Department of Informatics
Technical University of Munich

# Localization of Cyber-Physical Systems: Privacy, Security and Efficiency

**Amr Abdelhafez Mohamed Alanwar Abdelhafez**

Vollständiger Abdruck der von der Fakultät für Informatik der Technischen Universität München zur Erlangung des akademischen Grades eines

**Doktor der Naturwissenschaften (Dr. rer. nat.)**

genehmigten Dissertation.

**Vorsitzender:**
Prof. Dr. Bernd Brügge

**Prüfende der Dissertation:**
1. Prof. Dr.-Ing. Matthias Althoff
2. Prof. João P. Hespanha,
University of California, Santa Barbara

Die Dissertation wurde am 23.12.2019 bei der Technischen Universität München eingereicht und durch die Fakultät für Informatik am 02.03.2020 angenommen.

# Foreword

This Ph.D. is a life-changing experience for me, and it would not have been possible to do without the support of many people. I would like to thank first God for giving me the strength and opportunity to undertake this Ph.D. study.

I am very fortunate to have Professor Matthias Althoff as my advisor and would like to express my sincere gratitude to him. He has been a wonderful mentor for me, and I can not thank him enough for his patience and immense knowledge. He fosters a personal relationship with his students that is based on mutual respect. I am grateful to him for providing the environment necessary to grow, develop, and pursue my passion. I appreciate his advice, support, and guidance. I am indebted to Professor Althoff very much as he taught me how to define a research problem, find a solution to it, and finally write a scientific paper and publish the results.

Besides my advisor, I would like especially to thank Professor João P. Hespanha. This thesis could be done without his significant support. This crucial remarks that shaped my research were very helpful. I greatly appreciate his encouragement, supervisory role, and valuable input. Professor Hespanha is someone that you instantly love and never forget forever. He is one of my best role models for a scientist, mentor, and teacher.

My thanks also go out to the support I received from Professor Paulo Tabuada, Professor Mani Srivastava, Doctor Hazem Said, Doctor Supriyo Chakraborty, Doctor Fei Miao, Doctor Yuan Tian, Professor Ankur Mehta and Professor George J. Pappas. Also, my collaborators Doctor Bernhard Etzlinger, Henrique Ferraz, Victor Gaßmann, Jagat Rath, M. Tarek Ibn Ziad, Paul D Martin, Kevin Hsieh, and Justin Pearson.

A special thanks goes to a large group of friends and lab-mates who supported me including but not limited to Osama Keraitam, Doctor Ahmed Ismail, Doctor Sherif Hammad, Ahmed Maher, Mekdad Keraitam, Abdallah Keraitam, Niklas Kochdumper, Markus Koschi, Amr Alaa, Mohamed Soliman, Mohamed Nour, Kream Gamal, Stefanie Manzinger, Christian Pek, Edmond Irani, Anna-Katharina Rettinger, Felix Gruber, Xiao Wang, Sebastian Maierhofer, and Egon Ye.

Finally, I would like to express my deepest gratitude to my father, mother, beloved wife, sister, daughter, father-in-law, mother-in-law, and brothers in law. Thanks for all of the sacrifices that have made on my behalf. Their prayers were what sustained and supported me thus far. This dissertation would not have been possible without their warm love, continued patience, and endless support. Words can not express how grateful I am to all of them. They supported me during critical times of my Ph.D. My life is so blessed with some of the most amazing people. Thank you for being part of my journey.

# Abstract

The tight coupling of information technology with physical sensing and actuation has opened the door for privacy leakage and security vulnerability in the localization algorithms of Cyber-Physical Systems (CPS). Also, localization algorithms make use of resource-constrained sensor nodes. Thus, we need a paradigm shift in the algorithmic design to localize CPS privately, securely, and efficiently. Interdisciplinary approaches are required that bring a diverse set of disciplines together to bear on the design process of the localization. Such approaches should take care of trustworthy, privacy-aware, and efficiency aspects of the localization process. The objective of this dissertation is tackling these emerging concerns and developing algorithms to address them effectively.

The contribution of this thesis is multi-fold. We start by tackling the privacy and security concerns while satisfying the real-time requirements in the localization process. More specifically, we present PrOLoc (Private Observers for secure Localization), a set of novel protocols and algorithms for estimating the location of a given target while guaranteeing the privacy of locations and measurements of the observers. Moreover, and unlike previously proposed perturbation based techniques, PrOLoc is also resilient to malicious active false data injection attacks on sensor and link levels. Then, we move from the centralized estimation algorithms to the distributed ones. More specifically, we propose D-SLATS (Distributed Simultaneous Localization and Time Synchronization), a framework comprised of three different and independent algorithms to jointly solve time synchronization and localization problems in a distributed fashion. The performance of a distributed network state estimation problem depends strongly on collaborative signal processing, which often involves excessive communication and computation overheads on a resource-constrained sensor node. Therefore, we next propose an event-triggered diffusion Kalman filter, which only collects measurements and exchanges relative messages between nodes based on a local signal indicative of the estimation error. This leads to an energy-aware state estimation algorithm, which we apply to the distributed simultaneous localization and time synchronization problem. Next, we attach the utmost importance to the security of the distributed estimation algorithms. A secure distributed state estimation algorithm that works in the presence of modeling and measurement noise between a network of nodes with pairwise measurements is presented. Reachability analysis is utilized to establish a security layer providing secure estimate shares for the distributed diffusion Kalman filter. Finally, we touch upon the problem of Human-Computer Interface (HCI) as an application to the accurate localization. We propose SeleCon, stands for device Selection and Control, a pointing approach to interacting with devices, as pointing is arguably a natural way for device selection.

# Zusammenfassung

Die enge Kopplung von Informationstechnologie mit Sensoren und Aktoren birgt ein erhöhtes Risiko für Datenschutzverletzungen und Sicherheitslücken in den Lokalisierungsalgorithmen von Cyber-Physical Systems (CPS). Zudem verwenden Lokalisierungsalgorithmen ressourcenbeschränkte Sensorknoten. Daher wird ein Paradigmenwechsel im algorithmischen Entwurf benötigt, um die Lokalisierung von CPS vertraulich, sicher und effizient zu gestalten. Hierbei sind interdisziplinäre Ansätze gefordert, um den Entwurfsprozess zu bewerkstelligen und dabei auf Vertrauenswürdigkeit, Datenschutz und Effizienz der Lokalisierungsalgorithmen zu achten. Das Ziel dieser Dissertation ist es, diese aufkommenden Herausforderungen mit Hilfe neuartiger Algorithmen zu bewältigen. Die Beiträge dieser Arbeit umfassen verschiedene Aspekte. Zu Beginn präsentieren wir PrOLoc (engl.: Private Observers for secure Localization)—eine Reihe neuartiger Protokolle und Algorithmen zur Positionsschätzung eines vorgegebenen Ziels, die gleichzeitig das Verbergen der Positionen und Messungen der Beobachter gewährleisten. Hierbei werden neben Datenschutz- und Sicherheitsbedenken gleichzeitig Echtzeitanforderungen im Lokalisierungsprozess berücksichtigt. Im Gegensatz zu bisherigen störungsbasierten Techniken erweist sich PrOLoc als robust gegen bösartige Datenmanipulationen auf Sensor- und Verbindungsebene. Neben zentralisierten Schätzungsalgorithmen werden im Rahmen dieser Arbeit verteilte Algorithmen untersucht. Hierfür präsentieren wir das Framework D-SLATS (Distributed Simultaneous Localization and Time Synchronization), welches aus drei unterschiedlichen und unabhängigen Algorithmen besteht, um Zeitsynchronisationsprobleme zusammen mit Lokalisierungsproblemen verteilt zu lösen. Die Performance einer verteilten Zustandsschätzung in einem Netzwerk hängt stark von der kollaborativen Signalverarbeitung ab, die oft mit einem erhöhten Kommunikations- und Rechenaufwand auf einem ressourcenbeschränkten Sensorknoten einhergeht. Daher wird als nächstes ein ereignisgesteuerter Diffusion-Kalman-Filter vorgeschlagen, der nur Messungen sammelt und Nachrichten zwischen Knoten austauscht, die auf einem lokalen Signal basieren, das auf den Schätzfehler hinweist. Dies führt zu einem energiebewussten Zustandsschätzungsalgorithmus, der zur verteilten und simultanen Lokalisierung und Zeitsynchronisation verwendet wird. Im nächsten Schritt beschäftigen wir uns mit der Sicherheit der verteilten Schätzungsalgorithmen. Dazu präsentieren wir einen sicheren Algorithmus zur verteilten Zustandsschätzung, der Modellierungs- und Messrauschen in einem Netzwerk von Knoten mit paarweisen Messungen berücksichtigt. Für den Security-Layer, der sichere Schätzanteile für den verteilten Diffusion-Kalman-Filter liefert, wird Erreichbarkeitsanalyse verwendet. Als Anwendungsfall für die genaue Lokalisierung gehen wir schließlich auf die Mensch-Computer-Schnittstelle (engl.: HCI) ein. Dazu stellen wir ein Framework, genannt SeleCon (engl.: device Se-

lection and Control), zur Interaktion mit Geräten vor. Dieses Framework verwendet das Zeigen mit der Hand auf ein Gerät als eine natürliche Methode zur Geräteauswahl.

# Contents

# List of Figures

# List of Tables

# List of Symbols

| | |
|---|---|
| $\mathbb{R}$ | Set of real numbers |
| $\mathbb{Z}$ | Set of integers |
| $\mathbb{T}$ | Localization target |
| $\mathbb{S}$ | Localization sensor (anchor) |
| $\mathbb{A}$ | Aggregator node |
| $\mathbb{Q}$ | Query node |
| $\|x\|_2$ | Second norm of vector $x$ |
| $\lambda_{\text{max}}\{A\}$ | Maximum eigenvalue of a symmetric matrix $A$ |
| pk | Public key |
| sk | Private key |
| $[\![a]\!]_{\text{pk}}$ | Encrypted value of $a$ using public key using Paillier crypto system |
| $\oplus$ | Homomorphic addition |
| $\ominus$ | Homomorphic subtraction |
| $\otimes$ | Homomorphic self-blinding |
| $[\![a]\!]_{\text{pk}}^{\text{FHE}}$ | Fully homomorphic encryption of $a$ using public key |
| $\mathcal{P}(A, b)$ | polyhedron with matrix $A$ and vector $b$ |
| $\mathcal{H}$ | Halfspace |
| $A(i)$ | ith row of matrix $A$ |
| $\mathbb{EN}(z)$ | Encoding of a float number $z$ |
| $n_i$ | Process noise at time step $i$ |
| $v_{k,i}$ | Measurement noise of node $k$ at time step $i$ |
| $f_i$ | State update function |
| $h_{k,i}$ | Measurement function |
| $\mathcal{N}_k$ | Neighborhood of node $k$ |
| $\boldsymbol{p}$ | Three dimensional position vector |
| $o$ | Clock offset |
| $b$ | Clock bias |
| $c$ | Speed of light in a vacuum |
| $Q_i$ | Process covariance matrix at time step $i$ |
| $R_{j,i}$ | Measurement noise covariance matrix of node $j$ at time step $i$ |
| $d_{kj,i}$ | Counter difference between node $k$ and node $j$ |
| $r_{kj,i}$ | Single-sided two-way distance measurement between node $k$ and node $j$ |
| $\Gamma_{kj,i}$ | Double-sided two-way distance measurement between node $k$ and node $j$ |
| $T_{RSP}$ | Response time between the two pairs of timestamps |
| $T_{RND}$ | Round-trip time |
| $\boxtimes$ | Kronecker product |

| | |
|---|---|
| $\mathcal{Z} = \langle c, G \rangle$ | Zonotope with $c$ as center and $G$ as generator |
| $\boxplus$ | Minkowski sum |
| $a_{kj,i}$ | Attack vector on the measurements between node $k$ and node $j$ at time $i$. |
| $\bar{n}_i$ | Smart device |
| $\bar{n}_u$ | Smartwatch |
| $p_u(t)$ | Position of a user wearing a smartwatch $n_u$ at time $t$ |
| $r_i(t)$ | Ranging measurements between the smartwatch $n_u$ and smart device $n_i$ |
| $t_s$ | Starting time of the pointing gesture |
| $t_f$ | Finishing time of the pointing gesture |
| $N(\mu_r, \sigma_r^2)$ | Gaussian distribution with a mean $\mu_r$ and a standard deviation $\sigma_r$ |
| $\tilde{\theta}$ | Pointing angular error |
| $\theta_{min}$ | Angle formed by the true device $n_{i^*}$, the user $n_u$, and the closest device $n_j$ |

# List of Abbreviations

| | |
|---|---|
| CPS | Cyber-Physical Systems |
| HCI | Human-Computer Interface |
| IoT | Internet of Things |
| UWB | Ultra-wideband |
| SCADA | Supervisory Control and Data Acquisition |
| D-SLATS | Distributed simultaneous localization and time synchronization |
| PrOLoc | Private Observers for secure Localization |
| SeleCon | Device Selection and Control |
| SFE | Secure Function Evaluation |
| FHE | Fully Homomorphic Encryption |
| PHE | Partially Homomorphic Encryption |
| GC | Garbled Circuits |
| SMC | Secure Multi-Party Computation |
| Poly-LSQ | Polyhedra-based Least-Squares localization |
| Poly-AP | Polyhedra-based Alternating Projection localization |
| SMC-Poly-AP | Secure Multiparty Computation Polyhedra-based Alternating Projection localization |
| RLWE | Ring Learning With Error |
| GDOP | Geometric Dilution Of Precision |
| DCRA | Decisional Composite Residuosity |
| RSSI | Received Signal Strength Indication |
| TOA | Time of Arrival |
| TDOA | Time Difference of Arrival |
| AOA | Angle of Arrival |
| MLE | Maximum Likelihood Estimator |
| GPS | Global Positioning System |
| TX | Transmission |
| RX | Reception |
| DKAL | Distributed Kalman filter for Localization |
| DKALarge | Distributed Kalman filter for Localization of Large scale systems |
| DICI-OR | Distributed Iterate Collapse Inversion Overrelaxation |
| DW | DecaWave |
| ROS | Robot Operating System |
| EKF | Extended Kalman Filter |
| MMSE | Minimum Mean-Squared Error |
| OSTP | Office of Science and Technology Policy |
| IMU | Inertial Measurement Unit |
| CSI | Channel State Information |
| NIC | Network Interface Card |
| DOF | Degree of freedom |
| SNR | Signal to Noise Ratio |
| SVM | Support Vector Machine |

# 1 Introduction

Position estimation is an essential requirement for different applications, such as military [2], indoor and outdoor localization [3], security surveillance, and wildlife habitat monitoring; it is often a fundamental requirement for Cyber-Physical Systems (CPS). Advances in localization techniques have enabled a multitude of services, including navigation, targeted advertisements, and location-aware applications. With the growing prevalence of powerful mobile computing devices, methods for localization both in outdoor and indoor environments are becoming more wide-spread and more varied. Many of these localization techniques depend on analyzing various observations of a distinct phenomenon to infer the location of some object or event whose location is unknown and possibly changing over time. In such scenarios, measured signals are often captured by several sensors or observers and then typically communicated to a centralized computer or an aggregator for post-processing and, eventually, location estimation. For example, cellular signals from mobile devices can be measured by base stations to infer the location of the device based on geometric multilateration, and wireless sensor nodes can use microphones along with beamforming algorithms to localize the source of specific acoustic signatures. The final location estimate is then sent to the interested party who initiated the inquiry.

Location-based services have to deal with challenges to ensure privacy, security, and efficiency. We are going to discuss these challenges in the next few lines in more detail. First, we discuss the information leakage within the localization process. Next, we mention security challenges, and then we discuss the need for distributed and efficient localization algorithms.

**Private Localization**: Traditionally, the infrastructure which is used for localization—microphones, radios, light sensors, etc.—are treated as non-private entities whose locations and measurements are either known or easily inferred. Knowing both the locations and the measurements of these devices is required for the localization of other objects. While in many cases the privacy of the locations and measurements of the observers or the infrastructure is of little concern—for example, WiFi routers in a public shopping mall—in other cases publicizing this data can have serious security implications, for example, the infrastructure of military base, the position of observing soldiers, and observing participants for a particular events. In fact, observer data leakage can occur easily through one of the following two ways:

- Malicious Aggregators: The locations of observers, as well as the distance measurements, are typically stored at a remote server to perform a joint position estimation. In multilateration and multiangulation systems, for example, infrastructure positions, ranges, and angles are often communicated to a single device, which

then performs an optimization routine in order to arrive at an accurate position estimate of some target. Upon receiving the sensitive data, a malicious aggregator could relay this information to interested third parties with more nefarious intents. In many scenarios, this information can be incredibly sensitive for example the military signal towers used to locate soldiers in the field, distributed and crowd-sourced detection of gunshots, or other illicit activities for which the observers should remain anonymous in order to be protected.

- Aggregator Information Leaks: The aggregation and centralization for the information of the observers creates a point of weakness in traditional systems, where even a trusted aggregator may inadvertently leak sensitive data—e.g., the infamous examples of hackers gaining root access to the WordPress[1] server and the data breach of customers' information on JCPenny[2] servers.

**Secure Localization**: One of the key components in the localization process is physical sensing to get distance measurements. However, research evidence shows that the introduced tight coupling of information technology with physical sensing and actuation leads to more vulnerability and security weaknesses. For instance, the Maroochy Water Breach [4] made it possible to attack the underlying infrastructure at Maroochy Water Services in Queensland. Also, one popular attack is the Stuxnet attack on Supervisory Control and Data Acquisition (SCADA) systems, which are used in industrial process control [5, 6]; other security issues on SCADA networks are shown in [7]. Attacks on analog sensors that have increasingly become an indispensable part of many modern systems are shown in [8]. Moreover, the vulnerability of drones is a perilous threat if attackers can take control of them [9, 10]. Also, consider the case of generating a fake GPS signal that appears identical to those sent out by the real GPS [11] to take over some unmanned aircraft [12]. While considerable research has explored CPS security using cryptographical techniques, it is not sufficient to ensure CPS security. Such cyber-security techniques cannot protect against the compromised physical environment around a sensor node, which may inject a corrupted signal. Thus, researchers came up with techniques that address the problem of secure state estimation under attacks on sensors, actuators, and communication network. Secure state estimation allows the estimation of the state of the CPS from corrupted measurements. The state, in our case, is the location of the target in CPS.

**Efficient and Distributed Localization:** Centralized algorithms of localization might attain perfect performance. However, they are neither robust nor scalable to complex, large-scale dynamical systems with their measurements distributed over a large geographical region. In order to perform localization using a centralized algorithm, all nodes should send their measurements to a central entity, which uses conventional techniques to obtain the location of different nodes. The information is then sent back to every node. This strategy requires a large communication overhead and has a potentially critical fail-

[1] http://it.slashdot.org/story/11/04/13/1925244/wordpress-hacked-attackers-get-root-access

[2] http://www.nbcnews.com/id/22718442/ns/technology_and-science_security/t/credit-card-data-breach-could-affect/

ure point at the fusion center. Besides ensuring the accuracy of estimating the localization, one has to consider power constraints [13], limitations in terms of bandwidth [14], and limitations in computation [15] and communication [16]. One of the most popular estimation algorithms for sensor networks is the distributed Kalman filtering algorithm. Among distributed algorithms, diffusion algorithms have favorable properties with respect to performance and robustness to node and link failures. The performance of the distributed diffusion Kalman filter [17] depends on frequent measurements and message exchanges between nodes. On the other hand, the capabilities of individual nodes are minimal, and they are battery-powered. So, decreasing the communication overhead and the number of measurements is of great importance.

Given the mentioned challenges in the localization process, a paradigm shift in the algorithmic design to enhance the localization process is essential. Interdisciplinary approaches that bring a diverse set of disciplines to bear on the design process of localization algorithms are required. The objective of this dissertation is tackling these emerging concerns and developing algorithms to address them effectively. This thesis is a summary of the work in theses publications [18–32].

The key contributions of this thesis can be summarized as follows:

- We present PrOLoc (Private Observers for secure Localization) in Chapter 2, which consists of three algorithms for performing localization of a mobile target based on encrypted measurements from private observers. PrOLoc leverages the Paillier additive homomorphic cryptosystem to efficiently estimate target positions in real-time without revealing the locations or measurements of any given observer. We analyze the performance of the proposed localization algorithms in terms of computational efficiency and privacy preservation. We evaluate the PrOLoc using an end-to-end system of range measurement hardware in a laboratory in terms of accuracy and efficiency. We provide strong theoretical guarantees regarding observer privacy and resilience for each of the demonstrated algorithms. Our experiments on real hardware demonstrate that PrOLoc yields accurate location estimate at least $500x$ faster than state-of-art secure function evaluation techniques.

- Distributed Simultaneous Localization and Time Synchronization (D-SLATS) algorithms are proposed in Chapter 3. We propose distributed approaches to achieve network time synchronization and accurate localization estimates using distributed filters and optimization techniques. While distributed estimators benefit from improved scalability, the joint estimator presented lends itself more naturally to sensor networks requiring high fidelity, high-frequency synchronization, and localization, e.g., autonomous robotics and indoor pedestrian tracking. Therefore, we aim to target both advantages by providing accurate, distributed approaches. This chapter leverages ultra-wideband communication in order to make precise timing measurements. We take into consideration the scalability factor and show the advantages and disadvantages of using our algorithms. We demonstrate the benefit of D-SLATS using custom ultra-wideband wireless devices and a quadrotor. We achieve micro seconds synchronization error and half meter localization error.

- Chapter 4 introduces the event-triggered distributed diffusion Kalman filter to reduce the communication, computation, and measurements overhead. We show that our event-triggered estimator is unbiased and derive the relationship between the triggering signal and the expected error covariance. Then, we apply the proposed algorithm in localizing and time-synchronizing distributed nodes in an ad-hoc network. We evaluate the proposed strategy on a real testbed using custom ultra-wideband wireless devices and a quadrotor, representing a network of both static and mobile nodes. Our experimental results show that we are able to save $86\%$ of the communication overhead, while only introducing $16\%$ performance degradation.

- We propose in Chapter 5 an approach for distributed linear secure state estimation in the presence of measurement noise and modeling errors. By combining the diffusion Kalman filter [33] with reachability analysis [34], we provide a new algorithm for distributed secure state estimation between a network of nodes. We apply the proposed algorithm on a localization example of a rotating target where the measurements are under attack.

- We introduce SeleCon (Device Selection and Control) in Chapter 6 which provides a practical and scalable method of Internet of Things (IoT) device selection and control using pointing and hand gestures as an application to accurate localization. Given the popularity and the growing number of IoT devices, selecting one out of many devices becomes a hurdle in a typical smart home environment. Thus, we propose SeleCon with a hardware prototype of smartwatch equipped with a ultra-wideband radio and inertial sensors. We develop machine-learning models for device selection and hand gesture recognition from ultra-wideband ranging and inertial sensors data. To interact with a device in our system, people can point to the device to select it then draw a hand gesture in the air to specify a control action. The results demonstrate that SeleCon can achieve $84.5\%$ accuracy for device selection and $97\%$ accuracy for hand gesture recognition.

# 2 Resilient Localization with Private Observers Using Partially Homomorphic Encryption

It is well established that location information, when associated with a user, can reveal sensitive information about the user and violate privacy. For example, we can disclose the health conditions, religious inclination, home address and workplaces of people using their locations. This motivates the design of a large number of techniques that protect the privacy of the target user while simultaneously enabling the sharing of location information [35–38]. In comparison, relatively insignificant attention has been given towards protecting the location privacy of the observers that enable the process of localization. We consider the problem of secure localization of a target while maintaining the privacy of the participating observers.

This chapter presents PrOLoc which is a set of novel algorithms for private estimating the location of a given target based on measurements from hidden observers. Algorithms in PrOLoc re-formulate the localization process to leverage partially homomorphic encryption techniques in order to calculate a location estimate based on encrypted range measurements from observers with encrypted locations. The resulting encrypted estimate is then sent to the inquiring party that can only decrypt it without inferring the private locations or ranges of any given observer. PrOLoc provides strong privacy and security guarantees. Our technique, by design, produces negligible degradation in localization accuracy. We can further augment our private technique with residue-checking-based schemes to incorporate resilience against active attacks. We implemented PrOLoc on a real-time testbed consisting of four observers to demonstrate its feasibility on energy-constrained devices.

This chapter is an extended version of our publications in [18, 19]. The rest of this chapter is organized as follows. The private and secure localization challenges are presented in Section 2.1. We review the traditional non private least-squares localization in Section 2.2. The problem setup is presented in Section 2.3. We go through the related work in Section 2.4. Then, we provide in Section 2.5 the mathematical preliminaries and basic cryptographic blocks. The main contributions of this chapter are Sections 2.6 and 2.7 where we discuss novel localization algorithms named (i) polyhedra-based localization: least-squares approach and (ii) polyhedra-based localization: an alternating projection approach. The details of these algorithms are first given for non-privacy-aware implementations followed by efficient privacy-aware implementations of the proposed algorithms along with the theoretical guarantees for each. The resilient privacy-aware

localization is introduced in Section 2.8. The proposed algorithms are evaluated using a real-time testbed in Section 2.10. Finally, we conclude the chapter by disclosing the limitations in Section 2.11.

