$:

$$\begin{aligned}\mathbf{c}_p^* &= \mathbf{c}_{pM} - \mathbf{D}_{mod}(\hat{\mathbf{x}}_p - \hat{\mathbf{x}}_{pM}) + \mathbf{c}_{pC} - (\mathbf{E}_t^{(p-1)})^T(\mathbf{B}_{(p-1)}^*)^{-1}\mathbf{c}_{(p-1)}^* \\ &= \mathbf{c}_{pM} - \mathbf{D}_{mod}(\hat{\mathbf{x}}_p - \hat{\mathbf{x}}_{pM}) + \mathbf{c}_{mod}, \end{aligned} \tag{A.20}$$

where $\mathbf{c}_{mod} = \mathbf{c}_{pC} - (\mathbf{E}_t^{(p-1)})^T(\mathbf{B}_{(p-1)}^*)^{-1}(\mathbf{c}_{p-1}^*)$.

Appendix B

Appendices for Chapter 6

B.1 Proof of Proposition 2

Proof. Taking the expectation of the process model in (6.1), conditioned on $\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{z}_{0:k-1}^{i}$, we obtain:

$$\hat{\mathbf{x}}_{k|k-1}^{H_i} := E[\mathbf{x}_k|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{z}_{0:k-1}^{i}] = \mathbf{F}_{k-1}E[\mathbf{x}_{k-1}|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{z}_{0:k-1}^{i}] = \mathbf{F}_{k-1}\hat{\mathbf{x}}_{k-1|k-1}^{H_i} \tag{B.1}$$

since the $E[\mathbf{w}_{k-1}|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{z}_{0:k-1}^{i}] = \mathbf{0}$. The corresponding covariance is given by:

$$\begin{aligned}
\mathbf{P}_{k|k-1}^{H_i} &:= E[(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k-1}^{H_i})(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k-1}^{H_i})^T|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{z}_{0:k-1}^{i}] \\
&= E[(\mathbf{F}_{k-1}(\mathbf{x}_{k-1} - \hat{\mathbf{x}}_{k-1|k-1}^{H_i}) + \mathbf{G}_{k-1}\mathbf{w}_{k-1}) \\
&\qquad \times (\mathbf{F}_{k-1}(\mathbf{x}_{k-1} - \hat{\mathbf{x}}_{k-1|k-1}^{H_i}) + \mathbf{G}_{k-1}\mathbf{w}_{k-1})^T|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{z}_{0:k-1}^{i}] \\
&= \mathbf{F}_{k-1}E[(\mathbf{x}_{k-1} - \hat{\mathbf{x}}_{k-1|k-1}^{H_i})(\mathbf{x}_{k-1} - \hat{\mathbf{x}}_{k-1|k-1}^{H_i})^T|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{z}_{0:k-1}^{i}]\mathbf{F}_{k-1}^T \\
&\qquad + \mathbf{G}_{k-1}E[\mathbf{w}_{k-1}\mathbf{w}_{k-1}^T]\mathbf{G}_{k-1}^T \\
&= \mathbf{F}_{k-1}\mathbf{P}_{k-1|k-1}^{H_i}\mathbf{F}_{k-1}^T + \mathbf{G}_{k-1}\mathbf{Q}_{k-1}\mathbf{G}_{k-1}^T
\end{aligned} \tag{B.2}$$

where, line 2 is obtained from (6.1) and (B.1). Line 3 is based on the fact that the noise $\mathbf{w}_{k-1}$ is zero-mean and independent of the measurements, $\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{z}_{0:k-1}^{i}$, and the state $\mathbf{x}_{k-1}$. Thus, we note that (B.1) and (B.2) are identical to the state/covariance propagation of the standard KF.

The derivation for the state/covariance update equations for robot i processing its

own analog measurement z_k^i, proceeds as follows:

$$E[\mathbf{x}_k|\mathbf{b}_{0:k-1}^{q\neq i},\mathbf{b}_k^{m\neq i},\mathbf{z}_{0:k}^i] = \int_{\mathbb{R}^r}\mathbf{x}_k p(\mathbf{x}_k|\mathbf{b}_{0:k-1}^{q\neq i},\mathbf{b}_k^{m\neq i},\mathbf{z}_{0:k}^i)d\mathbf{x}_k$$
$$= \int_{\mathbb{R}^r}\mathbf{x}_k\frac{p(z_k^i|\mathbf{x}_k,\mathbf{b}_{0:k-1}^{q\neq i},\mathbf{b}_k^{m\neq i},\mathbf{z}_{0:k-1}^i)p(\mathbf{x}_k|\mathbf{b}_{0:k-1}^{q\neq i},\mathbf{b}_k^{m\neq i},\mathbf{z}_{0:k-1}^i)}{p(z_k^i|\mathbf{b}_{0:k-1}^{q\neq i},\mathbf{b}_k^{m\neq i},\mathbf{z}_{0:k-1}^i)}d\mathbf{x}_k \tag{B.3}$$

In the above equation, $p(z_k^i|\mathbf{x}_k,\mathbf{b}_{0:k-1}^{q\neq i},\mathbf{b}_k^{m\neq i},\mathbf{z}_{0:k-1}^i)\sim\mathcal{N}\left(\mathbf{h}_k^{i^T}\mathbf{x}_k,\sigma_k^{i^2}\right)$ and $p(z_k^i|\mathbf{b}_{0:k-1}^{q\neq i},\mathbf{b}_k^{m\neq i},\mathbf{z}_{0:k-1}^i)\sim\mathcal{N}\left(\mathbf{h}_k^{i^T}\hat{\mathbf{x}}_{k|k,j-1}^{H_i},\mathbf{h}_k^{i^T}\mathbf{P}_{k|k,j-1}^{H_i}\mathbf{h}_k^i+\sigma_k^{i^2}\right)$. Therefore, all pdfs in (B.3) are Gaussian, similar to that of the KF. The derivation from this point onwards, is identical to that of the KF and can be found in [98].