## 2.1 Private and Secure Localization Challenges

To address privacy requirements and prevent leakage of sensitive observer information, researchers have begun to re-imagine localization techniques in a more privacy-aware and privacy-preserving manner. We identify below several challenges that need to be addressed for any realizable solution:

- Accepted localization accuracy: The accuracy of the localization process is key for its use in various applications (e.g., navigation, tracking gunshots, monitoring objects). Thus, any proposed solution should aim to re-formulate the existing localization algorithms in novel ways to preserve privacy, without sacrificing efficiency or localization accuracy.

- Energy-constrained observers: The observers providing the measurements are often energy-constrained miniature devices. Thus, any solution providing strong privacy guarantees should also be sufficiently efficient in terms of both computation and communication costs.

- Adversaries eavesdropping: While the location is in general sensitive information, there should be a protection against passive adversaries eavesdropping on the protocol messages and inferring sensitive observer information.

- Resilience under active attacks: The privacy guarantees provided by any solution should be resilient to *collusion attacks* and *false data injection attacks* from active adversarial observers. As part of a collusion attack, a group of observers can collude between themselves and/or with the aggregator node to reveal the sensitive information of other participating observers. Similarly, malicious observers can attempt to degrade the performance of the localization process itself by reporting false noisy measurements to the aggregator node.

One localization approach is to retain the structure of a given localization function and model its computation as a "secure function evaluation" (SFE) problem thereby protecting the privacy of the observers. This would also allow exploiting the repertoire of efficient multi-round message-passing protocols that have been designed over the years to perform SFE. However, SFE protocols require all observers to be online during the execution of a long protocol. This is a major drawback for battery-operated, resource-constrained sensor nodes that act as observers during the localization process. Even in cases where power is not a limitation (e.g., military espionage with human agents), it might not be feasible for an observer (human agent) to remain visible or online for the required duration for completing the protocol without compromising their location and hence their safety. Thus, instead of directly using SFE, we explore ways to reformulate

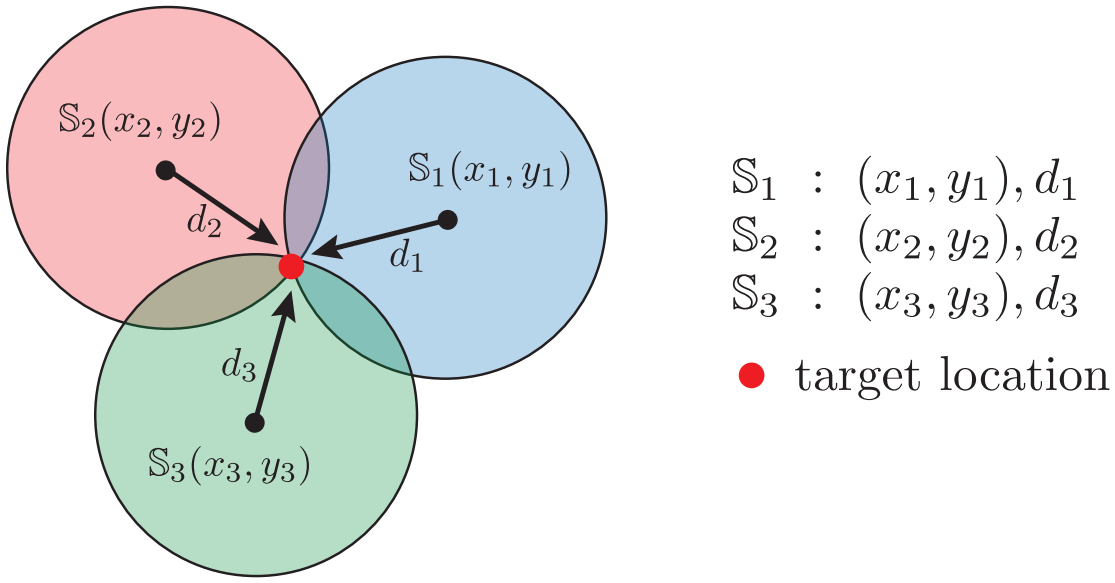



**Figure 2.1:** Trilateration using ranges from three observers ($\mathbb{S}_i$)

the localization process itself such that we can continue using the tools designed for SFE without requiring observers to be available after providing their reports.

Yet another method of preserving observer privacy, and one that we extensively use in our work, is to homomorphically encrypt data prior to transmission and to perform localization using encrypted data at the server. One way to encrypt data is to use *Fully* Homomorphic Encryption (FHE) that allows both addition and multiplication operations to be performed on the encrypted data itself. However, FHE is computationally very expensive and impractical for most cases [39]. An alternative strategy is to use *partially* homomorphic encryption (PHE) as the Pallier *additive* homomorphic cryptosystem [40]. Unlike the fully homomorphic cryptosystems, the partial ones have matured in recent years to a point where they can be carried out with much greater efficiency. Unfortunately, because standard localization algorithms have not been designed with privacy in mind, they cannot benefit directly from the performance advantage of PHE over FHE without a dramatic modification in the localization algorithm.

## 2.2 Traditional Least Squares Localization

The objective of any localization algorithm is to calculate the position of the target, denoted by $z_{\mathbb{T}} = (x_{\mathbb{T}}, y_{\mathbb{T}}) \in \mathbb{R}^2$, using all information provided by the $m$ sensors $\mathbb{S}_i$ where $i$ is the sensor index as shown in Figure 2.1 with $m = 3$. One common method for estimating the target position is by using range measurements $d_i$ between the sensors and the target to constrain the target's position on a 2D plane. Every sensor provides its known position $(x_i, y_i)$ and measured distance $d_i$ to the target location. We consider the geometric multilateration method, which calculates the target location based on the geometry imposed by sensor-target range estimates. The ranges measurements of sensor

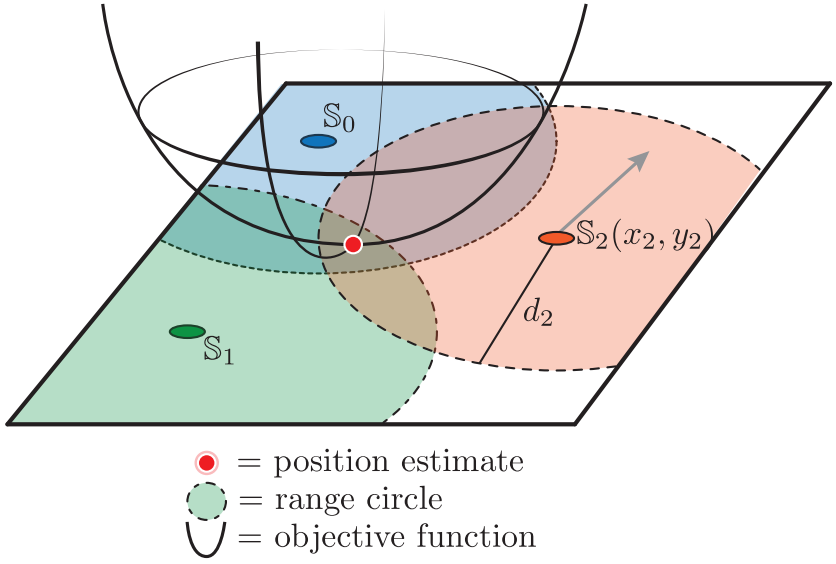



**Figure 2.2:** Objective function based on measurements $d_i$ from observers $\mathbb{S}_i$.

$\mathbb{S}_i$ are related to the positions by the relation $d_i^2 = (x_i - x_\mathbb{T})^2 + (y_i - y_\mathbb{T})^2$. Stating and rearranging the relations for all the sensors results in the traditional multilateration method, which can be casted as the solution of the following least-squares optimization problem:

$$\hat{z}_\mathbb{T} = (\hat{x}_\mathbb{T}, \hat{y}_\mathbb{T}) = \arg\min_{\hat{z}_\mathbb{T} \in \mathbb{R}^2} \|A_\mathbb{S} \hat{z}_\mathbb{T} - b_\mathbb{S}\|_2^2, \tag{2.1}$$

$$A_\mathbb{S} := \begin{bmatrix} 2(x_2 - x_1) & 2(y_2 - y_1) \\ \vdots & \vdots \\ 2(x_m - x_1) & 2(y_m - y_1) \end{bmatrix},$$

$$b_\mathbb{S} := \begin{bmatrix} x_2^2 + y_2^2 - d_2^2 - (x_1^2 + y_1^2 - d_1^2) \\ \vdots \\ x_m^2 + y_m^2 - d_m^2 - (x_1^2 + y_1^2 - d_1^2) \end{bmatrix}.$$

where $\hat{z}_\mathbb{T}$ denotes the estimated value of the target position $z_\mathbb{T}$. Note that we use the subscript $\mathbb{S}$ in $A_\mathbb{S}$ and $b_\mathbb{S}$ to emphasize the fact that both the matrix $A_\mathbb{S}$ as well as the vector $b_\mathbb{S}$ depend on the (potentially sensitive) information of sensors $\mathbb{S}$. This will play a vital role in designing our algorithms as we will show in the next section. To better understand the optimization problem in (2.1), we plot in Figure 2.2 the value of the objective function $\|A_\mathbb{S} \hat{z}_\mathbb{T} - b_\mathbb{S}\|_2^2$ for different values of $\hat{z}_\mathbb{T}$. As shown in Figure 2.2, the objective function has a minimum at the target location. The minimum of this optimization function (and hence the target location) can be obtained as:

$$(\hat{x}_\mathbb{T}, \hat{y}_\mathbb{T}) = (A_\mathbb{S}^T A_\mathbb{S})^{-1} A_\mathbb{S}^T b_\mathbb{S} \tag{2.2}$$

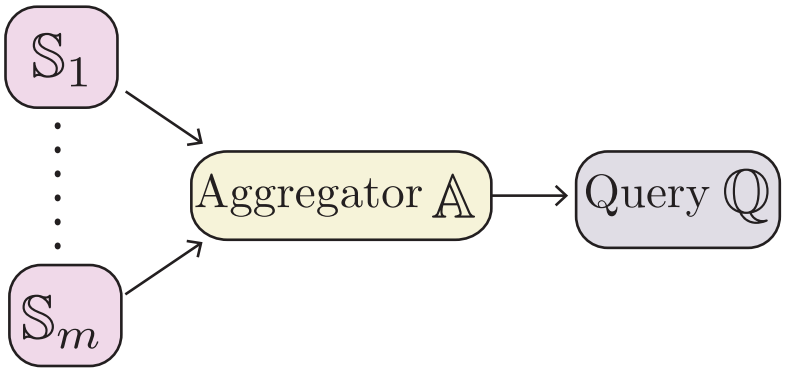



**Figure 2.3:** Communication flow for aggregator based localization.

To avoid matrix inversion, one alternative in solving (2.1) is to start from any initial guess and then iteratively enhance this guess by following the slope of the quadratic shape until we reach the minimum (shown in Figure 2.2). This technique is known as the gradient descent technique. In this iterative algorithm, we start by choosing any random initial estimate, e.g. $\hat{z}_{\mathbb{T}}^{(0)} = \begin{bmatrix} 0 & 0 \end{bmatrix}^T$, and then iteratively update the estimate as:

$$\hat{z}_{\mathbb{T}}^{(i+1)} = \hat{z}_{\mathbb{T}}^{(i)} + \eta A_{\mathbb{S}}^T \left( b_{\mathbb{S}} - A_{\mathbb{S}} \hat{z}_{\mathbb{T}}^{(i)} \right) \tag{2.3}$$

where $\eta$ is a design parameter known as the gradient step size and $i$ is the iteration number. Calculating the target location using (2.3) requires both multiplication and addition of sensitive information. While this cannot be implemented directly using only additive homomorphic encryption like Paillier algorithm, it can be implemented using other SFE techniques like FHE [41], Garbled Circuits (GC) [42], or hybrid SFE (partial homomorphic encryption plus Garbled circuits) [43]. However, we show in our experiments in Section 2.10 that FHE is still far from being used in practice due to its computation cost, and GC and hybrid SFE require large number of communication rounds between observers and aggregator which violate our utility measure (observer communication cost). Rethinking the optimization problem (2.1) from a privacy point of view motives us to propose PrOLoc.

## 2.3 Problem Setup

One straightforward approach to privatizing existing localization algorithms would be to encrypt the sensitive information of each observer using the public key of the inquiring party (the *query* node) which the party interested in the location of a given target. The query node could then decrypt all of the sensitive information and apply any of the various localization algorithms in existence. Such an approach relies on a *trusted* query node. However, as underlined in the introduction, such trustfulness is risky to assume. To address this issue, we propose two problem setups. In the first one, we introduce an *untrusted third party* who is assumed to not collude with the *untrusted query node*

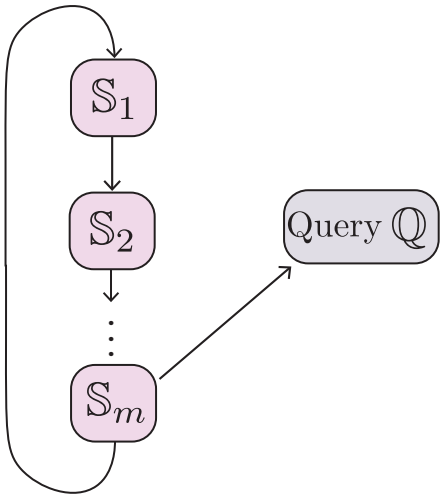



**Figure 2.4:** Secure Multi-Party Computation (SMC)-based setup.

for reasons explained in following sections. This *untrusted third party* (named the aggregator) is responsible for collecting all information from all observers, executing the localization algorithms in a privacy-preserving fashion, and sending the final estimate to the query node. This setup is shown in Figure 2.3. In the second setup, shown in Figure 2.4, the observers co-operate together to compute the final location of the target in a *secure multiparty computation* (SMC) fashion. In what follows, we describe the different entities involved in the localization process as well as the notions of privacy which we aim to achieve with our proposed algorithms.

### 2.3.1 Entities

The localization process involves the following entities:

- **Observers (or Sensors) $\mathbb{S}_i$:** Entities that make some measurements of the target that is then used for localization. In this chapter, these measurements are range estimates, as shown in Figure 2.1. We assume that we have $m$ observers (sometimes referred to as Sensors or anchors). Each observer $\mathbb{S}_i$ (with $k$ being the index of the observer, i.e. $k \in \{1, \ldots, m\}$) possesses three sensitive data entities $\mathbb{S}_i = (x_i, y_i, d_i)$, where $(x_i, y_i) \in \mathbb{R}^2$ denotes the x-y position of the $i$th observer in 2-dimensional space[1] and $d_i \in \mathbb{R}$ denotes the measured distance between the observer and a target $\mathbb{T}$ as described before. The objective of the proposed protocols in this chapter is to ensure the privacy of all observer locations $(x_i, y_i)$ as well as the distance $d_i$. Some of the observers are adversarial ones which launch false data injection attacks by reporting false observer locations $(x_i, y_i)$ and/or false distance $d_i$.

[1]For the sake of simplicity we focus on localization in 2-dimensional space, although the described algorithms can be naturally extended to localization of objects in 3-dimensional space.

- **Target $\mathbb{T}$:** A passive entity whose location $z_{\mathbb{T}} = (x_{\mathbb{T}}, y_{\mathbb{T}})$ needs to be computed using the sensitive information possessed by the observers.
- **Aggregator $\mathbb{A}$:** An *untrusted party* which has a known public key $\text{pk}_{\mathbb{A}}$ and a private key $\text{sk}_{\mathbb{A}}$. Upon receiving all encrypted information sent by observers, it applies a localization algorithm in order to calculate the (encrypted) target location.
- **Query Node $\mathbb{Q}$:** An *untrusted party* that has a known public key $\text{pk}_{\mathbb{Q}}$ and a hidden private key $\text{sk}_{\mathbb{Q}}$. The query node is the only node that is entitled to know the target location. This query node can be physically the same as the target, or it can be another entity (other than the aggregator, in order to preserve privacy).

Similarly, the SMC-based localization process involves the same previous entities except that rather than a single aggregator computing the localization algorithm, the algorithm is computed in a distributed fashion between all observers as shown in Figure 2.4.

### 2.3.2 Attacker Model

We focus on both active (false data injection) attacks as well as passive (privacy-leaking) attacks in which all entities involved in the localization process are *honest-but-curious*. That is, we assume that observers truthfully report their measured distances and that the aggregator or the observers in the SMC-based setup are following the localization algorithms correctly with the exception that some malicious observers may adversarially send false data about their locations and/or distance to the target.

### 2.3.3 Privacy Definitions

We define our privacy notions with the intuition that each entity involved in the localization algorithm should learn nothing about the sensitive information possessed by the other entities even if some of them collude together. However, we note that there exists some immutable privacy leak whenever the query node is involved. This can be defined as follows:

**Immutable privacy leak:** After each run of any localization protocol, the query node learns that all observers lie inside a circle whose center is the target location $z_{\mathbb{T}}$ and whose radius is the maximum sensing range (or RF communication range) of the observers. With this definition of *immutable privacy leaks*, we can define our privacy goals as follows:

- **Observer Obliviousness:** By the end of multiple runs of protocol execution, if all but one observers collude—by exchanging their private values—the coalition should learn nothing about the remaining observer.
- **Aggregator Obliviousness:** By the end of multiple runs of protocol execution, if the aggregator $\mathbb{A}$ colludes with all observers but one, the coalition learns nothing about the remaining observer.

**Table 2.1:** Summary of the privacy guarantees for the three proposed protocols.

| Protocol | Observer Obliviousness | Aggregator Obliviousness | Non-colluding Aggregator Obliviousness | Query node Obliviousness |
|---|---|---|---|---|
| Poly-LSQ (Theorem 2.6.3) | ✓ | ✓ | ✓ | ✓ |
| Poly-AP (Theorem 2.7.1) | ✓ | ✗ | ✓ | ✓ |
| SMC-Poly-AP (Theorem 2.7.2) | ✓ | N/A | N/A | ✗ |

- **Non-colluding Aggregator Obliviousness:** A protocol satisfies the non-colluding aggregator obliviousness property if after multiple runs of the protocol the aggregator $\mathbb{A}$ learns nothing about the sensitive information of the observers.

- **Query Node Obliviousness:** By the end of multiple runs of protocol execution, if the query node $\mathbb{Q}$ colludes with all observers but one, the coalition learns nothing about the remaining observer other than what is implied by the *immutable privacy leak*.

Our proposed protocols named Polyhedra-based Least-Squares localization (Poly-LSQ), Polyhedra-based Alternating Projection localization (Poly-AP), and Secure Multiparty computation Poly-AP localization (SMC-Poly-AP) achieve different utility and privacy guarantees. A summary of their privacy guarantees is given by Table 2.1.

**Remark 2.3.1.** *The traditional privacy notion of a "zero knowledge protocol" [44] considers a protocol secure if the adversarial agents do not leak any information other than what is implied by the final output (the computed target location obtained by the Query node in our problem). It is crucial to note that the previous privacy definitions ask for more stringent requirements in the sense that they require that even the final output should not leak any information about the sensitive information.*

Note that in all the previous privacy notions the aggregator $\mathbb{A}$ and query node $\mathbb{Q}$ are not allowed to combine their respective data in any malicious manner to infer more about the observers. This constraint is implicitly assumed in many privacy-preserving techniques like differential privacy where a malicious aggregator which does not corrupt the data correctly leads to hindering the privacy of the system. In practice, such a constraint is implemented by natural division of labor. Examples of which appear in multi-institution financial data analysis where an organization aggregates sensitive financial data gathered from multiple competing financial intuitions and it is trusted not to collude with any of them [45]. Because of this natural division of labor and this very typical publish-subscribe model, we can safely assume that the two parties - querier and aggregator - operate under different authorities and are disallowed to collude.

### 2.3.4 Utility and Performance Measures

Recall that localization systems are used in many critical systems (e.g., gunshot detection) for which the accuracy of localizing the object or the event is pivotal. Therefore, protocols which establish privacy by degrading localization accuracy by perturbing measurements and/or adding noise (e.g., differential privacy-based techniques) present problems in a practical setting. Furthermore, as motivated in the introduction, many of the observers used in localization are battery-powered. Since network interfaces (e.g. radio) are known to be a major source of power drain [46], it is then crucial to design protocols that minimize the number of messages exchanged within the protocol. Therefore, we define the utility notions as follows:

- **Localization Error:** The distance between the actual target location (ground truth) and the calculated (or estimated) target location.
- **Execution Time:** The time needed by the aggregator to fuse the measurements and produce the final estimate of the target.
- **Observer Communication Cost:** Total number of bits sent and received by all observers.

We aim to achieve the best utilities by decreasing the aforementioned utilities measures.

### 2.3.5 Resilience Definition

Under a strict minority of malicious active adversaries, we define *resilience* as a negligible increase in the overall localization error compared to when no active adversaries are present. The increased communication cost and execution time, incurred to achieve protocol resilience should also be negligible.

We now provide a brief overview of prior work and comment on the suitability of existing techniques to our problem setting.

## 2.4 Related Work

A closely-related problem to localization with private observations is that of secure data aggregation either in a distributed setting or in the presence of an untrusted aggregator. We classify the various techniques that have been proposed for the above problem into the following broad categories and discuss why they are not suitable for our problem setting.

### 2.4.1 Differential Privacy Techniques

Differential privacy techniques rely on the addition of structured noise to the data before sharing it with the aggregator such that the aggregated data preserves the privacy of each

individual datum. Differential privacy [47, 48] provides rigorous privacy guarantees by requiring the output of the aggregator to not change significantly even if an observer has opted out of the data collection. More recently, variant schemes such as local differential privacy [49] and geo-indistinguishability [50] have been designed to ensure differential privacy for location data. However, to achieve these guarantees in a distributed setting without a trusted aggregator, each observer needs to add noise to make their own data differentially private. This results in an aggregated noise that far exceeds the required amount to ensure differential privacy for the aggregated result, severely degrading the accuracy of the localization result and making it unsuitable for use in critical systems.

To overcome the addition of excessive noise, a combination of homomorphic encryption strategies with distributed noise generation has been proposed. In [38], each observer generated his share of the aggregate noise required for differential privacy [51] and sent encrypted and obfuscated data to the aggregator. In [37], differential privacy is achieved by perturbing the most significant coefficients of the discrete Fourier transform of the query answers. The distributed noise is generated by participants using a vector of four Gaussian random variables. The obfuscated values are encrypted using an additive homomorphic threshold Paillier cryptosystem. The aggregator operates on the received encrypted values to compute the aggregate sum. Threshold-based decryption is performed in a distributed fashion using the private key belonging to each participant. Similarly in [38], homomorphic encryption is used to secure shared data and the encryption key is generated using the current iteration number and the originally established key. Differential privacy of the aggregated data is established using noise drawn from a symmetric geometric distribution. The above techniques are specifically designed for simple aggregation (mean computation) using collected data whereas in our case the localization process requires more sophisticated operations making these noise generation strategies inapplicable.

### 2.4.2 Secure Multi-Party Computation

A multi-party computation protocol is secure if no information about the private data held by the parties can be inferred from the messages exchanged during the execution of the protocol. The only information that can be inferred about the private data is that which could be inferred from the output of the overall function alone [52,53]. This makes secure multiparty computation an excellent candidate for secure aggregation without requiring an aggregator. A comparative study on using SMC for secure aggregation can be found in [54], where again the aggregation of the data is limited to performing summation operations, which, as noted before, is insufficient for localization purposes. Furthermore, Yao's work on garbled circuits [52] has resulted in the construction of general-purpose SMC circuits [42] that can compute any arbitrary polynomial function over encrypted data. However, these general-purpose black box techniques are not only complicated to implement, but also very slow because it requires multiple rounds of message exchanges and significantly increases the communication cost of the observer. We made these observations from our experimentation with these techniques and similar conclusions have also been noted in [55].

### 2.4.3 Fully Homomorphic Encryption

Fully Homomorphic Encryption [41] has recently been used as a countermeasure for server-side information leakages. Over the past few years, significant work (in the form of an FHE library [56]) has gone into making it practical, and it has been used for computationally expensive tasks over genome data [57] and recently for classification over encrypted data [58]. However, the resulting scheme is still impractical for real-time localization services owing to high latencies [59] as reported in our experiments in Section 2.10. Separately, the orthogonal problem of maintaining the integrity of the localization process in the presence of colluding observers has been studied in [60] and [61].

### 2.4.4 Privacy Through Selective Obfuscation

One common technique to preserve privacy is through obfuscation via addition of noise to specific dimensions of the sensor signal such that, when aggregated, the noise cancels out (i.e. sums to zero) in the dimension of interest. Specifically in the context of localization, such intermediate data obfuscation has been formulated in [1, 62], where each node generates random noise which is then added to its private input. Unfortunately, this technique assumes a non-adversarial setup where the observer nodes are always fully compliant and follow the designed localization protocol. However, in case of adversarial nodes that do not add the right amount of noise, and/or colluding nodes that reveal their noise values, the localization accuracy and the privacy guarantees can be significantly degraded, violating our model requirements in Section 2.3.2.

## 2.5 Preliminaries

### 2.5.1 Homomorphic Encryption

A homomorphic cryptosystem is a cryptographic primitive which supports computation over encrypted data. This provides the foundation for the proposed protocols. Two of the most famous homomorphic cryptosystems are *Paillier additive homomorphic encryption* and the *leveled fully homomorphic encryption* cryptosystem. We discuss these cryptosystems briefly below.

#### 2.5.1.1 Paillier Homomorphic Cryptosystem

Our protocols make heavy use of a particular homomorphic cryptosystem named Paillier additive homomorphic cryptosystem [40]. This is a probabilistic public key cryptography scheme that allows two important operations, namely (i) addition of two encrypted values and (ii) multiplication of an encrypted value by a plaintext value. That is, if we denote by $[\![a]\!]_{\text{pk}}$ the encryption of $a$ using the public key pk then the Paillier cryptosystem supports two operations namely $\oplus$, which is addition over encrypted data, (and accordingly $\ominus$)

and $\otimes$, which is multiplication by a plaintext, such that:

$$\begin{aligned}
&\textsc{Decrypt}_{\text{sk}}(\llbracket a \rrbracket_{\text{pk}} \oplus \llbracket b \rrbracket_{\text{pk}}) = \textsc{Decrypt}_{\text{sk}}(\llbracket a + b \rrbracket_{\text{pk}}) = a + b \\
&\textsc{Decrypt}_{\text{sk}}(a \otimes \llbracket b \rrbracket_{\text{pk}}) = \textsc{Decrypt}_{\text{sk}}(\llbracket a \times b \rrbracket_{\text{pk}}) = a \times b
\end{aligned}$$

where sk is the private key associated with the public key pk. We denote by $\mathbb{Z}$ the set of integers. Note that these two properties can be generalized to perform multiplication between a plaintext matrix and a vector of encrypted variables. That is given a vector $x \in \mathbb{Z}^n$ and a matrix $M \in \mathbb{Z}^{n \times n}$, we can compute $M \otimes \llbracket x \rrbracket_{\text{pk}}$ as:

$$M \otimes \llbracket x \rrbracket_{\text{pk}} = \begin{bmatrix} M(1,1) \otimes \llbracket x(1) \rrbracket_{\text{pk}} \oplus \ldots \oplus M(1,n) \otimes \llbracket x(n) \rrbracket_{\text{pk}} \\ \vdots \\ M(n,1) \otimes \llbracket x(1) \rrbracket_{\text{pk}} \oplus \ldots \oplus M(n,n) \otimes \llbracket x(n) \rrbracket_{\text{pk}} \end{bmatrix},$$

where with some abuse of notation, we use $\otimes$ in $M \otimes \llbracket x \rrbracket_{\text{pk}}$ to denote the multiplication of a plaintext matrix $M$ with an encrypted vector $\llbracket x \rrbracket_{\text{pk}}$. An important fact about Paillier cryptosystem, that we will use later, that its encryption is probabilistic. That is, given $a$ and $b$ such that $a = b$, the encryption $\llbracket a \rrbracket_{\text{pk}} \neq \llbracket b \rrbracket_{\text{pk}}$ in general and similarly $1 \otimes \llbracket a \rrbracket_{\text{pk}} \neq \llbracket a \rrbracket_{\text{pk}}$ in general. The security guarantees of the Paillier cryptosystem rely on a standard cryptographic assumption named Decisional Composite Residuosity Assumption (DCRA) [40].

#### 2.5.1.2 Fully Homomorphic Encryption Cryptosystem

The Fully Homomorphic cryptosystem is also a public key cryptosystem which supports (i) addition of two encrypted values and (ii) multiplication of two encrypted values. If we denote by $\llbracket a \rrbracket_{\text{pk}}^{\text{FHE}}$ the encryption of $a$ using the public key pk then FHE supports two operations $\oplus^{\text{FHE}}$ and $\otimes^{\text{FHE}}$ such that:

$$\begin{aligned}
\textsc{Decrypt}_{\text{sk}}(\llbracket a \rrbracket_{\text{pk}}^{\text{FHE}} \oplus^{\text{FHE}} \llbracket b \rrbracket_{\text{pk}}^{\text{FHE}}) &= \textsc{Decrypt}_{\text{sk}}(\llbracket a + b \rrbracket_{\text{pk}}^{\text{FHE}}) \\
&= a + b \\
\textsc{Decrypt}_{\text{sk}}(\llbracket a \rrbracket_{\text{pk}}^{\text{FHE}} \otimes^{\text{FHE}} \llbracket b \rrbracket_{\text{pk}}^{\text{FHE}}) &= \textsc{Decrypt}_{\text{sk}}(\llbracket a \times b \rrbracket_{\text{pk}}^{\text{FHE}}) \\
&= a \times b
\end{aligned}$$

The security guarantees of the leveled FHE algorithm presented in [63] are based on a cryptographic assumption named Ring Learning With Error (RLWE) [63]. The protocols described in this chapter rely on the Paillier cryptosystem, while FHE is used primarily for comparison.

### 2.5.2 Secure Comparison Protocol

The second basic block of our protocols is the comparison over encrypted data. In this protocol, if party A has a set of encrypted data $\llbracket a_1 \rrbracket_{\text{pk}_B}, \ldots, \llbracket a_m \rrbracket_{\text{pk}_B}$ encrypted using

the Paillier cryptosystem using the public key of party B, then by the end of this protocol, entity A knows the index of the maximum (or minimum) value, i.e. A learns $\arg\max\{a_1, \ldots, a_m\}$, and nothing else whenever both A and B are honest-but-curious. Different instantiations of this protocol are proposed in literature and are either based on data obfuscation [58] or on Garbled Circuits [43]. We call this *secure comparison* protocol SECCOMPARE $([\![a_1]\!]_{\text{pk}_B}, \ldots, [\![a_m]\!]_{\text{pk}_B})$.

### 2.5.3 Floats Encoding

Recall that cryptosystems are designed to encrypt and decrypt integers. However, localization measurements and algorithms depend on floating point numbers. Hence, there is a desire to carry out encrypted computation over floating point numbers by encoding them as integers with an acceptable quantization error. A common encoding is achieved by scaling all floating point variables by a constant factor [58]. We denote the encode operation of float number $a$ by $\mathbb{EN}(a)$. Let us define $\phi_{add}$ and $\phi_{mul}$ as the accumulated error after addition and multiplication operation as shown in (2.4) and (2.5), respectively.

$$\phi_{add} := \left| \mathbb{EN}(a+b) - \Big( \mathbb{EN}(a) + \mathbb{EN}(b) \Big) \right| \tag{2.4}$$

$$\phi_{mul} := \left| \mathbb{EN}(a \times b) - \Big( \mathbb{EN}(a) \times \mathbb{EN}(b) \Big) \right| \tag{2.5}$$

Optimal encoding mechanism should have $\phi_{mul} = \phi_{add} = 0$. Standard work over encrypted data represents floating numbers by appropriate scaling. However, float scaling representation causes $\phi_{mul}$ to equal the scale factor after each multiplication operation. Therefore, it can not be used with arbitrary number of float numbers multiplication.

We take a different approach with improved numerical performance as supported by our experiments in Section 2.10 and as used by Google's Encrypted BigQuery Client[2], where each float number $a$ is encoded as $m_a \times 10^{-e_a}$ where $m_a$ and $e_a$ are positive integers referred to as the mantissa and the positive part of the exponent, respectively. We omit the encoding symbol $\mathbb{EN}$ for simplicity. It is enough to encrypt only the mantissa to protect the privacy of the float number. That is, the encryption of $a = m_a \times 10^{-e_a}$ becomes $[\![a]\!]_{\text{pk}} = [\![m_a]\!]_{\text{pk}} \times 10^{-e_a}$. By using the addition and multiplication primitives of the Paillier cryptosystem, we can perform addition and multiplication over this encoding

[2]https://github.com/google/encrypted-bigquery-client

as follows:

$$a = m_a \times 10^{-e_a}, [\![a]\!]_{\text{pk}} = [\![m_a]\!]_{\text{pk}} \times 10^{-e_a}$$
$$b = m_b \times 10^{-e_b}, [\![b]\!]_{\text{pk}} = [\![m_b]\!]_{\text{pk}} \times 10^{-e_b}$$

**Multiplication:**

$$[\![c]\!]_{\text{pk}} = a \otimes [\![b]\!]_{\text{pk}} \quad \Leftrightarrow \quad [\![m_c]\!]_{\text{pk}} = m_a \otimes [\![m_b]\!]_{\text{pk}},\ e_c = e_a + e_b$$

**Addition:**

$$[\![c]\!]_{\text{pk}} = [\![a]\!]_{\text{pk}} \oplus [\![b]\!]_{\text{pk}} \Leftrightarrow$$
$$\begin{cases} [\![m_c]\!]_{\text{pk}} = [\![m_a]\!]_{\text{pk}} \oplus (10^{-(e_b - e_a)} \otimes [\![m_b]\!]_{\text{pk}}),\ e_c = e_b \ \textbf{if}\ e_a \leq e_b \\ [\![m_c]\!]_{\text{pk}} = (10^{-(e_a - e_b)} \otimes [\![m_a]\!]_{\text{pk}}) \oplus [\![m_b]\!]_{\text{pk}},\ e_c = e_a \ \textbf{if}\ e_a > e_b \end{cases}$$

Negative numbers are supported by defining ranges for the encoded number over the ring $r$ of the Paillier cryptosystem. Positive and negative number ranges are $(0, r/3]$ and $(r/3, 2r/3]$, respectively. The remaining range $(2r/3, r)$ is used for overflow detection. The exponent must be negative to support floats. Multiplication leads to decreasing the exponents and raising the mantissa value to a specific power. On the other hand, summation picks the smaller exponent to be the resultant exponent. Smaller exponent representation causes also raising the mantissa value to a specific power. Thus, a combination of addition and multiplication operations might cause an overflow in the encrypted domain. The presented encoding accelerates reaching the overflow limits. Decryption and re-encoding the ciphertext after a specific number of operation is a simple solution to increase the exponent and decrease the mantissa to avoid overflow. We must also ensure not to overflow Paillier's message space while doing secure operations.

## 2.6 Polyhedra-based Localization Algorithm: Least Squares

In this section, we introduce our first contribution of this chapter: a re-formalization of the traditional localization algorithm in Section 2.2. We aim to ensure the privacy guarantees while depending only on *additive* homomorphic encryption rather than fully homomorphic encryption or garbled circuits. We call this localization algorithm *polyhedra-based localization*, and we describe it in detail in this section. We focus in this section on the case where no false data injection attack is present and focus only on the problem of estimating the location of the target node in a privacy-preserving manner. In Section 2.8, we show how to augment the proposed privacy preserving protocols to achieve resilience against false data injection attacks.

### 2.6.1 Polyhedra-Based Localization

The intuition behind the *polyhedra-based localization* algorithm is as follows. Traditional geometry-based localization methods like the least squares approach discussed

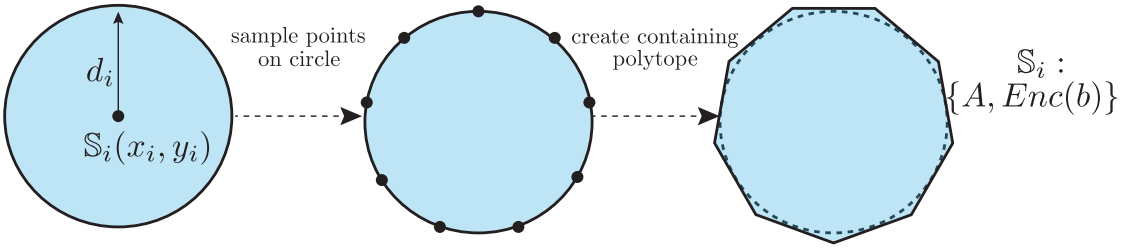



**Figure 2.5:** Discretizing the circle as a polyhedron

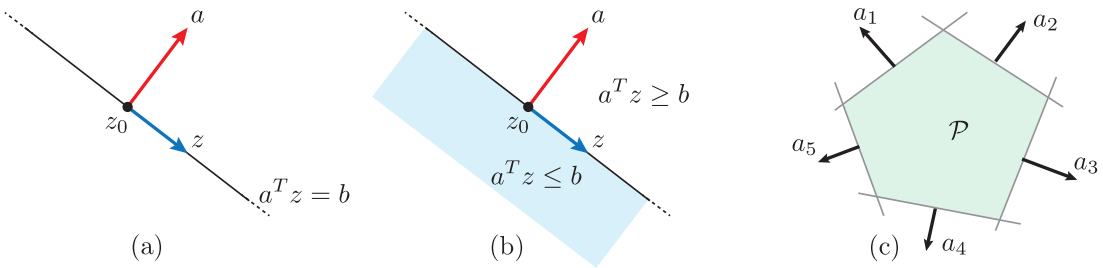



**Figure 2.6:** Algebraic representation of polyhedra: (a) a hyperplane in $\mathbb{R}^2$, (b) a half space in $\mathbb{R}^2$, and (c) a polyhedron defined by $f$ half spaces.

in Section 2.2 treat range measurements as a constraint that the target $\mathbb{T}$ lies on some circle whose radius is equal to the estimated distance and whose center is the coordinate of the sensor. In essence, communicating this constraint requires communicating both radius ($d_i$) and center $(x_{\mathbb{S}_i}, y_{\mathbb{S}_i})$, exposing sensitive information about the sensors themselves. Rather than parameterizing the sensing range as a circle, we seek another parameterization that can still represent the estimated range without sacrificing sensor privacy. Towards this end, we propose using polyhedra to represent the sensed range. As shown in Figure 2.5, we can obtain such polyhedra by sampling the circumference of the circle representing the measured range. To represent these polyhedra, we start by reviewing some geometric definitions illustrated in Figure 2.6:

- Hyperplane: a hyperplane in the 2D Cartesian plane represents a line. A hyperplane can be parameterized using a unit normal vector $a$ and an offset $b$ such that all points on the hyperplane are: $\{z = (x, y) \mid a^T z = b\}$

- Halfspace: each hyperplane splits $\mathbb{R}^2$ into two halfspaces: One halfspace which lies in the direction of the normal vector and the other in the reverse direction. The second of these halfspaces can be compactly written as:

$$\mathcal{H} = \{z = (x, y) \mid a^T z \leq b\}$$

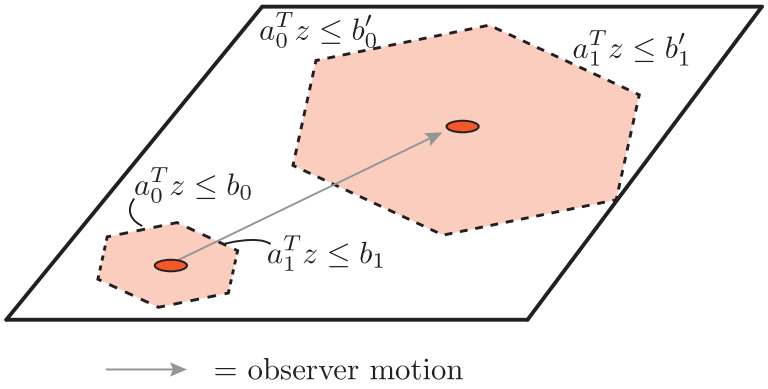



**Figure 2.7:** Polyhedron discretization of range circle under translation and scaling (varying $b_i$ and fixed $a_i$).

- Polyhedron: a polyhedron is the intersection of $f$ halfspaces and can be represented using their unit normal vectors and offsets as:

$$\mathcal{P}(A,b) = \{z = (x,y) \,|Az \leq b\} \quad \text{where} \quad A = \begin{bmatrix} a_1^T \\ \vdots \\ a_f^T \end{bmatrix}, b = \begin{bmatrix} b_1 \\ \vdots \\ b_f \end{bmatrix}$$

Since any polyhedron is parameterized by the matrix $A$ and the vector $b$, we use the following shorthand notation: $\mathcal{P}(A,b)$. An intrinsic characteristic of polyhedra is the following:

**Fact 2.6.1.** *For a polyhedron $\mathcal{P}(A,b)$, the matrix $A$ (which is the collection of the normal vectors of the facets) represents only the shape of the polyhedron, specifying neither the location nor the scale of the polyhedron.*

Figure 2.7 shows an example of this geometric fact. In Figure 2.7 we show two different polyhedra that have exactly the same matrix $A$ (same normal vectors to facets) but different $b$ vectors. The resulting two polyhedra have different locations and sizes. Hence, we use the notation $A_i$ and $b_{\mathbb{S}_i}$ to denote the polyhedron constructed by the $i$th observer where the subscript $\mathbb{S}_i$ in $b_{\mathbb{S}_i}$ is used to stress the fact that the sensitive information of the observer is encoded inside the vector $b_{\mathbb{S}_i}$ and not in the matrix $A_i$.

Given a set of polyhedra from different observers, our objective is to fuse them in order to calculate the target location. We note that the target lies in the intersection of all of the $m$ polyhedra constructed by observers. This intersection creates another polyhedron (denoted by $\mathcal{P}_{\mathbb{T}}$) that can be represented as the intersection of all halfspaces of all the $m$

polyhedra. That is:

$$\mathcal{P}_{\mathbb{T}} = \{z = (x, y) \in \mathbb{R}^2 \mid \overline{A}z \leq \overline{b}_{\mathbb{S}}\} \text{ where } \overline{A} = \begin{bmatrix} A_1 \\ \vdots \\ A_m \end{bmatrix}, \overline{b}_{\mathbb{S}} = \begin{bmatrix} b_{\mathbb{S}_1} \\ \vdots \\ b_{\mathbb{S}_m} \end{bmatrix}$$

Since the target location lies inside the polyhedron $\mathcal{P}_{\mathbb{T}}$, we can calculate the target location by solving the following optimization problem:

$$(\hat{x}_{\mathbb{T}}, \hat{y}_{\mathbb{T}}) = \arg \min_{(\hat{x}_{\mathbb{T}}, \hat{y}_{\mathbb{T}}) \in \mathbb{R}^2} \left\| \overline{A}\hat{z}_{\mathbb{T}} - \overline{b}_{\mathbb{S}} \right\|_2^2 \tag{2.6}$$

Such a solution can be calculated as:

$$(\hat{x}_{\mathbb{T}}, \hat{y}_{\mathbb{T}}) = \left(\overline{A}^T \overline{A}\right)^{-1} \overline{A}^T \overline{b}_{\mathbb{S}} \tag{2.7}$$

**Remark 2.6.2.** *The major difference between the solution* (2.7) *in the polyhedra-based algorithm and the corresponding one in the standard least squares one* (2.2) *is the fact that the matrix $\overline{A}$ does not hold any sensitive information and hence can be communicated as plaintext compared to the necessity of encrypting the matrix $A_{\mathbb{S}}$ in the standard least squares algorithm.*

Hence, by reducing the encryption requirement to only encrypting the vector $\overline{b}_{\mathbb{S}}$, our polyhedra-based algorithm is able to use additive homomorphism rather than full homomorphism. This in turn results in a much more efficient algorithm in terms of execution time as reflected by the experimental results shown in Section 2.10. The details of the protocol are given in the next subsection.

## 2.6.2 Poly-LSQ: Polyhedra-based Least-Squares Localization

Similarly to the traditional least squares localization algorithm, we implement least squares to our new localization reformalization using homomorphic encryption. The algorithm consists of three main functions as follows:

### 2.6.2.1 Encrypt

The shape of each polyhedron is fixed if we the same circle sampling is used every time. We can then assume without loss of generality that the matrix $A$ is known to all parties and is not required to be communicated. Hence, we focus on encrypting the vectors $\overline{b}_{\mathbb{S}_i}$ for all sensors $i \in \{1, \ldots, m\}$. For each facet, the $i$th observer encrypts the corresponding offset vector $\overline{b}_{\mathbb{S}_i}$ using the public key of the query node $\text{pk}_{\mathbb{Q}}$ followed by another encryption using the public key of the aggregator $\text{pk}_{\mathbb{A}}$ resulting in the following

message content:

$$msg_i = [\![[\![\bar{b}_{\mathbb{S}_i}]\!]_{\text{pk}_\mathbb{Q}}]\!]_{\text{pk}_\mathbb{A}} \quad \forall i \in \{1, \ldots, m\} \tag{2.8}$$

Note that the purpose of the second encryption is to prevent the query node from accessing $[\![\bar{b}_{\mathbb{S}_i}]\!]_{\text{pk}_\mathbb{Q}}$ which it could then decrypt using its own private key $\text{sk}_\mathbb{Q}$. Hence, this second encryption does not need to be carried out using a homomorphic cryptosystem but rather any public key cryptosystem. For sake of simplicity, and since the Paillier cryptosystem is indeed a public key cryptosystem, we use the same notation for this second encryption.

#### 2.6.2.2 Aggregate

Once all observers have sent their messages to the aggregator $\mathbb{A}$, the aggregator constructs the matrix $\tilde{A}$ by stacking all the matrices $A$ together. Similarly, it uses its secret key $\text{sk}_\mathbb{A}$ to decrypt $[\![[\![\bar{b}_{\mathbb{S}_i}]\!]_{\text{pk}_\mathbb{Q}}]\!]_{\text{pk}_\mathbb{A}}$ and constructs the vector $[\![\bar{b}_\mathbb{S}]\!]_{\text{pk}_\mathbb{Q}}$, i.e.,

$$\overline{A} = \begin{bmatrix} A_1 \\ \vdots \\ A_m \end{bmatrix}, \quad [\![\bar{b}_\mathbb{S}]\!]_{\text{pk}_\mathbb{Q}} = \begin{bmatrix} [\![\bar{b}_{\mathbb{S}_1}]\!]_{\text{pk}_\mathbb{Q}}, \\ \vdots \\ [\![\bar{b}_{\mathbb{S}_m}]\!]_{\text{pk}_\mathbb{Q}} \end{bmatrix}$$

In the next step, $\mathbb{A}$ computes the encrypted value of the target estimate $(\hat{x}_\mathbb{T}, \hat{y}_\mathbb{T})$ as:

$$(\hat{x}_\mathbb{T}, \hat{y}_\mathbb{T}) = \left(\overline{A}^T \overline{A}\right)^{-1} \overline{A}^T \otimes [\![\bar{b}_\mathbb{S}]\!]_{\text{pk}_\mathbb{Q}}$$

Also, our new reformalization allows to solve (2.6) using the iterative gradient descent method by starting at a fixed initial estimate $\hat{z}_\mathbb{T}^{(0)} = (\hat{x}_\mathbb{T}, \hat{y}_\mathbb{T}) = \begin{bmatrix} 0 & 0 \end{bmatrix}^T$ and then updating the estimate iteratively as:

$$[\![\hat{z}_\mathbb{T}^{(i+1)}]\!]_{\text{pk}_\mathbb{Q}} = [\![\hat{z}_\mathbb{T}^{(i)}]\!]_{\text{pk}_\mathbb{Q}} \oplus \eta \overline{A}^T \otimes \left( [\![\bar{b}_\mathbb{S}]\!]_{\text{pk}_\mathbb{Q}} \ominus \left( \overline{A} \otimes [\![\hat{z}_\mathbb{T}^{(i)}]\!]_{\text{pk}_\mathbb{Q}} \right) \right),$$

where $\eta$ is the gradient step size and must be chosen such that $0 < \eta < 2/\lambda_{\max}\{\overline{A}^T \overline{A}\}$. The max eignvalue of symmetric matrix is denoted by $\lambda_{\max}\{\}$. Thanks to the polyhedra representation, the matrix $\overline{A}$ is plaintext and hence the matrix $(\overline{A}^T \overline{A})^{-1} \overline{A}^T$ computed on plaintext before using Paillier cryptosystem to multiply it by the encrypted vector $[\![\bar{b}_\mathbb{S}]\!]_{\text{pk}_\mathbb{Q}}$. Also, the gradient descent formula can now be executed using the Paillier cryptosystem as we can multiply by unencrypted matrix $\overline{A}$.

#### 2.6.2.3 Decrypt

Finally, the aggregator $\mathbb{A}$ sends the computed estimate $[\![\hat{z}_\mathbb{T}]\!]_{\text{pk}_\mathbb{Q}}$ to the query node $\mathbb{Q}$. The query node decrypts the message using its private key $\text{sk}_\mathbb{Q}$ to retrieve the final estimate $\hat{z}_\mathbb{T}$.

#### 2.6.2.4 Privacy Analysis of the Poly-LSQ Protocol

We show that the proposed Poly-LSQ protocol satisfies the strong privacy notions. This is summarized in the following theorem:

**Theorem 2.6.3.** *Assume that all entities are honest-but-curious and under standard cryptographic assumptions (namely the Decisional Composite Residuosity assumption), the Poly-LSQ localization protocol ensures (i) observer obliviousness, (ii) aggregator obliviousness, and (iii) query node obliviousness.*

*Proof.* **Observer obliviousness:** Without loss of generality, assume that observers $\mathbb{S}_1$ , . . . , $\mathbb{S}_{m-1}$ are colluding together to form an adversarial observer $\mathbb{S}_{\text{ADV}}$ whose objective is computing the remaining sensor information $b_{\mathbb{S}_m}$. After one iteration of the protocol, the information view of this adversarial observer $V_{\mathbb{S}_{\text{ADV}}}$ is:

$$V_{\mathbb{S}_{\text{ADV}}} = \left( A, b_{\mathbb{S}_1}, \ldots, b_{\mathbb{S}_{m-1}}, [\![[\![b_{\mathbb{S}_m}]\!]_{\text{pk}_{\mathbb{Q}}}]\!]_{\text{pk}_{\mathbb{A}}}, \text{pk}_{\mathbb{Q}}, \text{pk}_{\mathbb{A}} \right)$$

As shown by Fact 2.6.1, the matrix $A$ describes only the shape of the polyhedra and reveals nothing about the location or the size of the polyhedra. Furthermore, the sensor information $b_{\mathbb{S}_1}, \ldots, b_{\mathbb{S}_{m-1}}$ themselves reveal nothing about the remaining sensor $b_{\mathbb{S}_m}$. Therefore, we conclude that the adversarial $\mathbb{S}_{\text{ADV}}$ needs to break the encryption of the Paillier cryptosystem to retrieve $b_{\mathbb{S}_m}$ from $[\![[\![b_{\mathbb{S}_m}]\!]_{\text{pk}_{\mathbb{Q}}}]\!]_{\text{pk}_{\mathbb{A}}}$ which is ensured under the DCRA assumption. The same argument holds for multiple rounds of the protocol.