To obtain the state/covariance update equations when processing quantized measurements received from other robots, we use the following concept of iterated expectation [137]:

$$E[g(\mathbf{x})|y\in\mathcal{R}_i] = E[E[g(\mathbf{x})|Y]|y\in\mathcal{R}_i] \tag{B.4}$$

where $g(\mathbf{x})$ is a function of the random variable $\mathbf{x}\in\mathbb{R}^r$, Y is a random variable in $\mathbb{R}$, and y is its realization. The proof of the above equality is as follows:

$$E[E[g(\mathbf{x})|Y]|y\in\mathcal{R}_i] = \int_{\mathbb{R}}\left[\int_{\mathbb{R}^r}g(\mathbf{x})p(\mathbf{x}|Y=\gamma)d\mathbf{x}\right]p(\gamma|y\in\mathcal{R}_i)d\gamma$$
$$= \int_{\mathcal{R}_i}\left[\int_{\mathbb{R}^r}g(\mathbf{x})p(\mathbf{x}|Y=\gamma)d\mathbf{x}\right]\frac{p(\gamma)}{\Pr\{y\in\mathcal{R}_i\}}d\gamma = \int_{\mathbb{R}^r}g(\mathbf{x})\left[\int_{\mathcal{R}_i}\frac{p(\mathbf{x},\gamma)}{\Pr\{y\in\mathcal{R}_i\}}d\gamma\right]d\mathbf{x}$$
$$= \int_{\mathbb{R}^r}g(\mathbf{x})p(\mathbf{x}|y\in\mathcal{R}_i)d\mathbf{x} = E[g(\mathbf{x})|y\in\mathcal{R}_i] \tag{B.5}$$

where

$$p(\gamma|y\in\mathcal{R}_i) = \begin{cases}\frac{p(\gamma)}{\Pr\{y\in\mathcal{R}_i\}} & \text{if } \gamma\in\mathcal{R}_i\\ 0, & \text{otherwise}\end{cases}$$
$$p(\mathbf{x}|y\in\mathcal{R}_i) = \int_{\mathcal{R}_i}\frac{p(\mathbf{x},\gamma)}{\Pr\{y\in\mathcal{R}_i\}}d\gamma \tag{B.6}$$

We define the random variable $\tilde{z}_k^j = z_k^j - E[z_k^j|\mathbf{b}_{0:k-1},\mathbf{b}_k^m]$. Therefore, from (6.6), we observe that b_k^j is a realization of the random variable $\tilde{z}_k^j$. When $b_k^j = n$, it implies $\mathcal{R}_k^j(n) := [\tau_k^j(n),\tau_k^j(n+1))$, i.e., $\tilde{z}_k^j\in\mathcal{R}_k^j(n)$. Therefore, when robot i is processing

the quantized measurement $b_k^j = n$ from robot j, $j \neq i$, the state update equation is obtained as follows (using the concept of iterated expectation):

$$E[\mathbf{x}_k|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i, b_k^j] = E[E[\mathbf{x}_k|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i, \tilde{z}_k^j]|b_k^j] \tag{B.7}$$

We will first evaluate the inner expectation in the above equation. For this, we compute the joint pdf $p(\mathbf{x}_k, \tilde{z}_k^j|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i)$ and then obtain the desired conditional pdf $p(\mathbf{x}_k|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i, \tilde{z}_k^j)$. The joint pdf is Gaussian and is calculated as follows:

$$\begin{aligned}
&p(\mathbf{x}_k|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i) \sim \mathcal{N}\left(\hat{\mathbf{x}}_{k|k,j-1}^{H_i}, \mathbf{P}_{k|k,j-1}^{H_i}\right) \\
&p(\tilde{z}_k^j|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i) \sim \mathcal{N}\left(\mathbf{h}_k^{j^T}(\hat{\mathbf{x}}_{k|k,j-1}^{H_i} - \hat{\mathbf{x}}_{k|k,j-1}^{Q}), \mathbf{h}_k^{j^T}\mathbf{P}_{k|k,j-1}^{H_i}\mathbf{h}_k^j + \sigma_k^{j^2}\right)
\end{aligned} \tag{B.8}$$

$$\begin{aligned}
&\mathrm{Cov}[\mathbf{x}_k, \tilde{z}_k^j|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i] \\
&\quad = E[(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k,j-1}^{H_i})(\tilde{z}_k^j - \mathbf{h}_k^{j^T}(\hat{\mathbf{x}}_{k|k,j-1}^{H_i} - \hat{\mathbf{x}}_{k|k,j-1}^{Q}))^T|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i] \\
&\quad = E[(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k,j-1}^{H_i})(\mathbf{h}_k^{j^T}(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k,j-1}^{H_i}) + v_k^j)^T|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i] \\
&\quad = \mathbf{P}_{k|k,j-1}^{H_i}\mathbf{h}_k^j
\end{aligned} \tag{B.9}$$

Therefore, the joint pdf is given by:

$$\begin{aligned}
&p(\mathbf{x}_k, \tilde{z}_k^j|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i) \\
&\quad \sim \mathcal{N}\left(\begin{bmatrix} \hat{\mathbf{x}}_{k|k,j-1}^{H_i} \\ \mathbf{h}_k^{j^T}(\hat{\mathbf{x}}_{k|k,j-1}^{H_i} - \hat{\mathbf{x}}_{k|k,j-1}^{Q}) \end{bmatrix}, \begin{bmatrix} \mathbf{P}_{k|k,j-1}^{H_i} & \mathbf{P}_{k|k,j-1}^{H_i}\mathbf{h}_k^j \\ \mathbf{h}_k^{j^T}\mathbf{P}_{k|k,j-1}^{H_i} & \mathbf{h}_k^{j^T}\mathbf{P}_{k|k,j-1}^{H_i}\mathbf{h}_k^j + \sigma_k^{j^2} \end{bmatrix}\right)
\end{aligned} \tag{B.10}$$