**Aggregator obliviousness:** We consider an aggregator $\mathbb{A}$ which runs gradient descent and, without loss of generality, assume that the aggregator $\mathbb{A}$ is colluding with the observers $\mathbb{S}_1, \ldots \mathbb{S}_{m-1}$ to form an adversarial aggregator $\mathbb{A}_{\text{ADV}}$ whose objective is computing the remaining sensor information $b_{\mathbb{S}_m}$. The information view of this adversarial aggregator $V_{\mathbb{A}_{\text{ADV}}}$ is:

$$V_{\mathbb{A}_{\text{ADV}}} = \left( A, b_{\mathbb{S}_1}, \ldots, b_{\mathbb{S}_{m-1}}, [\![b_{\mathbb{S}_m}]\!]_{\text{pk}_{\mathbb{Q}}}, \hat{z}_{\mathbb{T}}^{(0)}, [\![\hat{z}_{\mathbb{T}}^{(1)}]\!]_{\text{pk}_{\mathbb{Q}}}, \ldots, [\![\hat{z}_{\mathbb{T}}^{(K)}]\!]_{\text{pk}_{\mathbb{Q}}}, \text{pk}_{\mathbb{Q}}, \text{pk}_{\mathbb{A}}, \text{sk}_{\mathbb{A}} \right).$$

Using the same argument used in the observer obliviousness, we conclude that $A$, $b_{\mathbb{S}_1}$, . . . , $b_{\mathbb{S}_{m-1}}$ leaks nothing about $b_{\mathbb{S}_m}$. Similarly, the initial estimate $\hat{z}_{\mathbb{T}}^{(0)}$ is fixed and does not depend on the sensor information and therefore reveals nothing about $b_{\mathbb{S}_m}$. Since all the remaining information $[\![\hat{z}_{\mathbb{T}}^{(1)}]\!]_{\text{pk}_{\mathbb{Q}}}, \ldots, [\![\hat{z}_{\mathbb{T}}^{(K)}]\!]_{\text{pk}_{\mathbb{Q}}}$ are encrypted using the Paillier cryptosystem which is ensured under the DCRA assumption, we conclude that the adversarial aggregator leaks nothing about $b_{\mathbb{S}_m}$. The same argument holds for multiple rounds of the protocol.

**Query Node Obliviousness:** Finally, without loss of generality, assume that the query node $\mathbb{Q}$ is colluding with the observers $\mathbb{S}_1, \ldots \mathbb{S}_{m-1}$ to form an adversarial query node $\mathbb{Q}_{\text{ADV}}$ whose objective is computing the remaining sensor information $b_{\mathbb{S}_m}$. The informa-

tion view of this adversarial query node $V_{\mathbb{Q}_{\text{ADV}}}$ is:

$$V_{\mathbb{Q}_{\text{ADV}}} = \left( A, b_{\mathbb{S}_1}, \dots, b_{\mathbb{S}_{m-1}}, [\![[\![b_{\mathbb{S}_m}]\!]_{\text{pk}_{\mathbb{Q}}}]\!]_{\text{pk}_{\mathbb{A}}}, \hat{z}_{\mathbb{T}}^{(0)}, \hat{z}_{\mathbb{T}}^{(K)}, \text{pk}_{\mathbb{Q}}, \text{pk}_{\mathbb{A}}, \text{sk}_{\mathbb{Q}} \right)$$

Unlike the aggregator obliviousness, the query node has access to $\hat{z}_{\mathbb{T}}^{(K)}$ which is computed as a function of the sensitive information $\mathbb{S}_m$ and hence may be used to reverse engineer the localization algorithm and leak some information. The remainder of this proof addresses this issue. To that end, assuming that the protocol is executed $L$ times, and by collecting all the information from the $L$ executions together, the information view of this adversarial query node $V_{\mathbb{Q}_{\text{ADV}}}$ can be written as:

$$\begin{aligned} V_{\mathbb{Q}_{\text{ADV}}} = \Bigg( & A, b_{\mathbb{S}_1^1}, \dots, b_{\mathbb{S}_{m-1}^1}, [\![[\![b_{\mathbb{S}_m^1}]\!]_{\text{pk}_{\mathbb{Q}}}]\!]_{\text{pk}_{\mathbb{A}}}, \dots, \\ & b_{\mathbb{S}_1^L}, \dots, b_{\mathbb{S}_{m-1}^L}, [\![[\![b_{\mathbb{S}_m^L}]\!]_{\text{pk}_{\mathbb{Q}}}]\!]_{\text{pk}_{\mathbb{A}}}, \\ & \hat{z}_{\mathbb{T}}^{(0)}, \hat{z}_{\mathbb{T}^1}^{(K)}, \dots, \hat{z}_{\mathbb{T}^L}^{(K)}, \text{pk}_{\mathbb{Q}}, \text{pk}_{\mathbb{A}}, \text{sk}_{\mathbb{Q}} \Bigg) \end{aligned}$$

where we add the superscript $1, \dots, L$ in $\mathbb{S}$ and $\mathbb{T}$ in order to distinguish between the different executions of the protocol. Now, by assuming the role of the adversary, we would like to reverse the gradient descent equations in order to construct the sensitive information. We start by rewriting the equations in the unencrypted form

$$\hat{z}_{\mathbb{T}}^{(K)} = (I - \eta \overline{A}^T \overline{A})^K \hat{z}_{\mathbb{T}}^{(0)} + \underbrace{\sum_{j=0}^{K-1} \eta (I - \eta \overline{A}^T \overline{A})^j \overline{A}^T}_{\tilde{A}} \overline{b}_{\mathbb{S}}, \tag{2.9}$$

where we considered only one run for simplicity. Define the matrix $\tilde{A} \in \mathbb{R}^{2 \times mf}$ as $\tilde{A} = \sum_{j=0}^{K-1} \eta (I - \eta \overline{A}^T \overline{A})^j \overline{A}^T$. The matrices $\overline{A}$ and $\tilde{A}$ are known to adversary query node $\mathbb{Q}_{\text{ADV}}$ as the matrix $A$ is known to her. Recall the definition of the vector $\overline{b}_{\mathbb{S}}$ as:

$$\overline{b}_{\mathbb{S}} = \begin{bmatrix} b_{\mathbb{S}_1} \\ \vdots \\ b_{\mathbb{S}_m} \end{bmatrix} \in \mathbb{R}^{mf}$$

Since $b_{\mathbb{S}_m}$ is the only remaining sensitive information, we can partition both $\tilde{A}$ and $\overline{b}_{\mathbb{S}}$ accordingly as follows:

$$\tilde{A} = \begin{bmatrix} \tilde{A}_{\mathbb{S}\setminus\mathbb{S}_m} & \tilde{A}_{\mathbb{S}_m} \end{bmatrix}, \ \overline{b}_{\mathbb{S}} = \begin{bmatrix} b_{\mathbb{S}\setminus\mathbb{S}_m} \\ b_{\mathbb{S}_m} \end{bmatrix}, \ b_{\mathbb{S}\setminus\mathbb{S}_m} = \begin{bmatrix} b_{\mathbb{S}_1} \\ \vdots \\ b_{\mathbb{S}_{m-1}} \end{bmatrix} \in \mathbb{R}^{(m-1)f}$$

where $\tilde{A}_{\mathbb{S}\setminus\mathbb{S}_m} \in \mathbb{R}^{2\times(m-1)f}$ and $\tilde{A}_{\mathbb{S}_m} \in \mathbb{R}^{2\times f}$ results from the corresponding partitioning. Therefore, we can rewrite (2.9) as

$$\underbrace{\hat{z}_{\mathbb{T}}^{(K)} - (I - \eta\overline{A}^T\overline{A})^K \hat{z}_{\mathbb{T}}^{(0)} - \tilde{A}_{\mathbb{S}\setminus\mathbb{S}_m} b_{\mathbb{S}\setminus\mathbb{S}_m}}_{\tilde{z}_{\mathbb{T}}} = \tilde{A}_{\mathbb{S}_m} b_{\mathbb{S}_m}$$

Noting that all the terms on the left hand side are known to the adversary, we can reduce it once more as:

$$\tilde{z}_{\mathbb{T}} = \tilde{A}_{\mathbb{S}_m} b_{\mathbb{S}_m} \tag{2.10}$$

where

$$\tilde{z}_{\mathbb{T}} = \hat{z}_{\mathbb{T}}^{(K)} - (I - \eta\overline{A}^T\overline{A})^K \hat{z}_{\mathbb{T}}^{(0)} - \tilde{A}_{\mathbb{S}\setminus\mathbb{S}_m} b_{\mathbb{S}\setminus\mathbb{S}_m}. \tag{2.11}$$

Now, by combining all the information from the $L$ experiments, we can aggregate all these information as:

$$\begin{bmatrix} \tilde{z}_{\mathbb{T}^1} \\ \vdots \\ \tilde{z}_{\mathbb{T}^L} \end{bmatrix} = \begin{bmatrix} \tilde{A}_{\mathbb{S}_m} & & \\ & \ddots & \\ & & \tilde{A}_{\mathbb{S}_m} \end{bmatrix} \begin{bmatrix} b_{\mathbb{S}_m^1} \\ \vdots \\ b_{\mathbb{S}_m^L} \end{bmatrix} \tag{2.12}$$

By defining and examining the dimension of the vectors of the (2.12), we note that:

$$\bar{z}_{\mathbb{T}} := \begin{bmatrix} \tilde{z}_{\mathbb{T}^1} \\ \vdots \\ \tilde{z}_{\mathbb{T}^L} \end{bmatrix} \in \mathbb{R}^{2L}, \quad \bar{A}_{\mathbb{S}_m} := \begin{bmatrix} \tilde{A}_{\mathbb{S}_m} & & \\ & \ddots & \\ & & \tilde{A}_{\mathbb{S}_m} \end{bmatrix} \in \mathbb{R}^{2L\times fL}, \quad \bar{b}_{\mathbb{S}_m} := \begin{bmatrix} b_{\mathbb{S}_m^1} \\ \vdots \\ b_{\mathbb{S}_m^L} \end{bmatrix} \in \mathbb{R}^{fL}$$

It follows from the dimension of the matrix $\bar{A}_{\mathbb{S}_m}$ that the vector $\bar{z}_{\mathbb{T}}$ has $2L$ elements while the vector of the unknowns $\bar{b}_{\mathbb{S}_m}$ has $fL$ unknowns. Therefore, for the worst case when $f$ is equal to 3 (the minimal number of facets that can be used to construct a polyhedra), then we have the number of unknown variables more than the number of equations. Therefore, there exist infinitely many solutions to the equation $\bar{z}_{\mathbb{T}} = \bar{A}_{\mathbb{S}_m}\bar{b}_{\mathbb{S}_m}$. We conclude that for a number of facets $f \geq 3$, the mapping between the unknown variables $b_{\mathbb{S}_m}$ to the view of the adversary (encoded in the variables $\tilde{z}_{\mathbb{T}}$) has a non-trivial kernel. That in turn implies that, given the adversary view $V_{\mathbb{Q}_{\text{ADV}}}$, there exists infinitely many indistinguishable sensitive information $\bar{b}_{\mathbb{S}_m}$ that lead to the same values observed by the adversary. Moreover, since the result holds regardless of the number of protocol runs, we conclude that the query node reveals nothing by observing all the information from any arbitrary number of protocol runs. □

### 2.6.3 Geometric Dilution Of Precision

Geometric Dilution Of Precision (GDOP) refers to the degradation of the localization performance (measured by the error between the actual target location and the calculated one) due to geometrical arrangement amongst the range vectors between the target and the observers. This phenomenon is widely observed in many localization systems including GPS-based localization depending on the geometric positioning of GPS satellites (which represent the observers in our problem setup) compared to the actual target position [64]. Our experiments shown in Section 2.10 reveal that the proposed polyhedra-based least squares algorithm is sensitive to geometric positioning of observers.

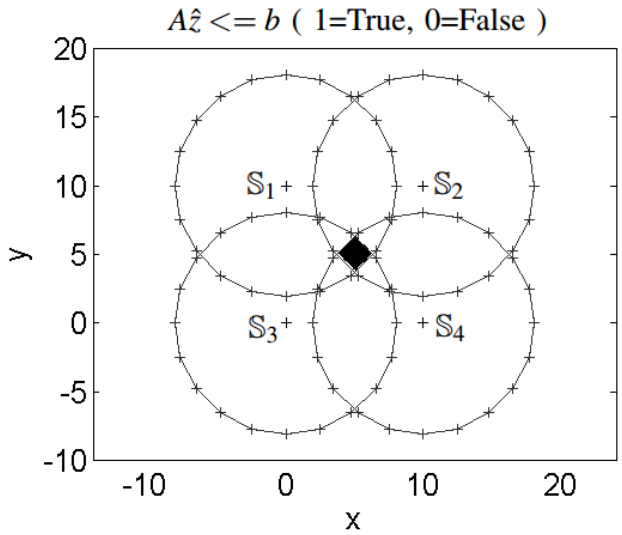
To better illustrate the sensitivity to the geometric configuration of the observers, Figures 2.8 show two cases where the locations of the observers are fixed while the target location is (i) inside the convex hull of the observers at point (5,5) and (ii) outside the convex hull of the observers at point (15,5). In each case, we plot the entire intersection polyhedron, as well as, the value of the objective function over the entire space. As shown in Figure 2.8 for the case (i), the center of the intersecting polyhedron $\mathcal{P}_{\mathbb{T}}$ lies exactly at the target location and hence the minimum of the objective function also lies at the target location. On the other hand, when the target location in case (ii) is outside the convex hull of the observers, the center of the intersecting polyhedron $\mathcal{P}_{\mathbb{T}}$ is away from the target location and also the minimum of the objective function. We give a more exhaustive experimental result showing the performance degradation in terms of localization accuracy for different observer/target configurations in Section 2.10. The objective of the algorithm discussed in the following section is to handle this degradation in performance.

## 2.7 Polyhedra-based Localization Algorithm: Alternating Projection Approach

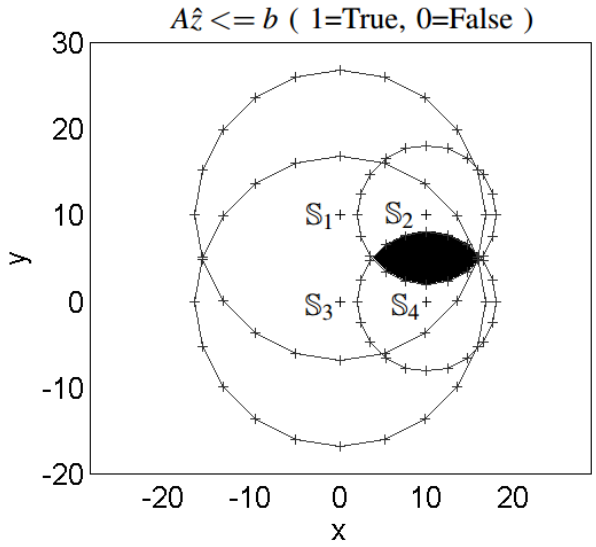
The alternating projection algorithm is a method for finding a point in the intersection of multiple convex sets (e.g. polyhedra). It was initially proposed by John von Neumann [65] for the case when the convex sets are hyperplanes and then extended by Dykstra for the general case of any convex set [66]. Given convex sets, the algorithm is guaranteed to converge to a point that minimizes the distance to all convex sets. In this section, we show how to use the alternating projection approach to reduce the degradation in performance due to the geometric dilution of precision.

### 2.7.1 Localization Using the Alternating Projection Algorithm

As discussed in the previous section and in Figure 2.8, we need to find the point that lies on the *boundary* of the intersection of all polyhedra generated by observers. Rethinking the polyhedra-based localization algorithm shown in the previous section, we can formulate the localization problem to that of finding a point that lies on the *boundary* of the intersection of all polyhedra generated by observers. Such an algorithm can be




(a)

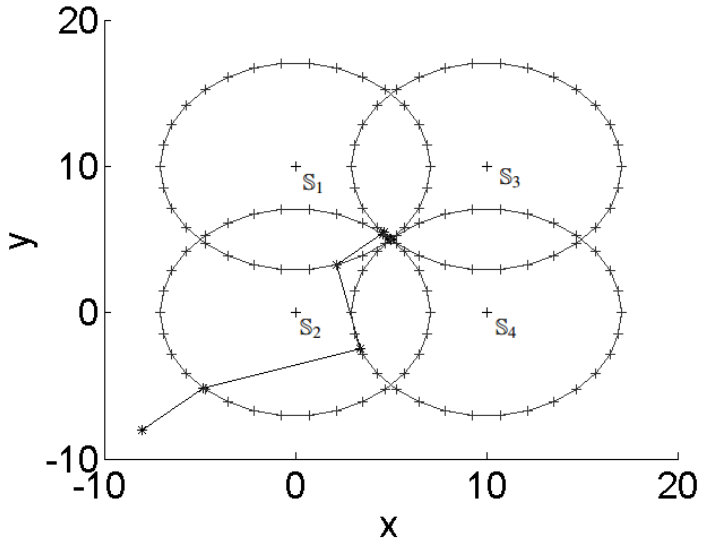



(b)

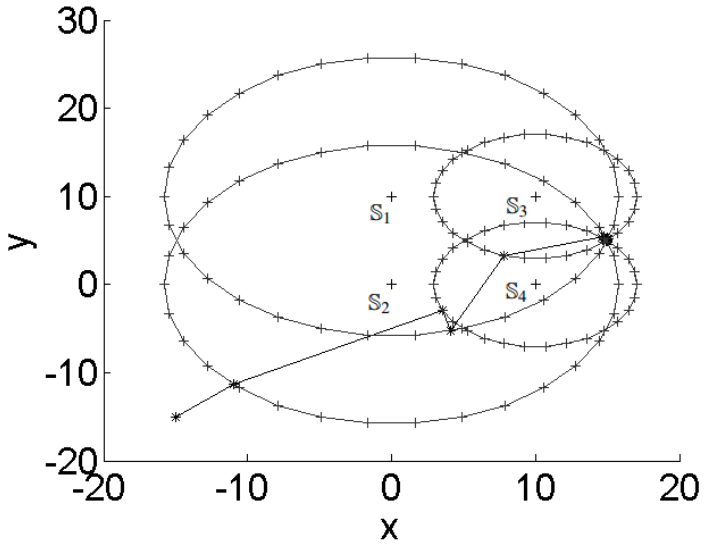
**Figure 2.8:** Different geometric configurations showing the size of the intersecting polyhedron and hence the geometric dilution of precision of the Poly-LSQ protocol. We plot two cases (a) a target located at $(5, 5)$ within the convex hull of the observers and (b) a target located at $(15, 5)$ outside the convex hull of the observers.

applied as follows. Starting from any arbitrary initial estimate, e.g. $\hat{z}_{\mathbb{T}}^{(0)} = \begin{bmatrix} 0 & 0 \end{bmatrix}^T$, we start by projecting this estimate onto the boundary of the first polyhedron $\mathcal{P}_1(A_1, b_{\mathbb{S}_1})$ generated by the first observer followed by a projection onto the boundary of the second polyhedron $\mathcal{P}_2(A_2, b_{\mathbb{S}_2})$ and so on until we project onto all $m$ polyhedra. By repeating

the previous sequence of projections until we reach the maximum number of iterations, the alternating projection algorithm guarantees that the estimate converges to the target location. An example of the sequence of points generated by this algorithm is shown in Figure 2.9 for the two cases when (i) the target is inside the convex hull of the observers and (ii) the target is outside the convex hull of the observers.




(a)




(b)

**Figure 2.9:** Several iterations of the alternating projection algorithm for different geometric configurations—a target at (5,5) inside the convex hull of the observers in subfigure (a) and at (15,5) outside the convex hull of the observers in subfigure (b)—showing the resilience of the polyhedra-based alternating projection to the geometric dilution of precision.

### 2.7.2 Projection onto Boundary of a Polyhedron

To complete the description of this algorithm, we need to show how the projection onto the boundary of a polyhedron can be computed. In this subsection, we describe how such a projection can be computed in plaintext. In the next subsection, we discuss how such an algorithm can be implemented in a privacy-preserving manner using additive homomorphic encryption. We aim to orthogonally project on the polyhedron by projecting on the nearest facet. Unfortunately, the description of the polyhedra (given by $A$ and $b$) involves hyperplanes not the facets itself. Hence (as shown in Figure 2.10) picking the *nearest* hyperplane, with the shortest distance ($R_7$ in Figure 2.10), does not correspond to picking the *nearest* facet. To sidestep this issue, we first search for the *furthest* hyperplane, with the longest distance ($R_4$ in Figure 2.10), and then pick the hyperplane on the other side of the polyhedra. Picking the hyperplane on the other side is an easy task as the polyhedra is symmetric with ordered hyperplanes $1, .., f$.

More formally, the proposed heuristic works as follows. Given a point $\hat{z}$ and a polyhedron $\mathcal{P}(A, b)$, we start by calculating the orthogonal distance between the point $\hat{z}$ and all the facets. This can be calculated as follows:

$$r_j = \frac{|a_j^T \hat{z} - b_j|}{\|a_j\|_2} \qquad a_j = A(j)^T, b_j = b(j), j \in \{1, \ldots, f\}, \tag{2.13}$$

where $A(j)$ is the jth row of matrix A. The next step is to compare the values of all the distances $r_j$ to select the furthest hyperplane which has the longest distance to $\hat{z}$. Then, we pick the hyperplane on the other side of the polyhedra which has the minimum distance and its index is denoted by $j^*$. We compute the projection of $\hat{z} = \left(\hat{z}(1), \hat{z}(2)\right)$ onto the $j^*$ facet as in (2.14). If the facet is horizontal ($a_{j^*}(1) = 0$), then the projection point $\hat{z}_\Pi$ takes $\hat{z}(1)$ as x-value and the y-value comes from the facet equation ($y = b_{j^*}/a_{j^*}(2)$). If the facet is vertical ($a_{j^*}(2) = 0$), then the projection point $\hat{z}_\Pi$ takes $\hat{z}(2)$ as y-value and the x-value comes from the facet equation ($x = b_{j^*}/a_{j^*}(1)$). The otherwise case comes from shifting the facet to the origin, preforming the projection, and then shift back to the original position.

$$\hat{z}_\Pi = \begin{cases} \begin{bmatrix} \hat{z}(1) \\ \frac{b_{j^*}}{a_{j^*}(2)} \end{bmatrix} & \text{if } a_{j^*}(1) = 0 \\ \begin{bmatrix} \frac{b_{j^*}}{a_{j^*}(1)} \\ \hat{z}(2) \end{bmatrix} & \text{if } a_{j^*}(2) = 0 \\ \frac{1}{v_{j^*} v_{j^*}^T} v_{j^*}^T v_j \begin{bmatrix} \hat{z}(1) \\ \hat{z}(2) \end{bmatrix} + \left( \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} - \frac{1}{v_{j^*} v_{j^*}^T} v_{j^*}^T v_{j^*} \right) \begin{bmatrix} 0 \\ \frac{b_{j^*}}{a_{j^*}(2)} \end{bmatrix} & \text{otherwise,} \end{cases} \tag{2.14}$$

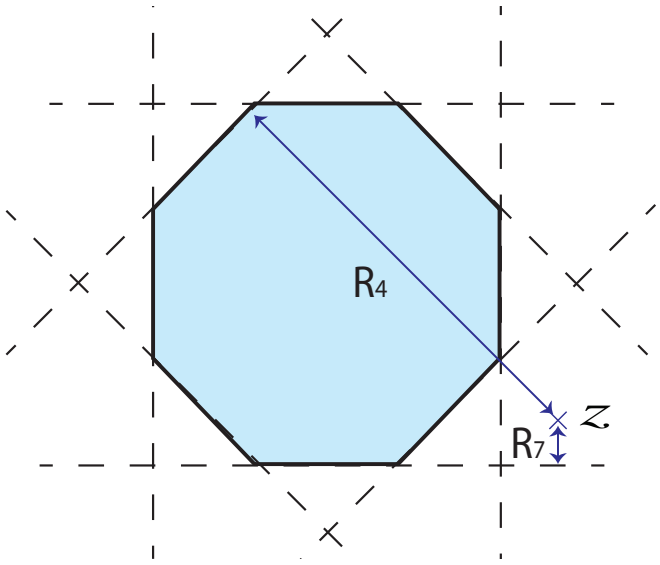



**Figure 2.10:** Projection of $z$ on the nearest and furthest hyper-planes.

where

$$v_{j^*} := \left[\frac{1}{a_{j^*}(1)} \quad \frac{-1}{a_{j^*}(2)}\right]. \tag{2.15}$$

### 2.7.3 Poly-AP: Polyhedra-Based Alternating Projection Protocol

The `Encrypt` and `Decrypt` algorithms of POLY-AP are similar to the POLY-LSQ protocol. Hence, we focus here on the `Poly-AP.Aggregate` algorithm as follows. The protocol starts with the query node $\mathbb{Q}$ generating a uniformly random initial estimate $\hat{z}_{\mathbb{T}}^{(0)}$, encrypting this estimate using its public key, and then sending $[\![\hat{z}_{\mathbb{T}}^{(0)}]\!]_{\text{pk}_{\mathbb{Q}}}$ back to the aggregator $\mathbb{A}$. It is crucial for the privacy of the POLY-AP protocol that the initial estimate is not known to the aggregator $\mathbb{A}$. Upon receiving all messages from the observers and the initial state from the query node, the aggregator performs a projection onto the boundary of the polyhedron of the first observer followed by the second observer and so forth until reaching the maximum number of iterations.

The basic component of the alternating projection algorithm is to compute the distance between $[\![\hat{z}]\!]_{\text{pk}_{\mathbb{Q}}}$ and all the facets of a given polyhedron $\mathcal{P}(A, [\![b]\!]_{\text{pk}_{\mathbb{Q}}})$, and then to perform a comparison between these distances. Thanks to the polyhedron-based representation chosen in the previous section, we can use additive homomorphism to compute the encrypted version of the distances as follows. First note that the $A$ matrix is revealed to all parties in plaintext. Hence, the observer computes the factor $\frac{1}{\|a_j\|_2}$ in plain text followed by:

$$[\![r_j]\!]_{\text{pk}_{\mathbb{Q}}} = \frac{1}{\|a_j\|_2} \otimes \left(a_j^T \otimes [\![\hat{z}]\!]_{\text{pk}_{\mathbb{Q}}} \ominus [\![b_j]\!]_{\text{pk}_{\mathbb{Q}}}\right) \tag{2.16}$$

To compute the absolute value $[\![r_j]\!]_{\text{pk}_\mathbb{Q}}$ we need to check the sign of $[\![r_j]\!]_{\text{pk}_\mathbb{Q}}$ (by comparing the value of $[\![r_j]\!]_{\text{pk}_\mathbb{Q}}$ to zero) and multiplying by $-1$ if needed. Unfortunately, additive homomorphic encryption does not preserve order, so such comparisons cannot be directly computed. Therefore we rely on the the *secure comparison* protocol (introduced in Section 2.5.2) to perform comparison between $[\![r_j]\!]_{\text{pk}_\mathbb{Q}}$ and $[\![0]\!]_{\text{pk}_\mathbb{Q}}$ as follows.

**Absolute Value Computation Protocol:** The intuition of this protocol is to use the *secure comparison* protocol to compute the absolute value of $[\![r_j]\!]_{\text{pk}_\mathbb{Q}}$. Based on a random coin flip, the aggregator issues a call to the query node using either a SECCOMPARE $([\![r_j]\!]_{\text{pk}_\mathbb{Q}}, [\![0]\!]_{\text{pk}_\mathbb{Q}})$ or a SECCOMPARE$([\![0]\!]_{\text{pk}_\mathbb{Q}}, [\![r_j]\!]_{\text{pk}_\mathbb{Q}})$ followed by sending the following encrypted values $[\![[\![r_j]\!]_{\text{pk}_\mathbb{Q}}]\!]_{\text{pk}_\mathbb{A}}$ and $[\![[\![0]\!]_{\text{pk}_\mathbb{Q}}]\!]_{\text{pk}_\mathbb{A}}$ using the same order used in the call of SECCOMPARE. Once the query node computes the index of the argument with the maximum value, instead of returning this index, it assigns the variables $maxArg$ and $minArg$ to $[\![[\![r_j]\!]_{\text{pk}_\mathbb{Q}}]\!]_{\text{pk}_\mathbb{A}}$ and $[\![[\![0]\!]_{\text{pk}_\mathbb{Q}}]\!]_{\text{pk}_\mathbb{A}}$ according to its knowledge of which argument has the maximum value. Finally, the query node returns $[\![[\![|r_j|]\!]_{\text{pk}_\mathbb{Q}}]\!]_{\text{pk}_\mathbb{A}} = maxArg \ominus minArg$ to the aggregator. Once the aggregator retrieves the absolute value from the query node, it decrypts it using its own private key and the rest of the alternating projection algorithm follows accordingly. The correctness of the *absolute value computation protocol* is trivial and follows from the fact that one of the arguments is zero and hence does not affect the value of the addition.

The next step is to compare the values of $[\![|r_j|]\!]_{\text{pk}_\mathbb{Q}}$ for all the facets $j \in \{1, \ldots, f\}$. Similarly, this can be done by relying on the *secure comparison* protocol. Once the correct facet $j^*$ is known, the final step is to compute the projection itself. Again thanks to the polyhedron representation and the fact that the $A$ matrix does not need to be encrypted, the projection $[\![\hat{z}_{\text{II}}]\!]_{\text{pk}_\mathbb{Q}}$ can be performed applying additive homomorphism to (2.14). The privacy of the proposed algorithm can be summarized as follows.

**Theorem 2.7.1.** *Assume that all entities are honest-but-curious and under standard cryptographic assumptions (namely the Decisional Composite Residuosity (DCRA) assumption), the Poly-AP localization algorithm ensures (i) observer obliviousness, (ii) non-colluding aggregator obliviousness, and (iii) query node obliviousness.*

*Proof.* **Observer obliviousness:** It follows the same argument used in the observer obliviousness of the POLY-LSQ shown in Theorem 2.6.3.

**Non-colluding Aggregator obliviousness:** Assume that the aggregator $\mathbb{A}$ is an adversarial node whose objective is to compute the sensitive sensor information $b_{\mathbb{S}_1}, \ldots, b_{\mathbb{S}_m}$. Recall that, due to the absolute value computation protocol, the aggregator does not know the sign of the intermediate results $r_j$ for each facet at each projection step. He also does not know the initial estimate $\hat{z}_\mathbb{T}^{(0)}$ or the final estimate $\hat{z}_\mathbb{T}^{(K)}$, nor is he allowed to collude with observers (due to the notion of non-colluding aggregator obliviousness). Hence his plaintext information view is:

$$V_{\mathbb{A}_{\text{ADV}}} = (A, j_1^{*,(0)}, \ldots, j_m^{*,(0)}, \ldots j_1^{*,(K)}, \ldots, j_m^{*,(K)}, \text{pk}_\mathbb{Q}, \text{pk}_\mathbb{A}, \text{sk}_\mathbb{A}).$$

where $j_s^{*,(k)}$ is the index of facet of the $s$th polyhedron on which the projection is going to take place within the $i$th iteration of the alternating projection algorithm. While again the matrix $A$ reveals no information about the sensitive sensor information, the sequence of $j_s^{*,(k)}$ indeed depends on the sensitive information. In what follows, we argue that there exist infinitely many assignments for the sensor's sensitive information which results in the same sequence of $j_s^{*,(k)}$. First, we note that the distance between a point and a hyperplane is *translational invariant*. That is, given a point $\hat{z}$ and a hyperplane $h$, which is represented by $a^T z = b$, and any arbitrary vector $\Delta z = (\Delta x, \Delta y)$, we can translate both $\hat{z}$ as $\hat{z}' = \hat{z} + \Delta z$ and $h$ as $a^T(z + \Delta z) = b + a^T \Delta z = b'$ which is denoted by $h'$. Let's define $r$ as the distance between $h$ and $z$, and $r'$ as the distance between $h'$ and $z'$. The mentioned translation preserves the following property:

$$\begin{aligned} r' &= \frac{|a^T \hat{z}' - b'|}{\|a\|_2} \\ &= \frac{|a^T(\hat{z} + \Delta z) - (b + a^T \Delta z)|}{\|a\|_2} = \frac{|a^T \hat{z} - b|}{\|a\|_2} \\ &= r \end{aligned}$$

That is, by translating both the point and the polyhedron, the distance between both of them remains constant. Since the surface of the polyhedron is defined as a set of halfspaces with corresponding hyperplanes, we conclude that given point $\hat{z}$ and a polyhedron $\mathcal{P}$, the projection operation is also *translation invariant*. That is the index of the facet for which the projection will take place does not change by performing a translation with any arbitrary vector $\Delta z = (\Delta x, \Delta y)$. Therefore, given a set of polyhedra and the sequence of indices $j_s^{*,(k)}$, and due to the fact that the aggregator does not know the initial estimate $\hat{z}_{\mathbb{T}}^{(0)}$ or the final estimate $\hat{z}_{\mathbb{T}}^{(K)}$, we conclude there exist infinitely many indistinguishable sensitive information (constructed by considering different values for the translation vector $\Delta z$) that lead to the same exact aggregator knowledge and accordingly this knowledge cannot be used to determine the sensitive information uniquely.

**Query Node obliviousness:** Assume that the query node $\mathbb{Q}$ is colluding with the observers $\mathbb{S}_1, \ldots \mathbb{S}_{m-1}$ to form an adversarial query node $\mathbb{Q}_{\text{ADV}}$ whose objective is computing the remaining sensor information $b_{\mathbb{S}_m}$. The only extra information comparing to the query node obliviousness proof in Theorem 2.6.3 is the information from the SECURECOMP protocol that the query node knows additionally the index of the argument with the maximum value. However, since the observers permutes the distance to the facets before issuing a call to SECURECOMP we conclude that the query node's knowledge about the index of the maximum value is useless.

□

### 2.7.4 SMC-Poly-AP: Alternating Projection Algorithm Based on Secure Multiparty Communication

Finally, in order to remove the dependency on the aggregator, the alternating projection algorithm described in the previous section can be easily modified to be computed in a *secure multiparty communication* fashion. Starting from observer $\mathbb{S}_1$, an arbitrary initial estimate e.g. $[\![\hat{z}_{\mathbb{T}}^{(0)}]\!]_{\text{pk}_{\mathbb{Q}}} = \begin{bmatrix}[\![0]\!]_{\text{pk}_{\mathbb{Q}}} & [\![0]\!]_{\text{pk}_{\mathbb{Q}}}\end{bmatrix}^T$ is chosen, projected onto the boundary of the polyhedron $\mathcal{P}_1(A, [\![b_{\mathbb{S}_1}]\!]_{\text{pk}_{\mathbb{Q}}})$, and then the result is sent to the second observer. The second observer projects the communicated estimate onto the boundary of the polyhedron $\mathcal{P}_2(A, [\![b_{\mathbb{S}_2}]\!]_{\text{pk}_{\mathbb{Q}}})$ and so on until we project onto all $m$ polyhedra. The last node in the ring of the communication, $\mathbb{S}_m$, sends the final estimate to the query node which decrypts it using its private key.

**Theorem 2.7.2.** *Assume that both the aggregator $\mathbb{A}$ and the query node $\mathbb{Q}$ do not collude, then under standard cryptographic assumptions (namely the Decisional Composite Residuosity (DCRA) assumption), the SMC-Poly-AP localization algorithm ensures (i) observer obliviousness and (ii) non-colluding query node obliviousness.*

The proof is omitted because it is similar to the proof of Theorem 2.7.1. Similarly to the previous algorithm, the observer obliviousness follows from the data being encrypted by the query node public key $\text{pk}_{\mathbb{Q}}$. The non-colluding query node obliviousness follows from the fact that revealing the $\overline{A}$ matrix as well as performing the secure comparison operation with the query node does not lead to any privacy leakage. Note that if the query node colludes with $m-1$ observers, then it can retrieve the intermediate projected results on the $m$ observer. Recall that these intermediate results are points that lie on the boundary of the polyhedra (since they result from the projection onto the polyhedron boundary algorithm). By obtaining many of these points, an adversarial query node can construct the polyhedron of the last observer. Hence, in Theorem 2.7.2 we rely on the weak notion of a *Non-colluding Query Node Obliviousness* which is defined as follows: A protocol satisfies non-colluding query node obliviousness if after multiple runs of the protocol the query node $\mathbb{Q}$ learns nothing about the sensitive information of the observers other than what is contained in the immutable privacy leak.

## 2.8 Resilient Privacy-Aware Localization

In the previous sections, we focused mainly on the first aspect of the problem, namely *privacy-preserving* localization. In this section, we demonstrate how the aggregator-based previously proposed algorithms can be augmented to achieve resilience as well. We base our resilience scheme on prior work on residue checking over noisy measurements [67], that was developed for the general case of secure estimation from heterogeneous sensors under false data injection attacks. In our scheme, the location error (or residue) is computed using measurements from subsets of observers and compared

against the maximum allowable residue. The subset of observers with the minimum residue is selected. A sufficient and necessary condition for this residue checking scheme to work is that the maximum number of malicious observers is $\lfloor \frac{m-3}{2} \rfloor$ [67], where $m$ is the number of all observers. While performing residue checking over subsets of observers is a combinatorial process, it was shown that using combinatorial search methods like satisfiability (SAT) solvers in conjunction with estimation methods leads to significant reduction in search time [67]. Indeed, such residue checking scheme does not work unless the localization error is bounded. Thanks to the polyhedra-based-localization algorithms discussed previously, and unlike differential privacy and other data corruption based privacy techniques, this condition follows automatically from the construction of the protocol.

To augment the polyhedra based localization algorithms with the residue checking scheme, the aggregator follows the same steps discussed before to compute the location estimate $[\![\hat{z}_{\mathbb{T}}]\!]_{\text{pk}_{\mathbb{Q}}}$ from all different subsets of $m - \lfloor \frac{m-3}{2} \rfloor$ observers. The next step is to compute the associated residual $[\![\xi]\!]_{\text{pk}_{\mathbb{Q}}} = \|\overline{A}[\![\hat{z}_{\mathbb{T}}]\!]_{\text{pk}_{\mathbb{Q}}} - [\![\bar{b}_{\mathbb{S}}]\!]_{\text{pk}_{\mathbb{Q}}}\|$ where $\|.\|$ denotes the sum of absolute values. Thanks to polyhedra-based representation, this residue can be computed using additive homographic encryption along with the absolute value protocol discussed before. Once all the residuals from all subsets of observers are calculated, the next step is to choose the minimum residue. This can be done by invoking the secure comparison protocol $\text{SECCOMPARE}([\![\xi_1]\!]_{\text{pk}_{\mathbb{Q}}}, \ldots, [\![\xi_h]\!]_{\text{pk}_{\mathbb{Q}}})$
where $h$ is the number of observer subsets. Finally, the location estimate $[\![\hat{z}_{\mathbb{T}}]\!]_{\text{pk}_{\mathbb{Q}}}$ corresponding to the subset of observers that produce the smallest residue is then sent to the query node for decryption. We call these algorithms resilient Poly-LSQ and resilient Poly-AP, respectively.

It is essential to note that by the end of the protocol, the query node learns only the location estimates produced by only one subset of observers (not all location estimates from all subsets of observers). Then, following the same algebraic construction shown in the proof of Theorem 2.6.3, we conclude that the resilience scheme described above does not leak any additional information. Based on this, we formulate the following theorem.

**Theorem 2.8.1.** *Assume that the number of malicious observers does not exceed $\lfloor \frac{m-3}{2} \rfloor$ and under standard cryptographic assumptions (namely the Decisional Composite Residuosity (DCRA) assumption), the resilient Poly-LSQ and resilient Poly-AP localization protocol produce resilient location estimates of the target while satisfying the same privacy guarantees in Theorem 2.6.3 and Theorem 2.7.1, respectively.*

The proof is based on the work [67].

## 2.9 Communication Overheads

The localization methods described in this chapter require some amount of communication between all parties involved which are the observers, the aggregator, and the

**Table 2.2:** Communication cost analysis with respect to the number of messages for private localization protocols.

| Protocol | Number of Messages |
|---|---|
| Poly-LSQ | $m+1$ |
| Poly-AP | $m\boldsymbol{I}[\boldsymbol{G}+\boldsymbol{E}]+m+1$ |
| SMC-Poly-AP | $m\boldsymbol{I}[\boldsymbol{G}+\boldsymbol{E}+1]+1$ |
| Level-II Pr [1] | $m^2+m-1$ |
| Level-III Pr [1] | $2m^2+2m-1$ |

**Table 2.3:** Communication cost analysis with respect to the message size for private localization protocols.

| Protocol | Message size (bits) |
|---|---|
| Poly-LSQ | $[2\boldsymbol{P}\boldsymbol{S}+\boldsymbol{C}\boldsymbol{S}]m+2\boldsymbol{C}$ |
| Poly-AP | $[2\boldsymbol{P}+\boldsymbol{C}]m\boldsymbol{S}+2\boldsymbol{C}+m\boldsymbol{I}\boldsymbol{C}(\boldsymbol{G}+\boldsymbol{E}\boldsymbol{S})$ |
| SMC-Poly-AP | $2m\boldsymbol{I}\boldsymbol{C}\ +\ 2\boldsymbol{C}+m\boldsymbol{I}\boldsymbol{C}(\boldsymbol{G}+\boldsymbol{E}\boldsymbol{S})$ |
| Level-II Pr [1] | $[5m^2+4m-8]\boldsymbol{P}$ |
| Level-III Pr [1] | $3m\boldsymbol{C}+[3m^2-m-1]\boldsymbol{P}$ |

**Table 2.4:** Detailed communication cost analysis for Poly-LSQ (left) Poly-AP (right) protocol with respect to number of messages between different entities.

| | Destination | | |
|---|---|---|---|
| **Source** | $\mathbb{S}$ | $\mathbb{A}$ | $\mathbb{Q}$ |
| $\mathbb{S}$ | 0 | $m$ | 0 |
| $\mathbb{A}$ | 0 | - | 1 |
| $\mathbb{Q}$ | 0 | 0 | - |

| | Destination | | |
|---|---|---|---|
| **Source** | $\mathbb{S}$ | $\mathbb{A}$ | $\mathbb{Q}$ |
| $\mathbb{S}$ | 0 | $m$ | 0 |
| $\mathbb{A}$ | 0 | - | $m\boldsymbol{I}[\boldsymbol{G}+\boldsymbol{E}]/2+1$ |
| $\mathbb{Q}$ | 0 | $m\boldsymbol{I}[\boldsymbol{G}+\boldsymbol{E}]/2$ | - |

**Table 2.5:** Detailed communication cost analysis for SMC-Poly-AP protocol with respect to number of messages between different entities.

| | Destination | |
|---|---|---|
| **Source** | $\mathbb{S}$ | $\mathbb{Q}$ |
| $\mathbb{S}$ | $m\boldsymbol{I}$ | $m\boldsymbol{I}[\boldsymbol{G}+\boldsymbol{E}]/2+1$ |
| $\mathbb{Q}$ | $m\boldsymbol{I}[\boldsymbol{G}+\boldsymbol{E}]/2$ | $-$ |

query nodes. This communication comes at a cost, as the observers may be resource-constrained devices and the communication links may be with low bandwidth. Thus, an efficient localization algorithm must be one that reduces communication overheads. We

**Table 2.6:** Detailed communication cost analysis for Level-II Pr (left) and Level-III Pr (right) protocols [1] with respect to number of messages between different entities.

| | **Destination** | |
|---|---|---|
| **Source** | $\mathbb{S}$ | $\mathbb{Q}$ |
| $\mathbb{S}$ | $m^2-1$ | $m$ |
| $\mathbb{Q}$ | $0$ | $-$ |

| | **Destination** | |
|---|---|---|
| **Source** | $\mathbb{S}$ | $\mathbb{Q}$ |
| $\mathbb{S}$ | $2m^2-m-1$ | $2m$ |
| $\mathbb{Q}$ | $m$ | $-$ |

denote the number of bits required to represent Plaintext and PHE Ciphtertext by $\boldsymbol{P}$ and $\boldsymbol{C}$, respectively. Additionally, we denote the number of samples used to discretize range estimate *circles* by $\boldsymbol{S}$, algorithm iterations by $\boldsymbol{I}$, the number of messages used in secure comparison by $\boldsymbol{E}$, and the messages in $\arg\max$ over an encrypted vector by $\boldsymbol{G}$. The cost associated with communication must necessarily incorporate both the length (bits) of each message, the total number of messages, and the source and destination of each message as some parties such as the processing server may have more computation and communication resources than other.

We note that the presented communication overheads can be reduced, if we do not communicate the $A$ matrix and assume it is fixed in the algorithms. In order to provide a baseline against which to compare the communication overheads of the methods presented in this chapter, we compare against the two algorithms proposed by Shu et al. for hiding the location of observers or *anchors* [1]. These algorithms—called *Level-II* and *Level-III* by the authors—demonstrate an improvement with respect to prior efforts based on the concept of oblivious transfers (OT) [68] and homomorphic encryption. Table 2.2 summarizes the required number of messages in our proposed protocols in comparison to the proposed protocols in [1]. This comparison assumes that localization occurs in $\mathbb{R}^2$. Table 2.3 shows the message size of the transmitted messages in Table 2.2.

Every observer $\mathbb{S}$ in both the Poly-LSQ and Poly-AP protocols sends one message to the aggregator $\mathbb{A}$. The aggregator $\mathbb{A}$ then prepares and sends the encrypted target location to the Query node $\mathbb{Q}$. Here, the Poly-AP protocol requires $\boldsymbol{G}+\boldsymbol{E}$ messages for each iteration to find the minimum distance at each observer. Thus, the Poly-LSQ and Poly-AP protocols require $m+1$ and $m\boldsymbol{I}[\boldsymbol{G}+\boldsymbol{E}]+m+1$ messages, respectively. The size of the message sent by an observer to the Aggregator is calculated as $2\boldsymbol{PS}+\boldsymbol{CS}$ bits, which is largely dominated by the size of plaintext matrix $A$ and encrypted matrix $B$. Furthermore, the final encrypted location sent to the querier is $2\boldsymbol{C}$ bits, containing the encrypted 2D localization result. As a comparison, SMC-Poly-AP requires additional communication cost in order to send the projected point from one observer to the next. On the other hand, SMC-Poly-AP does not need to send the $A$ or $B$ matrices to the aggregator. As shown in Table 2.2, the Poly-LSQ and Poly-AP protocols perform the best in terms of message counts, with linear dependencies on the number of observers $m$. The SMC-Poly-AP protocol is only marginally worse than Poly-AP in this regard, while both algorithms from [1] have quadratic dependencies on $m$. In analyzing the cost of each

transmission in bits in Table 2.3, we see that again those in [1] have quadratic dependencies on $m$ while Poly-LSQ, Poly-AP, and SMC-Poly-AP have linear dependencies with slight differences. However, Level-II Pr has an advantage in terms of the message size, as it avoids encrypted data.

In addition to the number and size of each message, it is important to consider the source and destination of each message, as the different parties involved usually have different computation and communication resources. The messages sent between each party in Poly-LSQ, Poly-AP, and SMC-Poly-AP are summarized in Tables 2.4 and 2.5. The results show that there is very little communication required from the observers in either protocol—no more in fact than the minimum $m$ observer messages. The messages sent between each party in Level-II and Level-III protocols are summarized in Table 2.6, where there is no aggregator $\mathbb{A}$. In comparison, the number of messages sent by each observer in these two protocols is more than twice that of either Poly-LSQ or Poly-AP, and quadratic in the worst case (Level-II Pr and Level-III Pr). This is significant, as localization systems in practice have many observers in order to reduce the effects of measurement noise, and oftentimes these observers are battery powered devices with constrained communication and computation resources.

## 2.10 Evaluation

In this section, we evaluate the PrOLoc localization protocols described in the previous sections. We begin with a case study for localization using FHE, followed by a numerical analysis of the localization performance for each protocol. Then, we present case studies including an end-to-end experiment using real custom range measurement hardware.

### 2.10.1 Localization using Fully Homomorphic Encryption

In our first case study, we demonstrate the high cost of performing the traditional least squares in Section 2.2 using FHE (specifically the leveled FHE library HELib[3]) and the real need for our localization reformalization. We basically run the gradient descent in (2.3) over encrypted numbers. Leveled FHE schemes create an *a priori* arithmetic circuit of depth $L$. Increasing the number of gradient descent iterations requires a deeper circuit which increases the total execution time with the benefit of decreasing localization error. Additionally, increasing the number of iterations over scaled values requires wider bit lengths, causing HElib to fail in some instances. Thus, we provide here measurements of the execution time for binary multiplication and addition in order to estimate a reasonable bound on HElib's total execution time.

We used the Karatsuba algorithm to calculate the number of binary multiplication and addition operations in each n-digit multiplication [69]. The execution time of primitive binary encryption, decryption, multiplication, and addition is measured on a Macbook Pro with an Intel Core i7-4870HQ CPU @2.5GHz using a range of circuit depth levels.