From the joint pdf in (B.10), we can obtain the mean and covariance of the conditional

pdf $p(\mathbf{x}_k|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i, \tilde{z}_k^j)$ as:

$$\begin{aligned}
&E[\mathbf{x}_k|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i, \tilde{z}_k^j] \\
&\quad = \hat{\mathbf{x}}_{k|k,j-1}^{H_i} + \frac{\mathbf{P}_{k|k,j-1}^{H_i}\mathbf{h}_k^j}{\mathbf{h}_k^{j^T}\mathbf{P}_{k|k,j-1}^{H_i}\mathbf{h}_k^j + \sigma_k^{j^2}}(\tilde{z}_k^j - \mathbf{h}_k^{j^T}(\hat{\mathbf{x}}_{k|k,j-1}^{H_i} - \hat{\mathbf{x}}_{k|k,j-1}^{Q})) \\
&\quad = \hat{\mathbf{x}}_{k|k,j-1}^{H_i} + \mathbf{k}^c(\tilde{z}_k^j - \mathbf{h}_k^{j^T}(\hat{\mathbf{x}}_{k|k,j-1}^{H_i} - \hat{\mathbf{x}}_{k|k,j-1}^{Q})) := \hat{\mathbf{x}}_{k|k,j}^{H_i^c} \qquad \text{(B.11)} \\
&\mathrm{Cov}[\mathbf{x}_k|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i, \tilde{z}_k^j] \\
&\quad = \mathbf{P}_{k|k,j-1}^{H_i} - \frac{\mathbf{P}_{k|k,j-1}^{H_i}\mathbf{h}_k^j\mathbf{h}_k^{j^T}P_{k|k,j-1}^{H_i}}{\mathbf{h}_k^{j^T}\mathbf{P}_{k|k,j-1}^{H_i}\mathbf{h}_k^j + \sigma_k^{j^2}} := \mathbf{P}_{k|k,j-1}^{H_i} - \mathbf{k}^c\mathbf{h}_k^{j^T}\mathbf{P}_{k|k,j-1}^{H_i} \qquad \text{(B.12)} \\
&\text{where,}\ \ \mathbf{k}^c = \frac{\mathbf{P}_{k|k,j-1}^{H_i}\mathbf{h}_k^j}{\mathbf{h}_k^{j^T}\mathbf{P}_{k|k,j-1}^{H_i}\mathbf{h}_k^j + \sigma_k^{j^2}} \qquad \text{(B.13)}
\end{aligned}$$

Substituting (B.11) in the inner expectation from (B.7), we obtain:

$$\begin{aligned}
&E[\mathbf{x}_k|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i, b_k^j] \\
&\quad = \hat{\mathbf{x}}_{k|k,j-1}^{H_i} + \mathbf{k}^c E[\tilde{z}_k^j|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i, b_k^j] - \mathbf{k}^c\mathbf{h}_k^{j^T}(\hat{\mathbf{x}}_{k|k,j-1}^{H_i} - \hat{\mathbf{x}}_{k|k,j-1}^{Q}) \qquad \text{(B.14)}
\end{aligned}$$

We need to evaluate $E[\tilde{z}_k^j|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i, b_k^j]$ in (B.14). The corresponding pdf can be expressed as:

$$\begin{aligned}
&p(\tilde{z}_k^j|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i, b_k^j = n) \\
&\quad = p(\tilde{z}_k^j|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i, \tilde{z}_k^j \in \mathcal{R}_i) \\
&\quad = \frac{p(\tilde{z}_k^j|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i)}{\Pr\{\tilde{z}_k^j \in \mathcal{R}_i|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i\}}\forall \tilde{z}_k^j \in \mathcal{R}_i, \text{and}\ \ 0\ \ \text{otherwise.} \qquad \text{(B.15)}
\end{aligned}$$

Therefore, in (B.14)

$$\begin{aligned}
&E[\tilde{z}_k^j|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i, b_k^j = n] \\
&\quad = \int_{\mathcal{R}_i} \tilde{z}_k^j \frac{p(\tilde{z}_k^j|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i)}{\Pr\{\tilde{z}_k^j \in \mathcal{R}_i|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i\}} d\tilde{z}_k^j \\
&\quad = \int_{\tau_k^j(n)}^{\tau_k^j(n+1)} \tilde{z}_k^j \frac{\mathcal{N}\left(\mathbf{h}_k^{j^T}(\hat{\mathbf{x}}_{k|k,j-1}^{H_i} - \hat{\mathbf{x}}_{k|k,j-1}^{Q}), \mathbf{h}_k^{j^T}\mathbf{P}_{k|k,j-1}^{H_i}\mathbf{h}_k^j + \sigma_k^{j^2}\right)}{\Pr\{\tau_k^j(n) \leq \tilde{z}_k^j < \tau_k^j(n+1)|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i)\}} \qquad \text{(B.16)}
\end{aligned}$$

Define $m := \mathbf{h}_k^{j^T}(\hat{\mathbf{x}}_{k|k,j-1}^{H_i} - \hat{\mathbf{x}}_{k|k,j-1}^{Q})$ and $\sigma_{H_i}^2 := \mathbf{h}_k^{j^T}\mathbf{P}_{k|k,j-1}^{H_i}\mathbf{h}_k^j + \sigma_k^{j^2}$. Also, since $\tilde{z}_k^j|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i$ is a Gaussian random variable with pdf given in (B.8), $\Pr\{\tau_k^j(n) \leq \tilde{z}_k^j < \tau_k^j(n+1)|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i)\} = \mathrm{Q}[\frac{\tau_k^j(n)-m}{\sigma_{H_i}}] - \mathrm{Q}[\frac{\tau_k^j(n+1)-m}{\sigma_{H_i}}]$. Here, $\mathrm{Q}[\cdot]$ is the Gaussian tail probability function with $\mathrm{Q}[x] \triangleq \frac{1}{\sqrt{2\pi}}\int_x^\infty \exp(-u^2/2)du$. Also, if $Y \sim \mathcal{N}(\mu, \sigma^2)$, then $\Pr\{Y > y\} = \mathrm{Q}[(y-\mu)/\sigma]$. Therefore, (B.16) can be written as:

$$
\begin{aligned}
&E[\tilde{z}_k^j|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i, b_k^j = n] \\
&= \frac{1}{\sqrt{2\pi}\sigma_{H_i}} \int_{\tau_k^j(n)}^{\tau_k^j(n+1)} \frac{\tilde{z}_k^j \exp(\frac{-(\tilde{z}_k^j-m)^2}{2\sigma_{H_i}^2})}{\mathrm{Q}[\frac{\tau_k^j(n)-m}{\sigma_{H_i}}] - \mathrm{Q}[\frac{\tau_k^j(n+1)-m}{\sigma_{H_i}}]} d\tilde{z}_k^j \\
&= \frac{\left(\sigma_{H_i}^2 \int_{\tau_k^j(n)}^{\tau_k^j(n+1)} \frac{(\tilde{z}_k^j-m)}{\sigma_{H_i}^2} \exp(-\frac{(\tilde{z}_k^j-m)^2}{2\sigma_{H_i}^2}) d\tilde{z}_k^j + m\int_{\tau_k^j(n)}^{\tau_k^j(n+1)} \exp(-\frac{(\tilde{z}_k^j-m)^2}{2\sigma_{H_i}^2}) d\tilde{z}_k^j\right)}{\sqrt{2\pi}\sigma_{H_i}\left(\mathrm{Q}[\frac{\tau_k^j(n)-m}{\sigma_{H_i}}] - \mathrm{Q}[\frac{\tau_k^j(n+1)-m}{\sigma_{H_i}}]\right)} \\
&= \frac{\left(\sigma_{H_i}^2 \int_{\tau_k^j(n)}^{\tau_k^j(n+1)} \frac{-d}{d\tilde{z}_k^j}(\exp(-\frac{(\tilde{z}_k^j-m)^2}{2\sigma_{H_i}^2}) d\tilde{z}_k^j + m\int_{\tau_k^j(n)}^{\tau_k^j(n+1)} \exp(-\frac{(\tilde{z}_k^j-m)^2}{2\sigma_{H_i}^2}) d\tilde{z}_k^j\right)}{\sqrt{2\pi}\sigma_{H_i}\left(\mathrm{Q}[\frac{\tau_k^j(n)-m}{\sigma_{H_i}}] - \mathrm{Q}[\frac{\tau_k^j(n+1)-m}{\sigma_{H_i}}]\right)} \\
&= \frac{\left(-\sigma_{H_i}^2 \left[\exp(-\frac{(\tilde{z}_k^j-m)^2}{2\sigma_{H_i}^2})\right]_{\tau_k^j(n)}^{\tau_k^j(n+1)} + m\int_{\tau_k^j(n)}^{\tau_k^j(n+1)} \exp(-\frac{(\tilde{z}_k^j-m)^2}{2\sigma_{H_i}^2}) d\tilde{z}_k^j\right)}{\sqrt{2\pi}\sigma_{H_i}\left(\mathrm{Q}[\frac{\tau_k^j(n)-m}{\sigma_{H_i}}] - \mathrm{Q}[\frac{\tau_k^j(n+1)-m}{\sigma_{H_i}}]\right)} \\
&= \frac{(\sigma_{H_i}/\sqrt{2\pi})\left(\exp(-\frac{(\tau_k^j(n)-m)^2}{2\sigma_{H_i}^2}) - \exp(-\frac{(\tau_k^j(n+1)-m)^2}{2\sigma_{H_i}^2})\right)}{\mathrm{Q}[\frac{\tau_k^j(n)-m}{\sigma_{H_i}}] - \mathrm{Q}[\frac{\tau_k^j(n+1)-m}{\sigma_{H_i}}]} \\
&\quad + \frac{m\left(\mathrm{Q}[\frac{\tau_k^j(n)-m}{\sigma_{H_i}}] - \mathrm{Q}[\frac{\tau_k^j(n+1)-m}{\sigma_{H_i}}]\right)}{\mathrm{Q}[\frac{\tau_k^j(n)-m}{\sigma_{H_i}}] - \mathrm{Q}[\frac{\tau_k^j(n+1)-m}{\sigma_{H_i}}]} \\
&= \frac{(\sigma_{H_i}/\sqrt{2\pi})\left(\exp(-\frac{(\tau_k^j(n)-m)^2}{2\sigma_{H_i}^2}) - \exp(-\frac{(\tau_k^j(n+1)-m)^2}{2\sigma_{H_i}^2})\right)}{\mathrm{Q}[\frac{\tau_k^j(n)-m}{\sigma_{H_i}}] - \mathrm{Q}[\frac{\tau_k^j(n+1)-m}{\sigma_{H_i}}]} + m
\end{aligned}
\tag{B.17}
$$

We obtained line 4 by using the equality $-\frac{d}{d\tilde{z}_k^j}\exp(-\frac{(\tilde{z}_k^j-m)^2}{2\sigma_{H_i}^2}) = \frac{(\tilde{z}_k^j-m)}{\sigma_{H_i}^2}\exp(-\frac{(\tilde{z}_k^j-m)^2}{2\sigma_{H_i}^2})$. In line 5, we use the equality, $\int_{\tau_k^j(n)}^{\tau_k^j(n+1)}(1/\sqrt{2\pi}\sigma_{H_i})\exp(-\frac{(\tilde{z}_k^j-m)^2}{2\sigma_{H_i}^2})d\tilde{z}_k^j = \mathrm{Q}[\frac{\tau_k^j(n)-m}{\sigma_{H_i}}] -$

$\mathrm{Q}[\frac{\tau_k^j(n+1)-m}{\sigma_{H_i}}]$, to obtain line 6.

Therefore, substituting (B.17) in (B.14), we get (6.9) as follows:

$$\begin{aligned}\hat{\mathbf{x}}_{k|k,j}^{H_i} &:= E[\mathbf{x}_k|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i, b_k^j] \\ &= \hat{\mathbf{x}}_{k|k,j-1}^{H_i} + \mathbf{k}^c \frac{(\sigma_{H_i}/\sqrt{2\pi})\left(\exp(-\frac{(\tau_k^j(n)-m)^2}{2\sigma_{H_i}^2}) - \exp(-\frac{(\tau_k^j(n+1)-m)^2}{2\sigma_{H_i}^2})\right)}{\mathrm{Q}[\frac{\tau_k^j(n)-m}{\sigma_{H_i}}] - \mathrm{Q}[\frac{\tau_k^j(n+1)-m}{\sigma_{H_i}}]} \\ &:= \hat{\mathbf{x}}_{k|k,j-1}^{H_i} + \mathbf{k}^c \sigma_{H_i} \alpha_{H_i}(n) \end{aligned} \tag{B.18}$$

where $\alpha_{H_i}(n)$ is as defined in Proposition 2.