[3]https://github.com/shaih/HElib

**Table 2.7:** Analysis of secure least squares localization using Leveled FHE.

| iteration | L | Mul [sec] | add [sec] | Enc. [sec] | Dec. [sec] | size [bytes] | Exec. Time[min] | Loc. error [m] |
|---|---|---|---|---|---|---|---|---|
| 1 | 27 | 0.2480 | 0.0025 | 0.1440 | 0.0854 | 9 | 4.10 | 3.172 |
| 2 | 36 | 0.5059 | 0.0052 | 0.3656 | 0.1593 | 14 | 16.745 | 2.115 |
| 3 | 45 | 0.7065 | 0.0085 | 0.4965 | 0.2111 | 21 | 44.278 | 1.410 |
| 4 | 54 | 0.8771 | 0.0113 | 0.5447 | 0.3082 | 27 | 81.677 | 0.940 |
| 5 | 63 | 1.1742 | 0.0232 | 0.7826 | 0.3845 | 33 | 195.52 | 0.626 |
| 6 | 72 | 2.2802 | 0.0233 | 1.6437 | 1.0399 | 40 | 480.06 | 0.4178 |
| 7 | 81 | 2.7026 | 0.0321 | 1.8588 | 1.0549 | 46 | 703.12 | 0.2785 |
| 8 | 90 | 2.7550 | 0.0367 | 2.0965 | 1.0945 | 52 | 865.10 | 0.0018 |
| 9 | 99 | 3.2930 | 0.0514 | 2.2922 | 1.1994 | 59 | 1260.33 | 0.0012 |

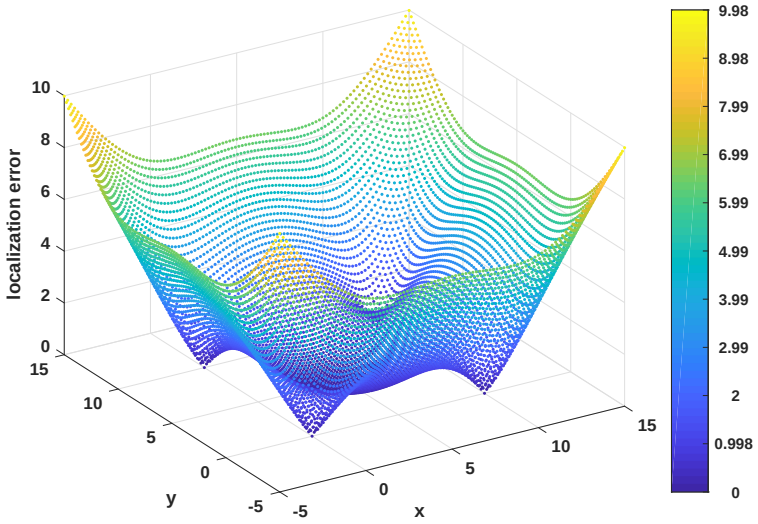



**Figure 2.11:** Localization error of Poly-LSQ protocol for a moving target with ideal range measurements.

Table 2.7 summarizes the results for each gradient descent iteration. The reported size is the number of bytes for each message. The total execution time is calculated based on the number of required multiplications in the secure least squares. We set HElib's security level to 80, resulting in a 1024-bit asymmetric key, and used a circuit depth of 27 for all iterations. As shown in Table 2.7, a single iteration of the FHE protocol results in 3.17m error and an execution time of 4.1 minutes. After 7 iterations, the error has reduced to a modest 27 cm error, but this comes at the cost of 703 minutes (=11.7 hours) of computation time. From this analysis, we see that the traditional localization using FHE is not practical option.

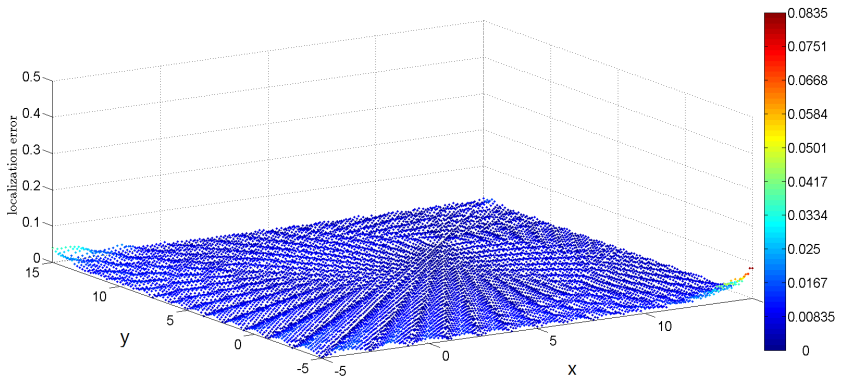



**Figure 2.12:** Localization error of Poly-AP protocol for a moving target with ideal range measurements.

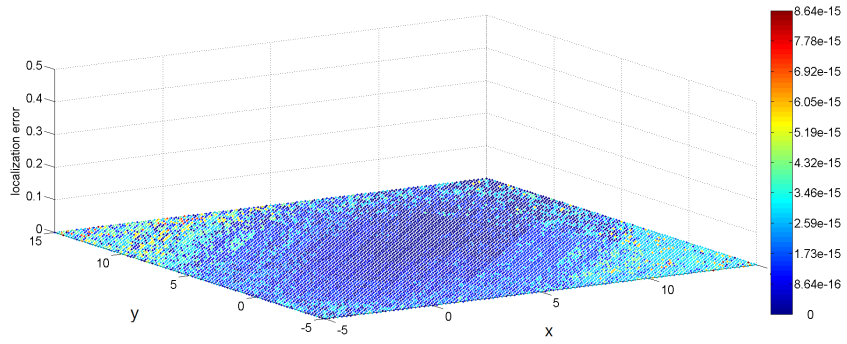


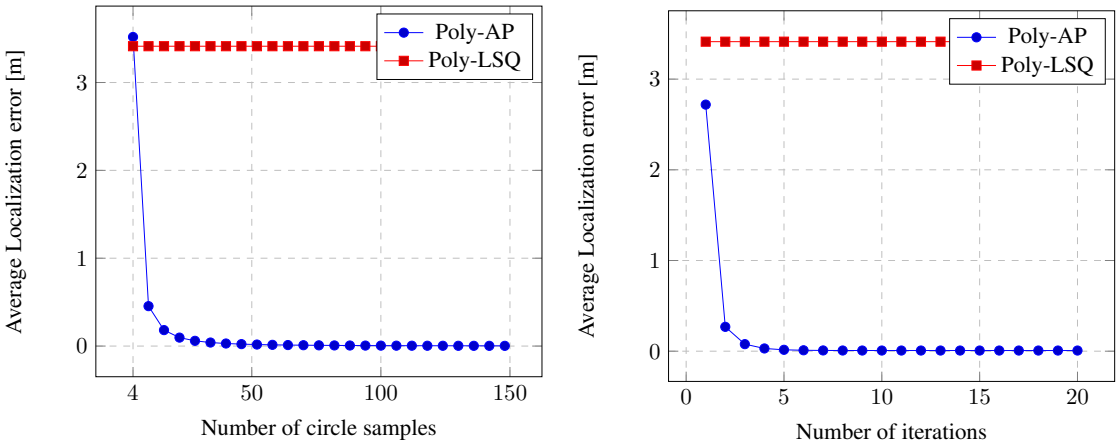

**Figure 2.13:** Localization error of unsecure traditional least squares algorithm for a moving target with ideal range measurements.

## 2.10.2 Numerical Analysis

Next, we analyze our proposed protocols. We begin with a simulated 400m$^2$ room with observers at four points $(x_{\mathbb{S}_i}, y_{\mathbb{S}_i}) \in \{(0,0),(0,10),(10,0),(10,10)\}$. At any point in time, the target may exist at any point $(x_{\mathbb{T}}, y_{\mathbb{T}})$, $-5 \leq x_{\mathbb{T}} \leq 15$, $-5 \leq y_{\mathbb{T}} \leq 15$. For each target location, we calculate a position estimate using the previously described secured and unsecured algorithms. While Poly-LSQ produces low estimation error within the convex hull of the observers, Figure 2.11 shows it suffers from high localization errors outside convex hull of the observers. The Poly-AP and SMC-Poly-AP protocols, on the other hand, produce very low error estimation errors within and outside the convex hull of the observers, as illustrated in Figure 2.12. Note that we have omitted results from the SMC-Poly-AP protocol, as it mirrors those from Poly-AP due to the underlying alternating projection algorithms. The unsecured localization in Figure 2.13 which is described in Section 2.2 serves as a baseline against which to compare the secured Poly-LSQ, Poly-AP, and SMC-Poly-AP protocols.

Additionally, we performed a series of tests in order to investigate the effect of the number of circle samples and iterations on the localization accuracy of the Poly-AP and Poly-LSQ protocols. The average absolute localization error is calculated over all points on a grid using a varying number of circle samples to construct the polyhedra. Figure 2.14a shows the effect of increasing the number of samples for both the Poly-AP and Poly-LSQ protocols. While Poly-LSQ is not affected by increased polyhedra edges beyond 4, Poly-AP benefits greatly from an increased number of samples on the range circle, converging after 10-20 samples. We performed a similar analysis in order to show the effect that the number of iterations has on the proposed protocols. Figure 2.14b shows that increasing the number of iterations has a greater effect on the Poly-AP protocol. The Poly-LSQ protocol shows a constant localization error across multiple iterations, while again the Poly-AP alternating projection protocol shows a convergence with increased number of iterations, greatly outperforming the Poly-LSQ algorithm. Here, Poly-AP has converged after around 5 iterations.

To investigate the effect that range estimation noise has on each algorithm, we add normally distributed random noise. The results of these tests are shown in Figure 2.15, where the average localization error is plotted against the standard deviation of the estimation noise. Here we can see that the Poly-AP and unsecure protocols show comparable results around a standard deviation of 4m. For very high estimation noise, Poly-LSQ outperforms both Poly-AP and unsecure protocols, as long as the target remains within the convex hull formed by the observers. Additionally, Poly-AP outperforms unsecure in terms of noise rejection with superior localization accuracy over Poly-LSQ.




**(a)** The effects of the number of range circle 'samples' on the localization error.

**(b)** The effect of the number of algorithm iterations on the localization error.

**Figure 2.14:** Performance analyses for polyhedron-based localization

## 2.10.3 End-to-end Analysis on Real Hardware

In this section, we report the experimental results for a end-to-end analysis of the proposed protocols. We implemented the Paillier cryptosystem and the Java BigInteger li-

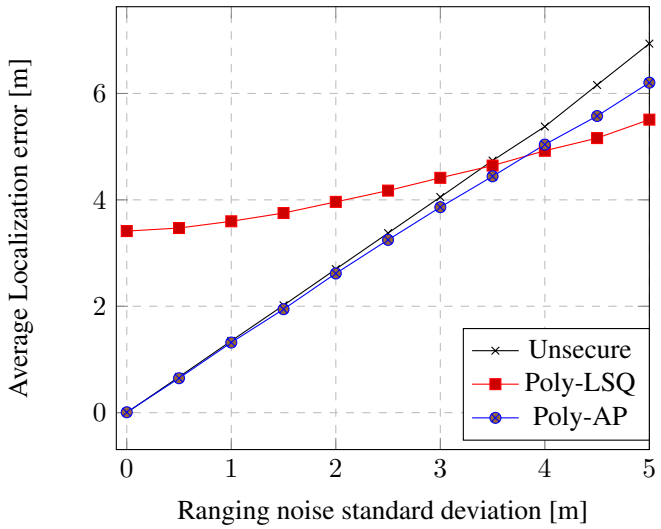



**Figure 2.15:** The effect of range estimation noise on localization error.

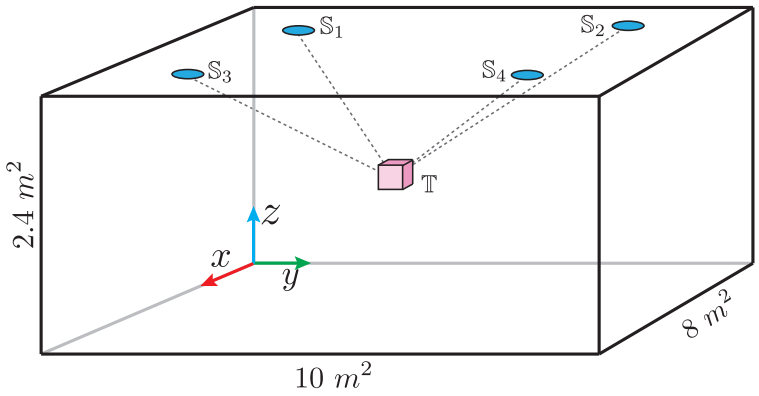



**Figure 2.16:** Ranging test bed configuration with four anchor nodes and one target node.

brary with full floating point support as described in Section 2.5.3. Both the Aggregator and Querier are implemented on a Macbook Pro laptop with Intel(R) Core i7-4870HQ CPUs @ 2.5GHz. We validated our proposed secure localization method using custom ranging hardware in an $8 \times 10 \times 2.4\ m^2$ room with four anchor (observer) nodes capable of symmetric double-sided time of flight range measurements and one mobile (target) node of unknown location as illustrated in Figure 2.16. The target node was constrained to motion on a fixed $z$ plane, allowing the problem to be reduced to localization in $\mathbb{R}^2$. Although the intent of this small-scale experiment is to evaluate the computation and localization performance of our protocols, we note that security concerns can arise even indoors, where malicious users may attempt to tamper with, destroy, or steal sensing hardware used for localization. The details of the ranging hardware for both anchor and target are described next.

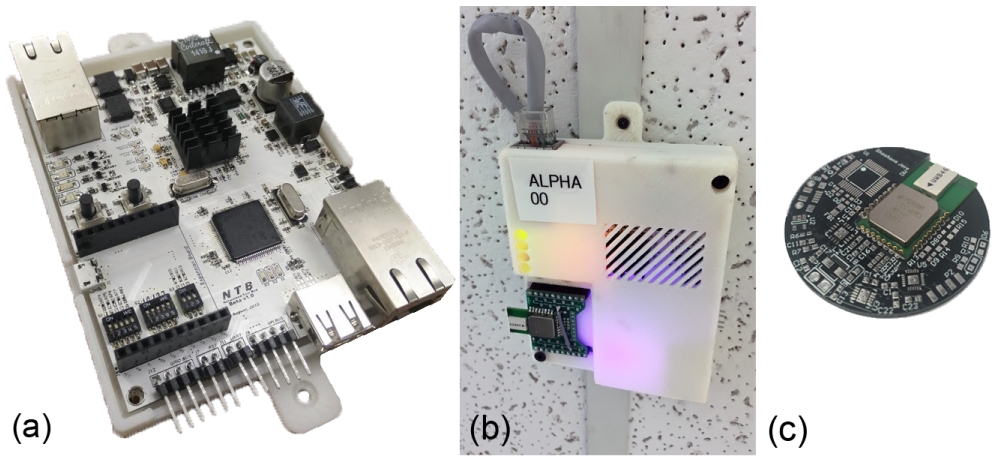



**Figure 2.17:** (a) Custom ranging anchor circuit board, (b) ceiling-mounted anchor node, and (c) mobile ranging target.

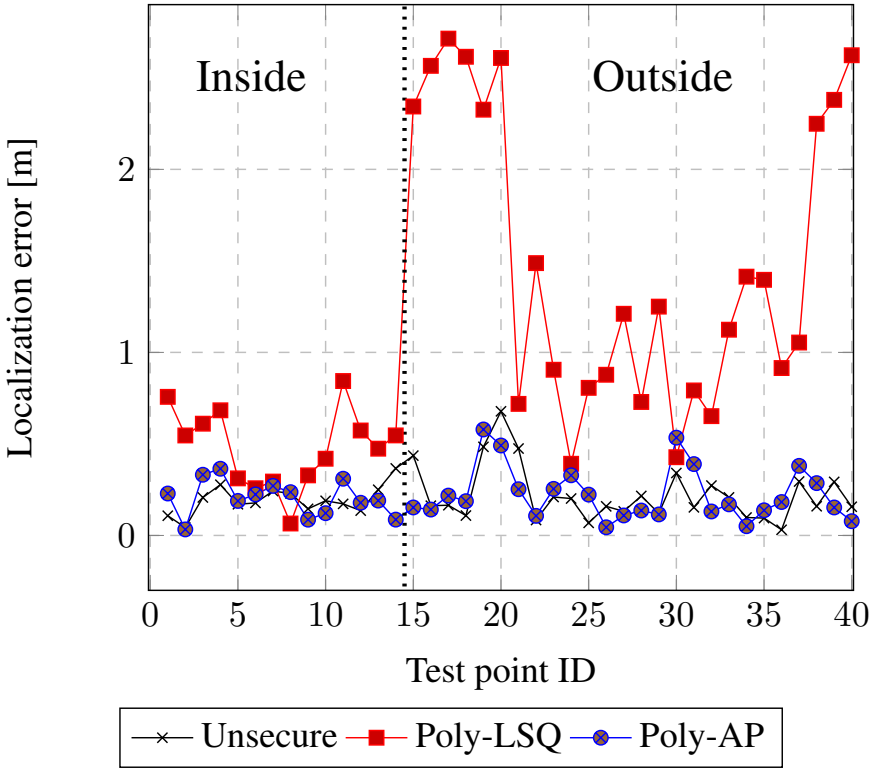



**Figure 2.18:** Localization error of Poly-LSQ and Poly-AP protocols. The dotted line separates the points inside and outside the observers' convex hull.

**Anchor Ranging Hardware:** The anchor nodes used in the following experiments consist of custom-built circuit boards equipped with ARM Cortex M4 processors at 196 MHz communicating to Decawave DW1000 ultra-wide band radios as shown in Figure 2.17. Each anchor node listens for incoming range messages from a target node and,

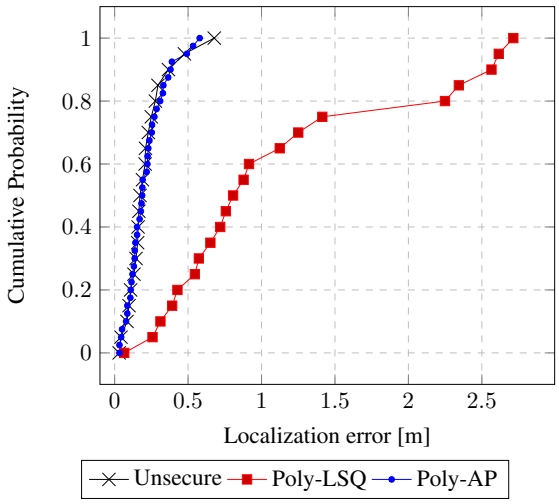
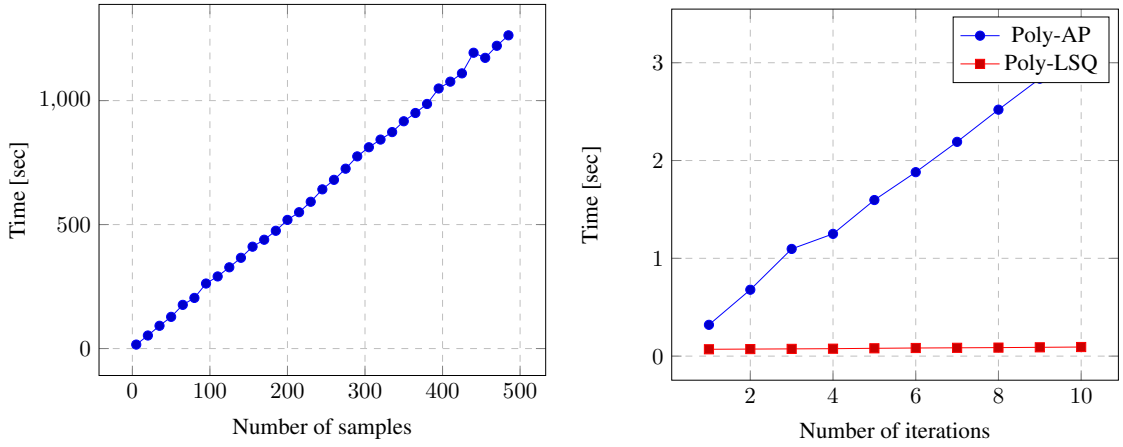



**Figure 2.19:** Cumulative probability of the localization error.




**(a)** Number of range circle samples versus the execution time on Nexus 5.

**(b)** Number of iteration versus the execution time where the number of samples is fixed at 80 samples.

**Figure 2.20:** Execution time analysis.

upon reception, begins a symmetric double-sided ranging sequence. After completing a range sequence, each anchor communicates the range estimate to an Android Nexus 5 smartphone to perform the secure communication to the Aggregator localization server. In effect, the Nexus 5 and custom ranging hardware serve together as a single cohesive observer node.

**Mobile Target Ranging Hardware:** The mobile target consists of battery-powered ARM Cortex M0+ processors with the same DW1000 ultra-wide band radios which are the same in the anchor. This allows for compatibility in the double-sided ranging technique used. It is important to note that the transmit power of the anchors is not known by external parties *a priori*, so that the target nodes or any other eavesdropping node cannot determine precisely where the anchor nodes are located. Furthermore, the anchor nodes

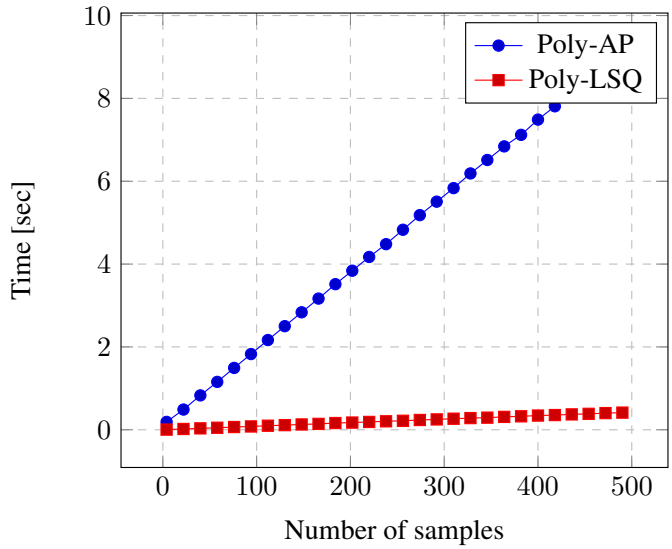



**Figure 2.21:** Number of range circle samples versus Aggregator execution time where the number of iterations is fixed at 5.

are capable of changing their transmit power dynamically without detrimental effects on ranging estimates, further obfuscating their location.

#### 2.10.3.1 Localization error

We evaluated $40$ test points in the $8 \times 10$ m room using each of the described localization protocols. The number of samples on each range *circle* is 80 samples to construct the polyhedra used in Poly-LSQ and Poly-AP, and the number of iterations for the Poly-AP protocol is fixed at $4$. Figure 2.18 shows the localization errors at each test point, where point IDs $\{1, \ldots, 14\}$ are within the polytope formed by the edges connecting each anchor, and point IDs $\{15, \ldots, 40\}$ are outside. The localization error is calculated as the mean squared error between the location estimate and the ground truth location. As seen in Figure 2.18, The Poly-LSQ protocol shows the highest error, with excessive errors when the target is outside the polytope of the anchors. The Poly-AP protocol provides good localization results compared to the traditional unsecure protocol. Table 2.8 summarizes the average and standard deviation of the localization errors for the three protocols, and Figure 2.19 shows the cumulative distribution functions for the location errors. On average, the Poly-LSQ protocol achieves around $1.13$m mean error while both Poly-AP (and equivalently SMC-Poly-AP) and unsecure achieve an average error of around $23$cm. Most importantly, the Poly-AP algorithm closely matches the performance of the traditional unsecure protocol. This lends credence to partial homomorphism as a viable method for privatizing observer data when performing high accuracy localization.

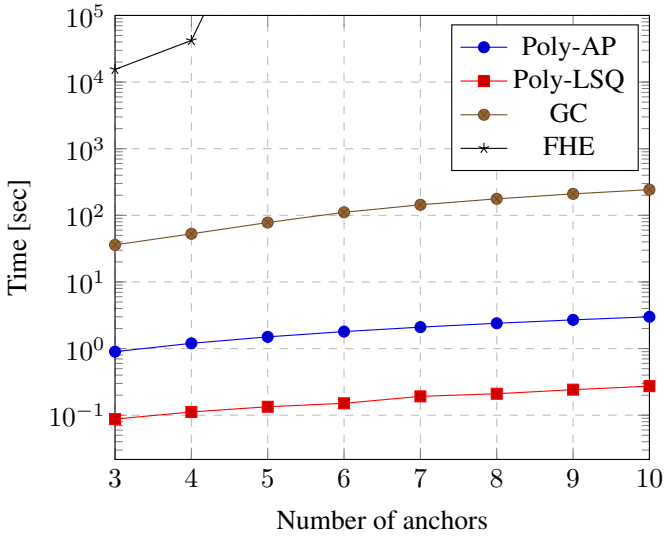



**Figure 2.22:** Aggregator execution time (semi-log scale) vs. number of anchors.

**Table 2.8:** Localization error comparison (m).

| | **mean** | **standard deviation** |
|---|---|---|
| unsecure | 0.2341 | 0.18738 |
| Poly-AP | 0.2381 | 0.18634 |
| Poly-LSQ | 1.1309 | 0.80183 |

### 2.10.3.2 Execution Time

Total execution time is determined by several different factors. At the observer side, execution is dominated by preparing the $A$ and encrypted $b$ matrices. The execution time on a Nexus 5 smartphone is shown in Figure 2.20a, as a function of number of circle samples. For 80 samples, as used in the above experiments, execution time on a Nexus 5 is below 250ms. The number of alternating projection iterations also has a great effect on the overall execution time at the aggregator side. Figure 2.20b shows that, while Poly-LSQ has a constant execution time of around 100ms on the aggregator, Poly-AP protocol has a linear dependency on number of iterations, with 4 iterations requiring around 1200ms on the aggregator. Finally, an increased number of samples of the ranging circle also causes an increased execution time on the Aggregator, with Poly-LSQ protocol showing an execution time of under 100ms for 80 samples while Poly-AP protocol shows an execution time of around 1800ms for 80 samples. While the Poly-AP algorithm requires considerably more computation than both Poly-LSQ and unsecure protocols, it can still be executed in real time on commodity hardware. Additionally, for the settings used in these experiments—80 polyhedra facets and 4 alternating projection iterations—the Poly-AP algorithm completes each localization round in just 1.25 seconds on the aggregator—the bottleneck of the system.

We compare the execution time of the proposed protocols against two secure function evaluation schemes, namely (1) Fully Homomorphic Encryption [41] and (2) Garbled Circuits (implemented using the TASTY tool [43]). As shown in Figure 2.22, the proposed Poly-LSQ and Poly-AP outperform the other algorithms by at least an order of magnitude. Moreover, and as expected, the FHE implementation is far away from being practical as it requires more than 4.3 hours to calculate the estimate of the target when measurements are available from only 3 observers. The Garbled Circuits scales better compared to FHE, however even for a small number of observers it requires multiple minutes of computation (on both observers and aggregator) to compute the target position. On the other hand, the execution time of the proposed Poly-LSQ is in the range of hundreds of milliseconds. These results show how our proposed protocols are better suited to real-time implementations of privacy-aware localization systems.

## 2.11 Conclusions

To conclude this chapter, we have presented PrOLoc, a set of protocols and algorithms designed to perform accurate localization in a manner that preserves the privacy of the observers whose measurements are used to calculate the location estimate considering a set of those observers can be maliciously injecting false data. We have built PrOLoc around the Pallier additive homomorphic cryptosystem, redesigning traditional localization algorithms to benefit from the privacy guarantees of partial homomorphism. Our experiments over both simulated and real, custom range measurement hardware demonstrate that PrOLoc can accurately and efficiently provide localization estimates comparable to traditional, unsecured methods. Specifically, we have provided results indicating that PrOLoc can yield location estimates that are comparable to state-of-art unsecured methods while operating in real-time. Also, our experiments on real hardware demonstrate that PrOLoc yields accurate location estimate at least $500x$ faster than state-of-art secure function evaluation techniques. Finally, we have provided strong theoretical privacy and resilience guarantees for the proposed protocols.

# 3 Distributed Simultaneous Localization and Time Synchronization

It is important to coordinate timing among networked devices and to provide contextual information, such as location, with a dramatic increase in the number of wireless devices. Maintaining a shared notion of time is critical to the performance of many cyber-physical systems (CPS) such as wireless sensor networks, Big Science [70], swarm robotics [71], high frequency trading [72], telesurgery [73], and global-scale databases [74]. In addition, position estimation is necessary for different fields such as military [2], indoor and outdoor localization [3].

Seeking a solution for time synchronization provides information about the localization since time and distance are causally associated with each other. In many cases, accurate time synchronization among nodes is necessary for their precise localization, as is the case in techniques based on time-of-arrival (TOA) [75] and time-difference-of-arrival (TDOA) [76] measurements. Moreover, a combined solution to time synchronization and localization problems in sensor networks could be obtained with less computational effort by tackling the two problems in a unified approach, instead of considering them as two separate problems. In order to perform simultaneous time synchronization and localization using centralized algorithms, all nodes must send their measurements to a fusion center that computes the estimates of position and clock parameters for every node. The information is then sent back to every node and the process is repeated. This strategy requires a large communication overhead, might not be energy efficient, and has a potentially critical failure point at the fusion center. This motivates proposing the schemes in this chapter, where clock models and device locations are solved concurrently in a distributed fashion. This joint optimization provides a principled way of extracting range estimates from time synchronization messages, as well as using range measurements to refine the time synchronization process based on precise models that relate timing variations, channel propagation delays, and both motion and clock dynamics.

This chapter presents D-SLATS, a framework that is comprised of three different, independent, and distributed algorithms to achieve network time synchronization and accurate position estimates using time stamped message exchanges, filters and optimization techniques. DKAL is the first proposed algorithm which is based mainly on the distributed Extended Kalman Filter (EKF) with diffusion between wireless nodes. DKAL stands for distributed Kalman filter for localization. DKAL improves some aspects of the algorithm in [77] by decreasing some of the computational efforts. The second al-

gorithm, DKALarge is based on the results from [78] and deals with large scale systems where DKAL might not be the best option. DKALarge stands for distributed Kalman filter for localization of large scale systems. It is important to note that the modifications proposed in our work to the algorithms presented in [78] and [77], can be used in the estimation in general, not only localization. Finally, we present DOPT, an optimization technique to synchronize and localize the wireless nodes in a distributed fashion. DOPT stands for distributed optimization algorithm for localization. Since D-SLATS operates in a distributed fashion, it does not require a fusion center and the results from the three algorithms can work perfectly with different network topologies. The three algorithms proposed are evaluated on custom ultra-wideband wireless hardware for networks with different topologies containing both static and mobile nodes. This chapter leverages ultra-wideband RF communication to make precise timing measurements. While UWB has improved non-line-of-sight performance in comparison to signal strength methods like those based on Received Signal Strength Indication (RSSI), it should be noted that UWB timing accuracy can deteriorate with increased environmental clutter and signal attenuation, as described in [79]. We compare the proposed algorithms with a conventional centralized EKF, and we present the accuracy of the node position estimation, as well as, the time synchronization.

This chapter is based on our publications in [21, 22]. The rest of this chapter is organized as follows: We review the ranging techniques for the localization process in Section 3.1. Related work is shown in Section 3.2. Section 3.3 provides an overall overview about the system model. We then go through D-SLATS algorithms in Section 3.4. Then, Section 3.5 evaluates the introduced algorithms on static and mobile network of nodes. Finally, Section 3.6 concludes this chapter.

## 3.1 Ranging Techniques for Localization

Many localization schemes rely on methods for predicting ranges between two devices. These ranges then serve as constraints in an estimation routine such as geometric trilateration, least squares regression, bayesian likelihood, etc. The generation of these range measurements—say between an anchor whose location is traditionally known and a target whose position is normally unknown—may be done in a variety of ways depending on the application requirements. In general, these ranging techniques can be largely divided into three categories based on the level of cooperation between the ranging party and the ranged target. These categories are described below for several exemplary ranging modalities as shown in Figure 3.1.

### 3.1.1 Beaconing methods

The simplest ranging technique involves one party transmitting *beacons* periodically while the other party observes these beacons. This technique includes (i) optical and acoustic signal-strength-based methods, (ii) power-based RF ranging requires where the RSSI of beacon messages are used in conjunction with knowledge of the transmit power

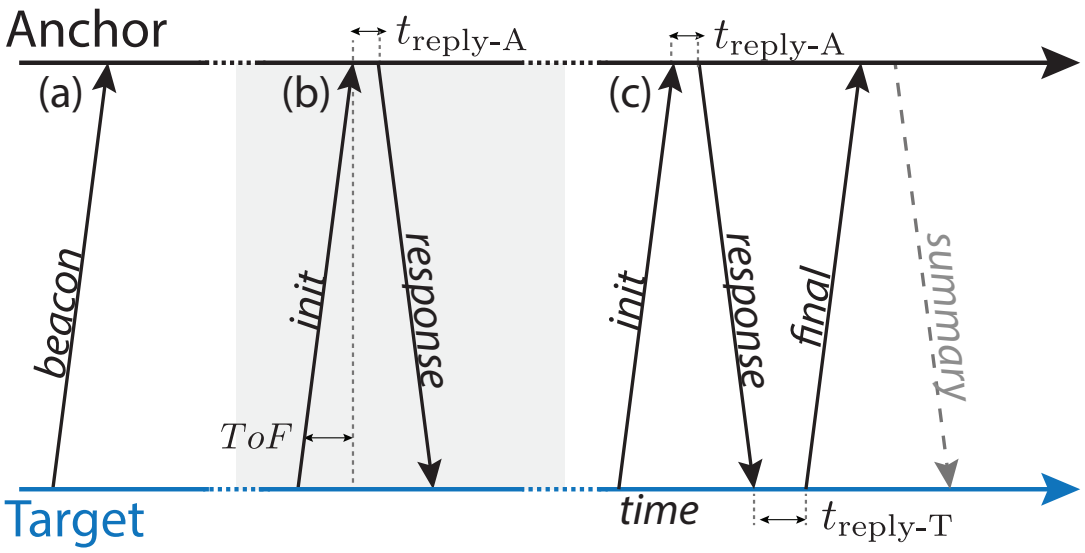



**Figure 3.1:** Two-party ranging techniques, including (a) beacon-based ranging, (b) one-way time-of-flight (ToF) ranging, and (c) symmetric double-sided ToF ranging.

to infer the distance over which the signal must have traveled, and (iii) TOA, TDOA, and angle of arrival (AOA) techniques in which beacons from multiple sources are analyzed jointly along with knowledge of the beacon source locations to constrain a target's position in space. In these types of ranging, only the receiving party (the anchors in our example or oftentimes the target in TOA, TDOA, and AOA methods) are able to estimate the range. Additionally, many of these techniques suffer from multi-path and signal fading in cluttered environments, causing high nonlinearities and high errors.

### 3.1.2 Single-sided Time of Flight Ranging

Many signals can also be used to estimate distance by measuring signal propagation time and comparing it against the known speed of a given medium—light, sound, etc. In this scenario, one party sends a signal that is then reflected by the second party. For example, an ultrasonic chirp or laser pulse can be transmitted and reflected by a physical object or an RF signal can be sent and then received by a party and retransmitted, effectively reflecting it to the source. This often requires precise timing hardware, and in the case of RF signals this inherently requires participation by both the target and the anchor. Specifically, one party (e.g. the target) will send a range 'init' message to the other party (e.g. the anchor) who must then process the message with some delay $t_{\text{reply-A}}$ and reply with a range 'response' message. The final round trip time is then measured and used to infer the propagation delay of the signal and consequently the distance between the two parties. For signals which are physically reflected such as acoustics and optics, the sequence is the same but $t_{\text{reply-A}}$ is, for all practical purposes, zero. In this scenario, only the party who initiated the ranging sequence (the target in our example) receives

the precise time of flight measurement, while the other party (e.g. the anchor) can only make coarse range estimates based on the power of the received signal as in the beaconing methods described above.

### 3.1.3 Symmetric Double-Sided Time of Flight Ranging

Measuring signal propagation time requires precise hardware clocks and deterministic signal transmission times. In single-sided time of flight measurement, even small variations in measurement clock frequencies can cause large range errors due to the discrepancies in the measured round trip times (and reply times in the case of RF ranging). To overcome this, many ranging systems rely on double-sided ranging. In double-sided two-way ranging, a terminating final range measurement is sent by the initiating party after receiving the range response message. This allows the non-initiating party to greatly reduce errors due to clock discrepancies between the two parties, resulting in improved range accuracy. In this scenario, only the non-initiating party receives accurate time of flight estimates, as the initiating party does not necessarily know the receiving party's delay, $t_{\text{reply-A}}$. Furthermore, if the receiving party does not publicize the transmit power of their signal or if the power is variable over time, the initiating party (target) may know only that the receiving party is within communication range.

## 3.2 Related Work

We review the related work in concurrent synchronization and localization. Gholami *et al.* [80] derived a maximum likelihood estimator (MLE) for joint clock skew and position estimation. However, they assume that a number of reference nodes are perfectly synchronized with a reference clock and transmit their signals at a common time instant. In [81], Denis et al. derived a distributed maximum log likelihood estimator that fuses clock offset, bias, and range estimates. Similarly, joint time synchronization and localization is solved in [82] with accurate and inaccurate anchors in terms of time synchronization and localization. Moreover, a weighted least squares solution is proposed in [83] to achieve joint synchronization and localization using TOAs. While these approaches outline theoretical estimators, they make several non practical assumptions of the underlying network such as perfect synchronization among anchor nodes, and do not evaluate their methods on actual hardware. Recently, ATLAS estimates locations using a maximum likelihood method and known-position beacon transmitters to synchronize the clocks of receivers [84]. Positioning using time-difference of arrival measurements is proposed in [85]. In [86] the problem of solving for the position is investigated based on time of flight measurements for asynchronous networks using linear least squares.

Accurate RF time-stamping has encouraged research on low error ranging techniques and their related time synchronization spearheaded by impulse-radio ultra-wideband (IR-UWB) devices [87]. For instance, Polypoint [88] introduced a RF localization system which enables the real-time tracking and navigating of quadrotors through complex indoor environments. Furthermore, clock bias measurements were used in [89] to allow

for simpler range measurements using one-way TOA/TDOA messages compared to the more expensive two-way protocols. Another quadrotor localization framework was introduced in [90]. These works are based on the popular DecaWave DW1000 radio[1]. Recent developments in UWB communications offer high precision positioning through which a new range of applications is enabled [91]. Moreover, authors in [88] demonstrated a 56 cm localization error with frequency 20 Hz and density 0.005 node/$m^3$. On the other hand, the work in [89] achieves a 28 cm localization error. However, it used 100 Hz frequency with 0.037 node/$m^3$. Finally, authors in [90] shows a 20 cm localization error with frequency 10 Hz given a density of 0.054 node/$m^3$. Hybrid use of sample-based and Gaussian belief propagation is used in [92].

Impact of clock frequency offsets on the accuracy of ranging systems based on TOA measurements is analyzed in [93]. More application-specific solutions include using unmanned aerial vehicles to relay GPS information to a network of energy constrained nodes, who then localize and synchronize themselves to the mobile GPS receiver, while saving energy [94] and leveraging clock bias estimation to improve RF time-of-flight estimates for non-UWB transceivers [95]. A linear data model for joint localization and clock synchronization is proposed in [96]. Its estimates included batch offline least squares methods. Estimators to precisely estimate range and clock parameters from the measurements are suggested in [97]. It used master-slave clock synchronization to improve round-trip time estimates. For more complete treatments of localization and synchronization methods, we refer the reader to [98] and [99].

## 3.3 System Model

Consider a set of $N$ nodes indexed by $k \in \{0, \ldots, N-1\}$ distributed geographically over some region. We say that two nodes are connected if they can communicate with each other, and we denote the neighborhood of a given node by the set $\mathcal{N}_k$ that contains all the nodes that are connected to node $k$. Our state-space model is

$$\begin{aligned} x_{i+1} &= f_i(x_i) + n_i \\ y_{kj,i} &= h_{k,i}(x_i) + v_{k,i}, \end{aligned} \tag{3.1}$$

where the state at time $i$ is denoted by $x_i$, and the measurement available to node $k$ from the neighborhood node $j \in \mathcal{N}_k$ is represented by $y_{kj,i}$. The process and measurement noise are $n_i$ and $v_{k,i}$, respectively, and assumed to be Gaussian. The state update function is $f_i$ and $h_{k,i}$ is the measurement function. The process covariance matrix, and the measurement noise covariance matrix of node $j$ at time $i$ are denoted by $Q_i$ and $R_{j,i}$, respectively. The state vector of node $k$ is the state of the whole network nodes

$$x_{k,i} = [\bar{x}_{1,i}, ..., \bar{x}_{N,i}]^T \tag{3.2}$$

[1] http://www.decawave.com/products/dw1000

where $\bar{x}_{k,i}$ consists of three components $\bar{x}_{k,i} = \left[\boldsymbol{p}_{k,i}^T,\ o_{k,i},\ b_{k,i}\right]^T$. We denote the three dimensional position vector by $\boldsymbol{p}_{k,i}$. The clock time offset and clock frequency bias are denoted by $o_{k,i}$ and $b_{k,i}$, respectively. We adopt a convention where both $o_{k,i}$ and $b_{k,i}$ are described with respect to the leader node, which can be any node. It is recommended to pick the leader node in the middle of the network graph to achieve faster time synchronization. It is assumed that the nodes are static and that the clock parameters evolve according to the first-order affine approximation of the following dynamics,

$$\begin{aligned} o_{k,i+1} &= o_{k,i} + b_{k,i}\delta_t, \\ b_{k,i+1} &= b_{k,i}, \end{aligned} \tag{3.3}$$

where $\delta_t := t_{L,i+1} - t_{L,i}$ given that $t_L$ is the leader node time which is the global time. Therefore, we can write the update function for a static node as

$$f(x_{k,i}) \quad = \quad \begin{bmatrix} \boldsymbol{p}_{k,i} \\ o_{k,i} + b_{k,i}\delta_t \\ b_{k,i} \end{bmatrix}. \tag{3.4}$$

### 3.3.1 Measurement types

The D-SLATS architecture supports three types of measurements which are distinguished by the number of messages exchanged between a pair of nodes. These measurements types are shown in detail in Figure 3.2, where time stamps $t_0(i)$ through $t_5(i)$ denote the locally measured transmission (TX) and reception (RX) times stamps, and $T_{RSP}(i)$ and $T_{RND}(i)$ define, respectively, the response and the round-trip durations between the appropriate pair of these timestamps. The propagation velocity of radio is taken to be the speed of light in a vacuum, denoted by $c$. The three message types are:

- **Counter Difference**: This is type $1$ measurement which is a single transmission is sent from $j$ to $k$, yielding two timestamps, one from the TX instant and another from the RX instant. Type $1$ results in finding the counter difference at time $i$ denoted by $d_{kj,i}$ which is the measurement of the difference between the clocks of each node. This is a time measurement that includes the effects of propagation delay $T_p$.

$$d_{kj,i} = t_5(i) - t_4(i) \tag{3.5}$$

- **Single-Sided Two-Way Range**: This is type $2$ measurement, where a type $1$ message is followed by a reply message for round-trip timing calculations. Type $2$ results in the single-sided two-way range $r_{kj,i}$ which is a *distance measurement* between a pair of nodes j and k, with an error proportional to the response turnaround time $T_{RSP1}$. This is a noisy measurement due to frequency bias discrepancies be-

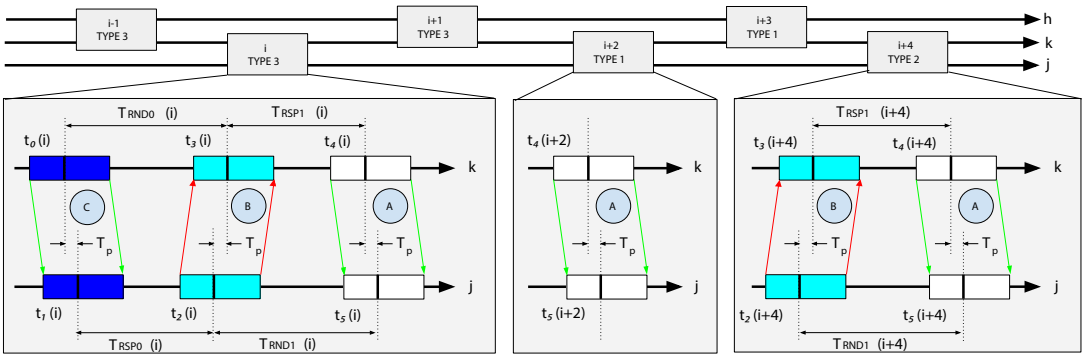



**Figure 3.2:** The top of this diagram shows six synchronization events between three devices, labeled $h$, $k$, and $j$. Each event is classified as type 1, type 2 or type 3 depending on the number of transmissions sent.

tween $k$ and $j$.

$$r_{kj,i} = \frac{c}{2}\left(T_{RND1}(i) - T_{RSP1}(i)\right) \tag{3.6}$$

- **Double-Sided Two-Way Range**: This is type 3 measurement. One last transmission completes a handshake trio allowing for a more precise round-trip timing calculation. Type 3 results in finding the *double-sided two-way range* $\Gamma_{kj,i}$, which is another *distance measurement* between nodes $k$ and $j$ based on a trio of messages between the nodes at time $i$. The error is proportional to the relative frequency bias between the two devices integrated over the period. This is a more accurate estimate than $r_{kj,i}$ due to mitigation of frequency bias errors from the additional message.

$$\Gamma_{kj,i} = c\frac{T_{RND0}(i)T_{RND1}(i) - T_{RSP0}(i)T_{RSP1}(i)}{T_{RND0}(i) + T_{RND1}(i) + T_{RSP0}(i) + T_{RSP1}(i)} \tag{3.7}$$

With that, the measurement vector at node $k$, from node $j \in \mathcal{N}_k$ has the form

$$y_{kj,i} \quad = \quad \begin{bmatrix} d_{kj,i} \\ r_{kj,i} \\ \Gamma_{kj,i} \end{bmatrix} \tag{3.8}$$

It is important to note that a subset of these measurements may be used rather than the full set, i.e, we can have experiments involving just $r_{kj,i}$, $\Gamma_{kj,i}$, or $d_{kj,i}$. The three measurements types can be translated to the state vector according to the clock model from (3.3) and the physics relating the propagation speed, the duration between the propagation and the distance between the nodes. Thus, we obtain measurement function in the following form [100]:

$$h_j(x_{k,i}) = \begin{bmatrix} (o_{j,i} - o_{k,i}) + \|\boldsymbol{p}_{j,i} - \boldsymbol{p}_{k,i}\|_2 / c \\ \|\boldsymbol{p}_{j,i} - \boldsymbol{p}_{k,i}\|_2 + \frac{c}{2}(b_{j,i} - b_{k,i}) T_{RSP1} \\ \|\boldsymbol{p}_{j,i} - \boldsymbol{p}_{k,i}\|_2 + c\tilde{R}_{kj,i} \end{bmatrix} \tag{3.9}$$

where

$$\tilde{R}_{kj,i} := \frac{\Big(b_{k,i} - b_{j,i}\Big)\Big(T_{RND0}(i)T_{RND1}(i) - T_{RSP0}(i)T_{RSP1}(i)\Big)}{\Big(1 + b_{k,i} - b_{j,i}\Big)T_{RND0}(i) + T_{RND1}(i) + T_{RSP0}(i) + \Big(1 + b_{k,i} - b_{j,i}\Big)T_{RPS1}(i)}$$

Next, we present our proposed algorithms for distributed localization and time synchronization between sensor nodes.

## 3.4 Proposed Algorithms

The time synchronization and localization problem focus on finding an estimate for the clock parameters and the 3D position (i.e the state of the system) using the timestamp measurements and the model of the system. We denote by $\hat{x}_{k,i|l}$ the estimate of $x_{k,i}$ given the observations up to time $l$ where every node seeks to minimize the mean-square error $\mathbb{E}\|x_{k,i} - \hat{x}_{k,i|l}\|^2$.

### 3.4.1 Distributed Kalman Filter

We are seeking a distributed implementation that avoids the use of a fusion center and instead distributes the processing and communication across the sensor network. Among distributed algorithms, diffusion algorithms are amenable for easy real-time implementations, for good performance in terms of synchronization and localization accuracy, and for the fact that they are robust to node and link failure. Our DKAL algorithm is derived from the Diffusion Extended Kalman filter presented in [77], which uses the information form rather than the conventional form. The advantage of the information form over the conventional form becomes more evident for some categories of problems. For instance, the information filter form is easier to distribute, initialize, and fuse compared to the conventional filter form in multisensor environments [101]. Furthermore, it can reduce dramatically the computation and storage which is involved in the estimation of specific classes of large-scale interconnected systems [102]. Also, the update equations for the estimator are computationally simpler than the equations of the conventional form.

We define the following matrices obtained by linearizing the state update and the measurement update functions around some point $z$.

**Algorithm 1:** DKAL

Start with $\hat{x}_{k,0|-1} = x_0$ and $P_{k,0|-1} = \Pi_0$ for all $k$, and at every time instant $i$, compute at every node $k$:

**Step 1**: Measurement update:

$$\hat{H}_{kj,i} = \bar{H}_j(\hat{x}_{k,i|i-1}) \tag{3.10}$$

$$\tilde{Q}_0 = P_{k,i|i-1} \tag{3.11}$$

$$\tilde{Q}_{j+1} = \tilde{Q}_j - \tilde{Q}_j \hat{H}_{kj,i}^T (R_{j,i} + \hat{H}_{kj,i} \tilde{Q}_j \hat{H}_{kj,i}^T)^{-1} \hat{H}_{kl,i} \tilde{Q}_j \quad \forall j \in \mathcal{N}_k \tag{3.12}$$

$$P_{i|i}^k = \tilde{Q}_N \tag{3.13}$$

$$\psi_{k,i} = \hat{x}_{k,i|i-1} + P_{k,i|i} \sum_{j \in \mathcal{N}_k} \hat{H}_{kj,i}^T R_{j,i}^{-1} [y_{kj,i} - h_j(\hat{x}_{k,i|i-1})] \tag{3.14}$$

**Step 2**: Diffusion update:

$$\hat{x}_{k,i|i} \leftarrow \sum_{j \in \mathcal{N}_k} c_{kj} \psi_{j,i} \tag{3.15}$$

**Step 3**: Time update:

$$\hat{x}_{k,i+1|i} = f_i(\hat{x}_{k,i|i}) \tag{3.16}$$

$$P_{k,i+1|i} = \bar{F}_i(\hat{x}_{k,i|i}) P_{k,i|i} \bar{F}_i(\hat{x}_{k,i|i})^T + Q_i \tag{3.17}$$

$$\bar{F}_i(z) := \frac{\partial f_i(x)}{\partial x}\Big|_{x=z}, \tag{3.18}$$

$$\bar{H}_{k,i}(z) := \frac{\partial h_{k,i}(x)}{\partial x}\Big|_{x=z}. \tag{3.19}$$

Given that, every node monitors the state on the whole network, we tackle the following challenges of the proposed algorithm in [77]:

- Taking the inverse of a large matrix is not feasible on embedded devices. A large-scale network implies inverting a matrix $P_{k,i|i}$ on each node with dimension $N \times N$ in (3.20).

$$P_{k,i|i}^{-1} = P_{k,i|i-1}^{-1} + \sum_{j \in \mathcal{N}_k} H_{j,i}^T R_{j,i}^{-1} H_{j,i}. \tag{3.20}$$

- The size of $P$ increases dramatically with large-scale systems.