Using the concept of iterated expectations, the covariance can be obtained as follows:

$$\begin{aligned}&\mathrm{Cov}[\mathbf{x}_k|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i, b_k^j] = E[(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k,j}^{H_i})(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k,j}^{H_i})^T|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i, b_k^j] \\ &= E[E[(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k,j}^{H_i})(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k,j}^{H_i})^T|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i, \tilde{z}_k^j]|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i, b_k^j] \end{aligned} \tag{B.19}$$

The term $\mathbf{x}_k - \hat{\mathbf{x}}_{k|k,j}^{H_i}$, in the above equation, using (B.11) and (B.18), can be written as:

$$\begin{aligned}\mathbf{x}_k - \hat{\mathbf{x}}_{k|k,j}^{H_i} &= \mathbf{x}_k - \hat{\mathbf{x}}_{k|k,j}^{H_i} + \hat{\mathbf{x}}_{k|k,j}^{H_i^c} - \hat{\mathbf{x}}_{k|k,j}^{H_i^c} \\ &= \mathbf{x}_k - \hat{\mathbf{x}}_{k|k,j}^{H_i^c} + \mathbf{k}^c \left(\tilde{z}_k^j - E[\tilde{z}_k^j|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i, b_k^j]\right) \end{aligned} \tag{B.20}$$

Substituting (B.20) in the inner expectation in (B.19) we obtain:

$$\begin{aligned}&E[(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k,j}^{H_i})(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k,j}^{H_i})^T|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i, \tilde{z}_k^j] \\ &= E[(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k,j}^{H_i^c})(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k,j}^{H_i^c})^T|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i, \tilde{z}_k^j] \\ &\quad + \mathbf{k}^c E[(\tilde{z}_k^j - E[\tilde{z}_k^j|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i, b_k^j])(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k,j}^{H_i^c})^T|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i, \tilde{z}_k^j] \\ &\quad + E[(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k,j}^{H_i^c})(\tilde{z}_k^j - E[\tilde{z}_k^j|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i, b_k^j])^T|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i, \tilde{z}_k^j]\mathbf{k}^{c^T} \\ &\quad + \mathbf{k}^c E[(\tilde{z}_k^j - E[\tilde{z}_k^j|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i, b_k^j]) \\ &\qquad \times (\tilde{z}_k^j - E[\tilde{z}_k^j|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i, b_k^j])^T|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i, \tilde{z}_k^j]\mathbf{k}^{c^T} \\ &= E[(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k,j}^{H_i^c})(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k,j}^{H_i^c})^T|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i, \tilde{z}_k^j] \\ &\quad + \mathbf{k}^c(\tilde{z}_k^j - E[\tilde{z}_k^j|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i, b_k^j])(\tilde{z}_k^j - E[\tilde{z}_k^j|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i, b_k^j])^T\mathbf{k}^{c^T} \\ &= \mathbf{P}_{k|k,j-1}^{H_i} - \mathbf{k}^c\mathbf{h}_k^{j^T}\mathbf{P}_{k|k,j-1}^{H_i} \\ &\quad + \mathbf{k}^c(\tilde{z}_k^j - E[\tilde{z}_k^j|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i, b_k^j])(\tilde{z}_k^j - E[\tilde{z}_k^j|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i, b_k^j])^T\mathbf{k}^{c^T} \end{aligned} \tag{B.21}$$

In the above equation, the cross terms are zero. The first term is the conditional covariance given in (B.12), and the variables in the last term are deterministic functions of the conditioning variables $\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i, \tilde{z}_k^j$. Now considering the outer expectation we obtain:

$$\begin{aligned}\mathrm{Cov}[\mathbf{x}_k|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i, b_k^j] = \mathbf{P}_{k|k,j-1}^{H_i} - \mathbf{k}^c \mathbf{h}_k^{j^T} \mathbf{P}_{k|k,j-1}^{H_i} \\ + \mathbf{k}^c \mathrm{Var}[\tilde{z}_k^j|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i, b_k^j]\mathbf{k}^{c^T} \qquad \text{(B.22)}\end{aligned}$$

We first calculate the variance term in the above equation:

$$\begin{aligned}&\mathrm{Var}[\tilde{z}_k^j|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i, b_k^j] \\ &\quad = E[\tilde{z}_k^{j^2}|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i, b_k^j] - E^2[\tilde{z}_k^j|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i, b_k^j] := c - d \qquad \text{(B.23)}\end{aligned}$$

Therefore,

$$\begin{aligned}c &= \int_{\tau_k^j(n)}^{\tau_k^j(n+1)} \tilde{z}_k^{j^2} \frac{p(\tilde{z}_k^j|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i)}{\Pr\{b_k^j = n|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i\}} d\tilde{z}_k^j \\ &= \int_{\tau_k^j(n)}^{\tau_k^j(n+1)} \tilde{z}_k^{j^2} \frac{\mathcal{N}(\tilde{z}_k^j; m, \sigma_{H_i}^2)}{\Pr\{b_k^j = n|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i\}} d\tilde{z}_k^j \\ &= \frac{\left[\int_{\tau_k^j(n)}^{\tau_k^j(n+1)} \tilde{z}_k^j(\tilde{z}_k^j - m)\exp(-\frac{(\tilde{z}_k^j-m)^2}{2\sigma_{H_i}^2}) d\tilde{z}_k^j + m\int_{\tau_k^j(n)}^{\tau_k^j(n+1)} \tilde{z}_k^j \exp(-\frac{(\tilde{z}_k^j-m)^2}{2\sigma_{H_i}^2}) d\tilde{z}_k^j\right]}{\sqrt{2\pi}\sigma_{H_i}\Pr\{b_k^j = n|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i\}} \\ &:= \frac{[g+h]}{\sqrt{2\pi}\sigma_{H_i}\Pr\{b_k^j = n|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{b}_k^{m\neq i}, \mathbf{z}_{0:k-1}^i\}} \\ &= \frac{[g+h]}{\sqrt{2\pi}\sigma_{H_i}\left(\mathrm{Q}[\frac{\tau_k^j(n)-m}{\sigma_{H_i}}] - \mathrm{Q}[\frac{\tau_k^j(n+1)-m}{\sigma_{H_i}}]\right)} \qquad \text{(B.24)}\end{aligned}$$

$$\begin{aligned}g &= \sigma_{H_i}^2 \int_{\tau_k^j(n)}^{\tau_k^j(n+1)} \tilde{z}_k^j \frac{d}{d\tilde{z}_k^j}(-\exp(-\frac{(\tilde{z}_k^j - m)^2}{2\sigma_{H_i}^2}))\tilde{z}_k^j \\ &= \sigma_{H_i}^2 \left(\left[-\tilde{z}_k^j \exp(-\frac{(\tilde{z}_k^j - m)^2}{2\sigma_{H_i}^2})\right]_{\tau_k^j(n)}^{\tau_k^j(n+1)} + \int_{\tau_k^j(n)}^{\tau_k^j(n+1)} \exp(-\frac{(\tilde{z}_k^j - m)^2}{2\sigma_{H_i}^2}) d\tilde{z}_k^j\right) \\ &= \sigma_{H_i}^2 \left(\tau_k^j(n)\exp(-\frac{(\tau_k^j(n) - m)^2}{2\sigma_{H_i}^2}) - \tau_k^j(n+1)\exp(-\frac{(\tau_k^j(n+1) - m)^2}{2\sigma_{H_i}^2})\right) \\ &\quad + \sqrt{2\pi}\sigma_{H_i}^3 \left(\mathrm{Q}[\frac{\tau_k^j(n) - m}{\sigma_{H_i}}] - \mathrm{Q}[\frac{\tau_k^j(n+1) - m}{\sigma_{H_i}}]\right) \qquad \text{(B.25)}\end{aligned}$$

$$
\begin{aligned}
h &= m\int_{\tau_k^j(n)}^{\tau_k^j(n+1)}(\tilde{z}_k^j - m)\exp(-\frac{(\tilde{z}_k^j-m)^2}{2\sigma_{H_i}^2})d\tilde{z}_k^j + m^2\int_{\tau_k^j(n)}^{\tau_k^j(n+1)}\exp(-\frac{(\tilde{z}_k^j-m)^2}{2\sigma_{H_i}^2})d\tilde{z}_k^j \\
&= m\sigma_{H_i}^2\int_{\tau_k^j(n)}^{\tau_k^j(n+1)}\frac{d}{d\tilde{z}_k^j}(-\exp(-\frac{(\tilde{z}_k^j-m)^2}{2\sigma_{H_i}^2}))d\tilde{z}_k^j \\
&\quad + m^2\sqrt{2\pi}\sigma_{H_i}\left(\mathrm{Q}[\frac{\tau_k^j(n)-m}{\sigma_{H_i}}] - \mathrm{Q}[\frac{\tau_k^j(n+1)-m}{\sigma_{H_i}}]\right) \\
&= m\sigma_{H_i}^2\left(\exp(-\frac{(\tau_k^j(n)-m)^2}{2\sigma_{H_i}^2}) - \exp(-\frac{(\tau_k^j(n+1)-m)^2}{2\sigma_{H_i}^2})\right) \\
&\quad + m^2\sqrt{2\pi}\sigma_{H_i}\left(\mathrm{Q}[\frac{\tau_k^j(n)-m}{\sigma_{H_i}}] - \mathrm{Q}[\frac{\tau_k^j(n+1)-m}{\sigma_{H_i}}]\right) \qquad \text{(B.26)}
\end{aligned}
$$

Therefore, substituting (B.26) and (B.25) in (B.24) we get:

$$
\begin{aligned}
c &= \frac{\sigma_{H_i}}{\sqrt{2\pi}}\frac{\left[\tau_k^j(n)\exp(-\frac{(\tau_k^j(n)-m)^2}{2\sigma_{H_i}^2}) - \tau_k^j(n+1)\exp(-\frac{(\tau_k^j(n+1)-m)^2}{2\sigma_{H_i}^2})\right]}{\mathrm{Q}[\frac{\tau_k^j(n)-m}{\sigma_{H_i}}] - \mathrm{Q}[\frac{\tau_k^j(n+1)-m}{\sigma_{H_i}}]} + \sigma_{H_i}^2 + m^2 \\
&\quad + \frac{m\sigma_{H_i}}{\sqrt{2\pi}}\frac{\left(\exp(-\frac{(\tau_k^j(n)-m)^2}{2\sigma_{H_i}^2}) - \exp(-\frac{(\tau_k^j(n+1)-m)^2}{2\sigma_{H_i}^2})\right)}{\mathrm{Q}[\frac{\tau_k^j(n)-m}{\sigma_{H_i}}] - \mathrm{Q}[\frac{\tau_k^j(n+1)-m}{\sigma_{H_i}}]} \\
&= \frac{\sigma_{H_i}}{\sqrt{2\pi}}\frac{\left[\tau_k^j(n)\exp(-\frac{(\tau_k^j(n)-m)^2}{2\sigma_{H_i}^2}) - \tau_k^j(n+1)\exp(-\frac{(\tau_k^j(n+1)-m)^2}{2\sigma_{H_i}^2})\right]}{\mathrm{Q}[\frac{\tau_k^j(n)-m}{\sigma_{H_i}}] - \mathrm{Q}[\frac{\tau_k^j(n+1)-m}{\sigma_{H_i}}]} \\
&\quad + \sigma_{H_i}^2 + m^2 + m\sigma_{H_i}\alpha_{H_i}(n) \qquad \text{(B.27)}
\end{aligned}
$$

where we obtained the last line from the definition of $\alpha_{H_i}(n)$ in Proposition 2. Also, the quantity d in (B.23) can be expressed in terms of $\alpha_{H_i}(n)$ using (B.17) as follows:

$$
d = (\alpha_{H_i}(n)\sigma_{H_i} + m)^2 \qquad \text{(B.28)}
$$

Therefore, using (B.24) and (B.28), (B.23) can be written as:

$$\begin{aligned}
&\mathrm{Var}[\tilde{z}_k^j|\mathbf{b}_{0:k-1}^{q\neq i},\mathbf{b}_k^{m\neq i},\mathbf{z}_{0:k-1}^i,b_k^j] \\
&\quad=\frac{\sigma_{H_i}}{\sqrt{2\pi}}\frac{\left[\tau_k^j(n)\exp(-\frac{(\tau_k^j(n)-m)^2}{2\sigma_{H_i}^2})-\tau_k^j(n+1)\exp(-\frac{(\tau_k^j(n+1)-m)^2}{2\sigma_{H_i}^2})\right]}{\mathrm{Q}[\frac{\tau_k^j(n)-m}{\sigma_{H_i}}]-\mathrm{Q}[\frac{\tau_k^j(n+1)-m}{\sigma_{H_i}}]} \\
&\qquad+\sigma_{H_i}^2-\alpha_{H_i}^2(n)\sigma_{H_i}^2-\sigma_{H_i}m\alpha_{H_i}(n) \qquad \text{(B.29)}
\end{aligned}$$

Define $\Delta_{H_i}(n)=\frac{(\tau_k^j(n)-m)}{\sigma_{H_i}}$ and $\Delta_{H_i}(n+1)=\frac{(\tau_k^j(n+1)-m)}{\sigma_{H_i}}$. Therefore,

$$\begin{aligned}
&\mathrm{Var}[\tilde{z}_k^j|\mathbf{b}_{0:k-1}^{q\neq i},\mathbf{b}_k^{m\neq i},\mathbf{z}_{0:k-1}^i,b_k^j] \\
&\quad=\frac{\sigma_{H_i}^2}{\sqrt{2\pi}}\frac{\left[\Delta_{H_i}(n)\exp(-\Delta_{H_i}^2(n)/2)-\Delta_{H_i}(n+1)\exp(-\Delta_{H_i}^2(n+1)/2)\right]}{\mathrm{Q}[\frac{\tau_k^j(n)-m}{\sigma_{H_i}}]-\mathrm{Q}[\frac{\tau_k^j(n+1)-m}{\sigma_{H_i}}]} \\
&\qquad+\sigma_{H_i}^2-\alpha_{H_i}^2(n)\sigma_{H_i}^2 \\
&\quad:=\sigma_{H_i}^2(1-\beta_{H_i}(n)) \qquad \text{(B.30)}
\end{aligned}$$

where $\beta_{H_i}(n)$ is as defined in Proposition 2. Therefore, substituting (B.1) in (B.22), the covariance update equation for the HBQKF can be obtained as:

$$\begin{aligned}
\mathrm{Cov}[\mathbf{x}_k|\mathbf{b}_{0:k-1}^{q\neq i},\mathbf{b}_k^{m\neq i},\mathbf{z}_{0:k-1}^i,b_k^j]&=\mathbf{P}_{k|k,j-1}^{H_i}-\mathbf{k}^c\mathbf{h}_k^{jT}\mathbf{P}_{k|k,j-1}^{H_i}+\mathbf{k}^c\sigma_{H_i}^2(1-\beta_{H_i}(n))\mathbf{k}^{cT} \\
&=\mathbf{P}_{k|k,j-1}^{H_i}-\beta_{H_i}(n)\frac{\mathbf{P}_{k|k,j-1}^{H_i}\mathbf{h}_k^j\mathbf{h}_k^{jT}\mathbf{P}_{k|k,j-1}^{H_i}}{\sigma_{H_i}^2} \qquad \text{(B.31)}
\end{aligned}$$

□

B.2 Proof of Proposition 3

Proof. Taking the expectation of the process model in (6.15), conditioned on $\mathbf{b}_{0:k-1}^{q\neq i}$, $\mathbf{z}_{0:k-1}^i$, we obtain

$$\hat{\breve{\mathbf{x}}}_{k|k-1}^{H_i}:=E[\breve{\mathbf{x}}_k|\mathbf{b}_{0:k-1}^{q\neq i},\mathbf{z}_{0:k-1}^i]=\breve{\mathbf{F}}_{k-1}E[\breve{\mathbf{x}}_{k-1}|\mathbf{b}_{0:k-1}^{q\neq i},\mathbf{z}_{0:k-1}^i]=\breve{\mathbf{F}}_{k-1}\hat{\breve{\mathbf{x}}}_{k-1|k-1}^{H_i} \qquad \text{(B.32)}$$

since the $E[\breve{\mathbf{w}}_{k-1}|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{z}_{0:k-1}^{i}] = \mathbf{0}$. The corresponding covariance is given by

$$\begin{aligned}
\breve{\mathbf{P}}_{k|k-1}^{H_i} &:= E[(\breve{\mathbf{x}}_k - \hat{\breve{\mathbf{x}}}_{k|k-1}^{H_i})(\breve{\mathbf{x}}_k - \hat{\breve{\mathbf{x}}}_{k|k-1}^{H_i})^T|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{z}_{0:k-1}^{i}] \\
&= E[(\breve{\mathbf{F}}_{k-1}(\breve{\mathbf{x}}_{k-1} - \hat{\breve{\mathbf{x}}}_{k-1|k-1}^{H_i}) + \breve{\mathbf{G}}_{k-1}\breve{\mathbf{w}}_{k-1}) \\
&\qquad \times (\breve{\mathbf{F}}_{k-1}(\breve{\mathbf{x}}_{k-1} - \hat{\breve{\mathbf{x}}}_{k-1|k-1}^{H_i}) + \breve{\mathbf{G}}_{k-1}\breve{\mathbf{w}}_{k-1})^T|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{z}_{0:k-1}^{i}] \\
&= \breve{\mathbf{F}}_{k-1}E[(\breve{\mathbf{x}}_{k-1} - \hat{\breve{\mathbf{x}}}_{k-1|k-1}^{H_i})(\breve{\mathbf{x}}_{k-1} - \hat{\breve{\mathbf{x}}}_{k-1|k-1}^{H_i})^T|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{z}_{0:k-1}^{i}]\breve{\mathbf{F}}_{k-1}^{T} \\
&\qquad + \breve{\mathbf{G}}_{k-1}E[\breve{\mathbf{w}}_{k-1}\breve{\mathbf{w}}_{k-1}^{T}]\breve{\mathbf{G}}_{k-1}^{T} \\
&= \breve{\mathbf{F}}_{k-1}\breve{\mathbf{P}}_{k-1|k-1}^{H_i}\breve{\mathbf{F}}_{k-1}^{T} + \breve{\mathbf{G}}_{k-1}\breve{\mathbf{Q}}_{k-1}\breve{\mathbf{G}}_{k-1}^{T}
\end{aligned} \tag{B.33}$$

where, line 2 is obtained from (6.15) and (B.32). Line 3 is based on the fact that the noise $\breve{\mathbf{w}}_{k-1}$ is zero-mean and independent of the measurements, $\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{z}_{0:k-1}^{i}$, and the state $\breve{\mathbf{x}}_{k-1}$. Thus, we note that (B.32) and (B.33) are identical to the state / covariance propagation of the standard KF.

Let us now look at the derivation for the state / covariance update equations for robot i processing its own analog measurement z_k^i. Note that since the state is augmented with the noise, v_k^j, corresponding to the quantized bits, $\mathbf{b}_k^{j(1:p-1)}$, being processed, we use the following measurement model for the robots processing their *own* analog measurements:

$$z_k^i = \bar{\mathbf{h}}_k^{i^T}\breve{\mathbf{x}}_k + v_k^i, \quad \bar{\mathbf{h}}_k^{i^T} = [\mathbf{h}_k^{i^T}\ 0] \tag{B.34}$$

$$\begin{aligned}
E[\breve{\mathbf{x}}_k|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{z}_{0:k}^{i}, \mathbf{b}_k^{m\neq i}, \mathbf{b}_k^{j(1:p-1)}] &= \int_{\mathbb{R}^{r+1}} \breve{\mathbf{x}}_k p(\breve{\mathbf{x}}_k|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{z}_{0:k}^{i}, \mathbf{b}_k^{m\neq i}, \mathbf{b}_k^{j(1:p-1)})d\breve{\mathbf{x}}_k \\
= \int_{\mathbb{R}^{r+1}} \breve{\mathbf{x}}_k \frac{p(z_k^i|\breve{\mathbf{x}}_k, \mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{z}_{0:k-1}^{i}, \mathbf{b}_k^{m\neq i}, \mathbf{b}_k^{j(1:p-1)})p(\breve{\mathbf{x}}_k|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{z}_{0:k-1}^{i}, \mathbf{b}_k^{m\neq i}, \mathbf{b}_k^{j(1:p-1)})}{p(z_k^i|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{z}_{0:k-1}^{i}, \mathbf{b}_k^{m\neq i}, \mathbf{b}_k^{j(1:p-1)})} d\breve{\mathbf{x}}_k
\end{aligned} \tag{B.35}$$

In the above equation, $p(z_k^i|\breve{\mathbf{x}}_k, \mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{z}_{0:k-1}^{i}, \mathbf{b}_k^{m\neq i}, \mathbf{b}_k^{j(1:p-1)}) \sim \mathcal{N}\left(\bar{\mathbf{h}}_k^{i^T}\breve{\mathbf{x}}_k, \sigma_k^{i^2}\right)$ since the measurement noise is independent of the conditioning variables.

Also, $p(z_k^i|\mathbf{b}_{0:k-1}^{q\neq i}, \mathbf{z}_{0:k-1}^{i}, \mathbf{b}_k^{m\neq i}, \mathbf{b}_k^{j(1:p-1)}) \sim \mathcal{N}\left(\bar{\mathbf{h}}_k^{i^T}\hat{\breve{\mathbf{x}}}_{k|k,j}^{H_i(p-1)}, \bar{\mathbf{h}}_k^{i^T}\breve{\mathbf{P}}_{k|k,j}^{H_i(p-1)}\bar{\mathbf{h}}_k^{i} + \sigma_k^{i^2}\right)$. Therefore, all pdfs in (B.35) are Gaussian, similar to that of the KF. The derivation from this point onwards, is identical to that of the KF and can be found in [98].

For deriving the state / covariance update equations for the MMSE estimator for robot i, processing the quantized measurement, $b_k^{j(p)}$, $j \neq i$, from robot j, we use the general approach of the batch estimator used in Proposition 2, for the special case of the binary quantizer when the quantization intervals are defined by $\mathcal{R}_k^{j(p)}(n) := [\tau_k^{j(p)}(n), \tau_k^{j(p)}(n+1))$, where $n = \{1, 2\}$, and $\tau_k^{j(p)}(1) = -\infty$, $\tau_k^{j(p)}(2) = 0$, and $\tau_k^{j(p)}(3) = \infty$. Proceeding as in Proposition 2, we define the random variable $\tilde{z}_k^{j(p)} = z_k^j - E[z_k^j | \mathbf{b}_{0:k-1}, \mathbf{b}_k^m, \mathbf{b}_k^{j(1:p-1)}]$ and calculate and obtain the following joint pdf

$$p(\breve{\mathbf{x}}_k, \tilde{z}_k^{j(p)} | \mathbf{b}_{0:k-1}^{q \neq i}, \mathbf{z}_{0:k-1}^i, \mathbf{b}_k^{m \neq i}, \mathbf{b}_k^{j(1:p-1)}) \\ \sim \mathcal{N}\left(\begin{bmatrix} \hat{\mathbf{x}}_{k|k,j}^{H_i(p-1)} \\ \mathbf{h}_k^{j^T} (\hat{\mathbf{x}}_{k|k,j}^{H_i(p-1)} - \hat{\mathbf{x}}_{k|k,j}^{Q(p-1)}) \end{bmatrix}, \begin{bmatrix} \mathbf{P}_{k|k,j}^{H_i(p-1)} & \mathbf{P}_{k|k,j}^{H_i(p-1)} \mathbf{h}_k^j \\ \mathbf{h}_k^{j^T} \mathbf{P}_{k|k,j}^{H_i(p-1)} & \mathbf{h}_k^{j^T} \mathbf{P}_{k|k,j}^{H_i(p-1)} \mathbf{h}_k^j + \sigma_k^{j^2} \end{bmatrix} \right) \tag{B.36}$$

and then obtain the required conditional pdf to finally arrive at the state / covariance update rules (6.18), and (6.19). □