To address the first issue, we propose modifying the measurement update in Algorithm 2 in [77] by using the Binomial inverse theorem. This provides a formula to compute

$P_{k,i|i}$ from $P_{k,i|i-1}$ without having to invert any $N \times N$ matrix. More specifically, if we consider one neighbor in (3.20), we get

$$P_{k,i|i} = \left(P_{k,i|i-1}^{-1} + H_{j,i}^T R_{j,i}^{-1} H_{j,i}\right)^{-1}. \tag{3.21}$$

The Binomial inverse theorem states that

$$(A + UBV)^{-1} = A^{-1} - A^{-1}U\left(B^{-1} + VA^{-1}U\right)^{-1} VA^{-1}. \tag{3.22}$$

Applying (3.22) to (3.21) results in

$$\begin{aligned} P_{k,i|i} &= \left(P_{k,i|i-1}^{-1} + H_{j,i}^T R_{j,i}^{-1} H_{j,i}\right)^{-1} \\ &\overset{(3.22)}{=} P_{k,i|i-1} - P_{k,i|i-1}\hat{H}_{kj,i}^T (R_{j,i} + \hat{H}_{kj,i} P_{k,i|i-1} \hat{H}_{kj,i}^T)^{-1} \hat{H}_{kj,i} P_{k,i|i-1} \end{aligned} \tag{3.23}$$

Repeating (3.23) to all neighbors results in the first step in Algorithm 1. The other issue of monitoring the whole network state will be addressed by the DKALarge and DOPT algorithms. The DKAL algorithm is comprised of three main steps: measurement update, diffusion update section, and time update steps. The algorithm starts with the measurement update where every node $k$ obtains $\psi_{k,i}$ at time step $i$. Next, in the diffusion step, information from the neighbors of node $k$ are combined in a convex combination to produce a new state estimate for the node. The $c_{kj}$ elements represent the weights that are used by the diffusion algorithm to combine neighborhood estimates. Finally, every node performs the time update step. The proposed algorithm can be summarized as shown in Algorithm 1.

### 3.4.2 Distributed Kalman Filter for Large Scale Systems

As mentioned before, the disadvantage of the DKAL algorithm that the size matrix $P$ is increasing dramatically for large-scale systems. Therefore, we are going to tackle these disadvantages by proposing the DKALarge algorithm. The key idea behind the DKALarge algorithm is to let every node monitor only its neighbors (its subsystem). Therefore, the size of the $x$ and $P$ will depend only on the number of neighbors which solves the two disadvantages of the DKAL algorithm. The DKALarge algorithm spatially decomposes a sparsely connected large-scale network before carrying out Kalman filtering operations. Network connectivity is used to break down the global system into sets of smaller subsystems, while maintaining inter-subsystem graph rigidity.

Each subsystem is indexed by $l$ and we denote the matrix $P$ of the subsystem by $P^{(l)}$ and the state of the subsystem by $x^{(l)}$. Figure 3.3 shows an example of how a network can be broken down into three subsystems. Subsystem $l = 0$, which corresponds to node $n_0$, monitors only nodes $n_0$, $n_1$, and $n_2$ so that the subsystem state vector and the $P^{(l)}$ matrix are with reduced dimension because they only contain information about estimates for three nodes. Similarly, subsystem $l = 3$ monitors only nodes $n_1$, $n_3$, and $n_4$, whereas subsystem $l = 4$ considers only nodes $n_3$ and $n_4$. The size of the state vectors and the $P$

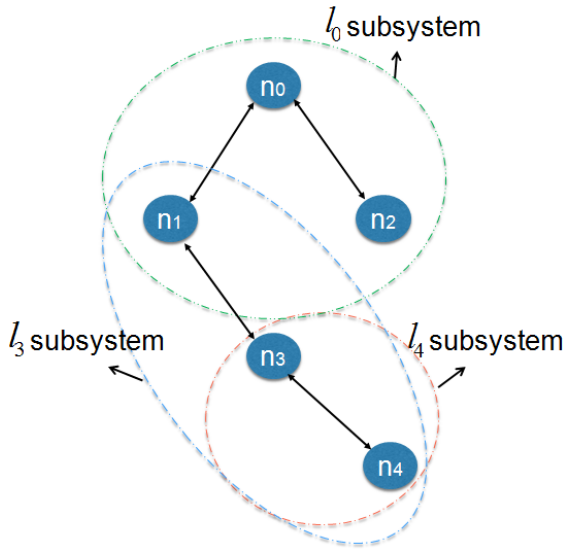



**Figure 3.3:** An example of how a system can be divided in three subsystems.

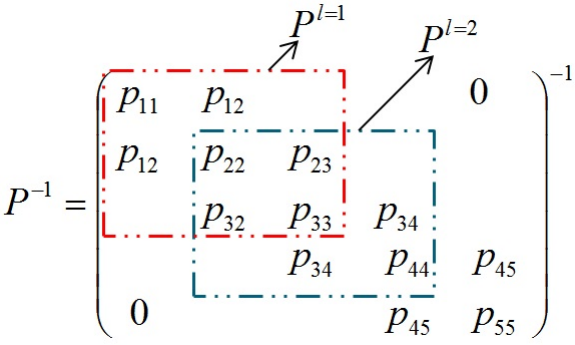



**Figure 3.4:** Example of $P^{(l)}$ values out of original $P$.

matrices are thus of a reduced form representative of the nodes defined to be in a given subsystem as shown in Figure 3.4. The important step in this algorithm is to decompose the whole systems into smaller subsystems and let every node monitors only the nodes that are in its own subsystem. For the local $P$ matrix, it is known that $(P^{(l)})^{-1}$, which is the normal inverse of the reduced $P^{(l)}$ matrix, does not equal $(P^{-1})^{(l)}$, which is the $l$ submatrix of $P^{-1}$; this represents a challenge in decomposing the whole system into subsystems. We remove the superscript $l$ for simplicity from now on. The DKALarge algorithm addresses the previous challenge of distributed matrix inversions by employing the L-banded inverse on the matrix $P$, followed by the DICI-OR method presented in [78] in order to approximate the inverse of the large covariance $P$ matrix. We approximate $P^{-1}$ matrices to be L-baneded matrices. This approach of approximation is studied in the centralized filter in [103] where the information loss due to the approximation is bounded. The methods which are used for inversion of the local matrix $P$ can be summarized as follows:

**Algorithm 2:** DKALarge

Start with $\hat{x}_{k,0|-1} = \mathbf{E}x_0$ and $P_{k,0|-1} = \Pi_0$ for all $k$, and compute at every node $k$:

**Step 1**: Measurement update:

$$P^{-1}_{k,i|i-1} \xleftarrow{Lbandinv} P_{k,i|i-1} \tag{3.24}$$

$$P^{-1}_{k,i|i} = P^{-1}_{k,i|i-1} + \sum_{j\in\mathcal{N}_k} H^T_{kj,i} R^{-1}_{j,i} H_{kj,i} \tag{3.25}$$

$$P_{k,i|i} \xleftarrow{DICIOR} P^{-1}_{k,i|i} \tag{3.26}$$

$$\psi_{j,i} = \hat{x}_{k,i|i-1} + P_{k,i|i} \sum_{j\in\mathcal{N}_k} H^T_{kj,i} R^{-1}_{j,i} (y_{kj,i} - h_{j,i}(\hat{x}_{k,i|i-1})$$

**Step 2**: Diffusion update:

$$\hat{x}_{k,i|i} \leftarrow \sum_{j\in\mathcal{N}_k} c_{kj}\psi_{j,i} \tag{3.27}$$

**Step 3**: Time update:

$$P_{k,i+1|i} = \bar{F}_i(\hat{x}_{k,i|i}) P_{k,i|i} \bar{F}_i(\hat{x}_{k,i|i})^T + Q_i \tag{3.28}$$

$$\hat{x}_{k,i+1|i} = f_i(\hat{x}_{k,i|i}) \tag{3.29}$$

- **L-Banded Inverse**: We use the L-banded inversion provided in [104] in order to obtain an L-banded approximate matrix $P_k^{-1}$ from $P_k$. The L-banded structure is necessary in the application of the DICI-OR method stated below. The L-banded inverse method contains properties for approximating Gaussian error processes, which help with inverses of submatrices. This has also been studied in [105] and [106].

- **DICI-OR Method**: The DICI-OR [78] method uses the L-banded structure of the $P_k^{-1}$ matrix in achieving distributed matrix inversion from $P_k^{-1}$ to $P_k$. With the assumption that the non L-band elements are not specified in the matrix $P_k$, a collapse step is effected in order to fill in the non L-band elements. The iterate step is then employed to produce the elements that lie within the L-band of $P_k$. The iterate step can be repeated to achieve quicker convergence in the state estimates.

The DKALarge algorithm leverages L-banded inverse and the DICI-OR method to achieve a distributed matrix inversion. Unlike DKAL algorithm, where the $P$ matrix and state vectors $x$ are needed to be maintained at each node, these two methods enable us to decompose the problem into smaller parts.

### 3.4.3 Distributed Optimization

Algorithm 3 tackle the changeless of Algorithm 1. It basically operates by letting each node monitors its neighbor nodes only, so the size of $\hat{x}$ is limited locally by the number of connecting nodes, an improvement comparing to the DKAL algorithm. In this algorithm, the unconstrained minimizations are solved numerically and simultaneously at each node by computing the values for the parameters that satisfy the first order optimality conditions. DOPT can be summarize in the steps presented in Algorithm 3.

**Algorithm 3:** DOPT

Start with $\hat{x}_{k,0} = x_0$. At every time instant $i$, do at every node $k$:
**Step 1**: Get from neighboring nodes $j \in \mathcal{N}_k$ their estimated state $\hat{x}_{j,i} = \left[\hat{\boldsymbol{p}}_{j,i}^T,\ \hat{o}_{j,i},\ \hat{b}_{j,i}\right]^T$.
**Step 2**: Find optimal estimate $\hat{x}_{k,i+1}$ of node $k$ based on the received estimates and measured distances:

$$[\hat{o}_{k,i+1}\ \hat{\boldsymbol{p}}_{k,i+1}] = \underset{o_{k,i},\boldsymbol{p}_{k,i}}{\operatorname{argmin}} \sum_{j\in\mathcal{N}_k} \left(d_{kj,i} - (o_{k,i} - \hat{o}_{j,i}) - \left\|\boldsymbol{p}_{k,i} - \hat{\boldsymbol{p}}_{j,i}\right\|_2 / c\right)^2 \quad (3.30)$$

$$[\hat{o}_{k,i+1}\ \hat{\boldsymbol{p}}_{k,i+1}] = \underset{o_{k,i},\boldsymbol{p}_{k,i}}{\operatorname{argmin}} \sum_{j\in\mathcal{N}_k} \left(t_2(i) - t_3(i) - (o_{k,i} - \hat{o}_{j,i}) - \left\|\boldsymbol{p}_{k,i} - \hat{\boldsymbol{p}}_{j,i}\right\|_2 / c\right)^2 \quad (3.31)$$

$$\left[\hat{b}_{k,i+1}\ \hat{\boldsymbol{p}}_{k,i+1}\right] = \underset{b_{k,i},\boldsymbol{p}_{k,i}}{\operatorname{argmin}} \sum_{j\in\mathcal{N}_k} \left(\Gamma_{kj,i} - \left\|\boldsymbol{p}_{k,i} - \hat{\boldsymbol{p}}_{j,i}\right\|_2 - c\tilde{R}_{kj,i}\right)^2 \quad (3.32)$$

The steps in Algorithm 3 have considered type 3 messages which are more accurate. However, depending on the application considered, type 2 can be enabled instead by replacing equation 3.32 with equation 3.33.

$$\underset{b_{k,i},\boldsymbol{p}_{k,i}}{\operatorname{argmin}} \sum_{j\in\mathcal{N}_k} \left(r_{kj,i} - \left\|\boldsymbol{p}_{k,i} - \boldsymbol{p}_{j,i}\right\|_2 - \frac{c}{2}\left(b_{k,i} - b_{j,i}\right) T_{RSP1}\right) \quad (3.33)$$

## 3.5 Evaluation

We present the experimental setup, where we conducted our experiments. Next, we evaluate case studies of our proposed algorithms.

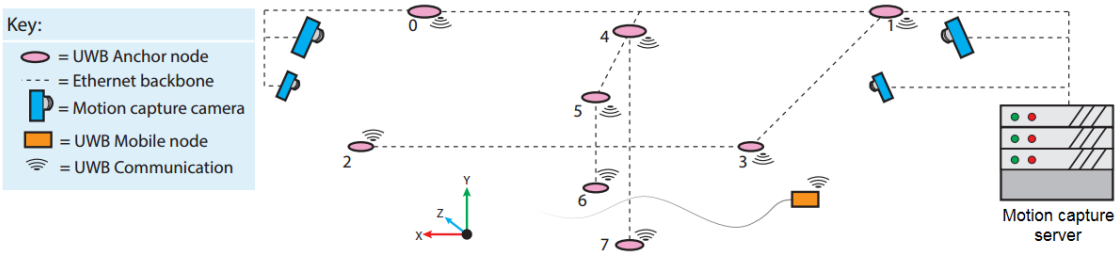



**Figure 3.5:** Experimental setup overview, including, UWB Anchor nodes, motion capture cameras, and mobile quadrotor UWB nodes.

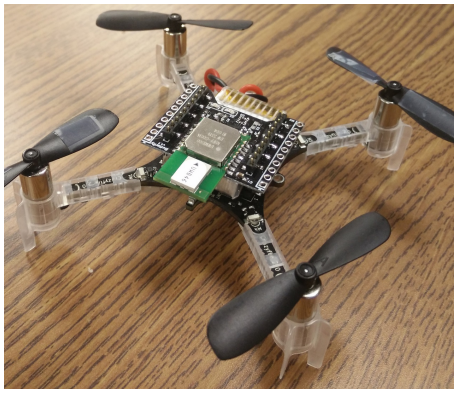


**Figure 3.6:** CrazyFlie 2.0 quadrotor helicopter with DW1000 UWB expansion.

### 3.5.1 Experimental Setup

We evaluated the performance of our algorithms on a custom ultra-wideband RF testbed based on the DecaWave DW1000 IR-UWB radio. The overall setup is shown in Figure 3.5. The main components of our testbed can be summarized as follows:

- A motion capture system capable of 3D rigid body position measurement with less than 0.5 mm accuracy. The system consists of an eight-camera set-up which is deployed to provide accurate ground truth position measurements. The ground truth positions from the motion capture cameras are sent to a centralized server that uses the Robot Operating System (ROS) [107] with a custom package. The presented results treat the motion capture estimates as true positions, though we qualify here that all results are accurate to within the motion capture accuracy. We adopt a right-handed coordinate system where $y$ is the vertical axis, and $x$ and $z$ make up the horizontal plane.

- Fixed nodes consist of custom-built circuit boards equipped with ARM Cortex M4 processors $196$MHz powered over Ethernet and communicating to Decawave DW1000 ultra-wideband radios as shown in Figure 2.17. Each anchor performs a single and double-sided two-way range with its neighbors. The used Decawave radio is equipped with a temperature-compensated crystal oscillator with frequency equals $38.4$ MHz and stated frequency stability of $\pm 2$ ppm. We installed eight

UWB anchor nodes in different positions in a $10 \times 9\ m^2$ lab. More specifically, six anchors are placed on the ceiling at about $2.5m$ height, and two were spotted at waist height at about 1m to better disambiguate positions on the vertical axis. Each anchor node is fully controllable over a TCP/IP command structure from the central server. These nodes are placed to remain mostly free from obstructions, maximizing line-of-sight barring pedestrian interference.

- A mobile node consists of a battery-powered node also with ARM Cortex M4 processors based on the CrazyFlie 2.0 helicopter[2], and equipped with the same DW1000 radio as shown in Figure 3.6. This allows for compatibility in the single and double-sided ranging technique used.

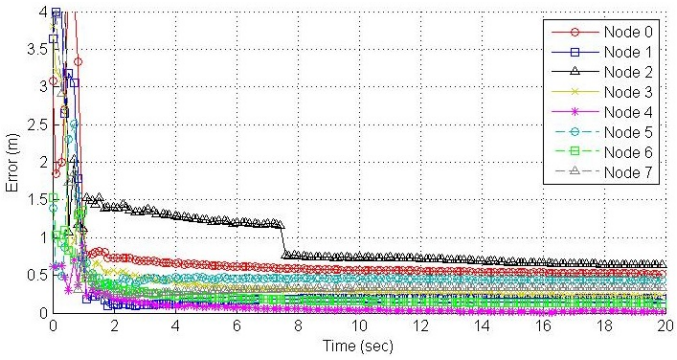



**Figure 3.7:** Average localization error with fully connected network for DKAL algorithm.

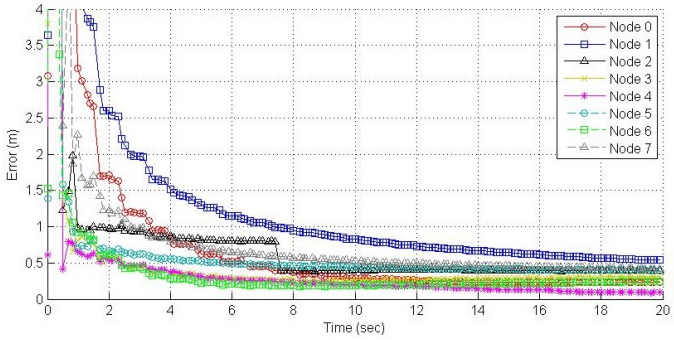



**Figure 3.8:** Average localization error with fully connected network for DKALarge algorithm.

### 3.5.2 Case Study: Static Nodes

We demonstrate D-SLATS performance first in distributed simultaneous localization and time synchronization of static nodes. The goal of D-SLATS is to accurately estimate

[2]Bitcraze CrazyFlie 2.0. `https://www.bitcraze.io/`

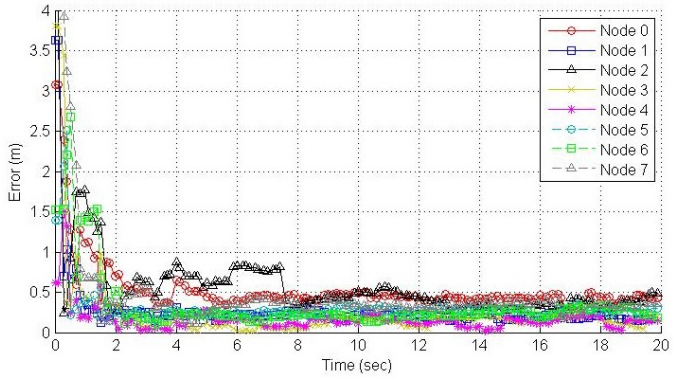



**Figure 3.9:** Average localization error with fully connected network for DOPT algorithm.

the positions of all network devices relative to one another, as well as, the relative clock offsets and frequency biases. This relative localization, or graph realization as it is sometimes called, is a well-researched field. Local minima, high computational complexity, and restrictions on graph rigidity are the main challenges in graph realization problems [108, 109]. D-SLATS does not put restrictions on the connectivity of the graph. A centralized Kalman filter (CKAL) is implemented as a baseline for comparison.

#### 3.5.2.1 Position Estimation

The D-SLATS algorithms find relative positions. Therefore, we first superimpose the estimated positions onto the true positions of each node by use of a Procrustes transformation [110]. Specifically, the network topology as a whole is rotated and translated without scaling, until it most closely matches the true node positions. Once transformed, the error of a given node's position is defined as the $\ell_2$ norm of the transformed position minus the true position.

We begin by showing the localization error of a fully-connected network for all algorithms. Figures 3.7, 3.8 and 3.9 show the localization errors for DKAL, DKALarge, and DOPT algorithms, respectively, with type 3 enabled for all algorithms. Enabling type 3 inherits enabling type 1 and type 2, as type 3 is a replicate of type 1 and type 2. Table 5.1 summarizes the localization error of the eight nodes using the DKAL, DKALarge, DOPT, and CKAL algorithms. DKAL algorithm achieves $0.311m$. While DKALarge and DOPT algorithms report $0.330m$ and $0.299m$, respectively. Also, we should not that enabling type 2 only instead of type 3 in our experiments does not have a great effect of the overall performance.

We summarize the mean error of the position estimation for all nodes and for different localization algorithms in Figure 3.10. The centralized algorithm outperforms the distributed algorithms in D-SLATS, as expected. Note that DOPT algorithm has the fastest convergence time. In a second experiment, we show that D-SLATS supports different network topologies without requiring the graph to be fully connected. Figure 3.11 shows the localization error for the network topology where every node is only connected to

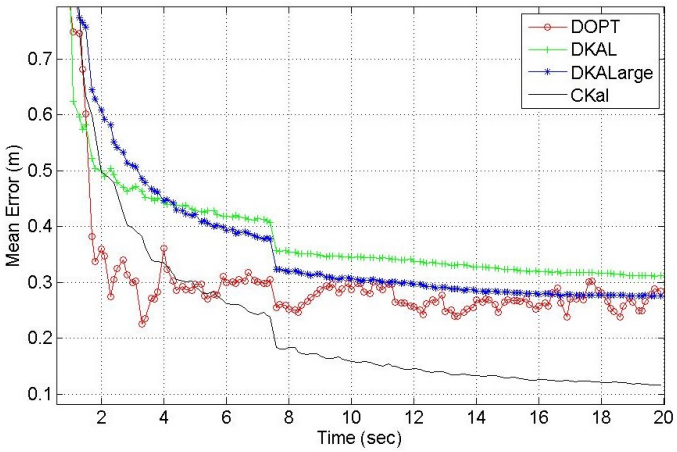



**Figure 3.10:** Average localization error with fully connected network.

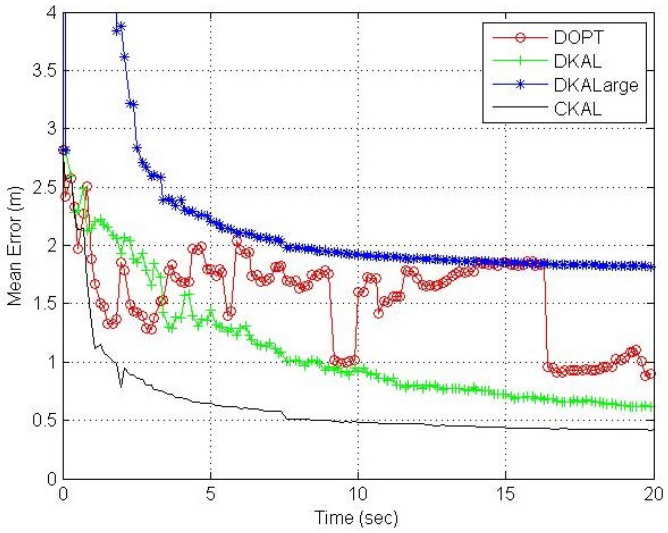



**Figure 3.11:** Average localization error where each node has 4 neighbors only.

four neighbors. The DKALarge algorithm gave the worst performance as the DICI-OR algorithms is approximating the full covariance matrix inverse, as described before.

#### 3.5.2.2 Time synchronization

In order to test the time synchronization, we could configure a third party node to send a query message and compare the timestamp upon receiving that message as done in [111]. However, in order to decrease the uncertainty in our testing mechanism, we choose the root node to do this job. This testing mechanism is better than others as reported in [28], and has been used also in [112]. As mentioned before, we choose node 0 to be the reference node, i.e, $\forall i : b_{0,i} := 0$ and $o_{0,i} := 0$. Table 3.2 shows the synchronization errors for all nodes with respect to node 0. The DKAL algorithm has the best performance,

**Table 3.1:** localization error (m) of different static nodes.

| Algorithm | node 0 | node 1 | node 2 | node 3 | node 4 | node 5 | node 6 | node 7 | mean | std |
|---|---|---|---|---|---|---|---|---|---|---|
| DKAL | 0.518 | 0.189 | 0.638 | 0.228 | 0.021 | 0.433 | 0.126 | 0.336 | 0.311 | 0.209 |
| DKALarge | 0.232 | 0.536 | 0.394 | 0.318 | 0.093 | 0.400 | 0.246 | 0.418 | 0.330 | 0.137 |
| DOPT | 0.402 | 0.202 | 0.530 | 0.193 | 0.156 | 0.309 | 0.199 | 0.397 | 0.299 | 0.133 |
| CKAL | 0.205 | 0.189 | 0.208 | 0.147 | 0.218 | 0.075 | 0.168 | 0.143 | 0.169 | 0.047 |

**Table 3.2:** Synchronization error ($\mu$ seconds) of different nodes with respect to node 0.

| Algorithm | node 1 | node 2 | node 3 | node 4 | node 5 | node 6 | node 7 | mean | std |
|---|---|---|---|---|---|---|---|---|---|
| DKAL | 0.807 | 9.088 | 1.868 | 2.332 | 5.936 | 9.624 | 5.342 | 5.000 | 3.502 |
| DKALarge | 5.223 | 6.448 | 5.203 | 5.339 | 5.863 | 3.313 | 4.222 | 5.087 | 1.036 |
| DOPT | 2.15 | 0.891 | 2.090 | 2.343 | 1.651 | 5.215 | 3.293 | 2.520 | 1.391 |
| CKAL | 1.362 | 2.045 | 1.440 | 1.517 | 1.792 | 0.267 | 0.708 | 1.304 | 0.617 |

followed by DKALarge and DOPT algorithms. We should note that the synchronization errors in Table 3.2 are reported for a fully connected network.

### 3.5.3 Case Study: Mobile Nodes

We have shown that D-SLATS can be used to localize and synchronize a network of static nodes. We now present the case of a heterogeneous network containing both static and mobile node. To the eight anchor nodes topology used in Section 3.5.2, we add one mobile node in the form of a CrazyFlie quadrotor as shown in Figure 3.6. We analyzed the results of running D-SLATS based on type 3 measurements with a fully connected graph.

A number of experiments were performed with quadrotors traveling with variable velocities. Figures 3.12, 3.13, and 3.14 show the results of traveling self-localizing quadrotor using DKAL, DKALarge, and DOPT, respectively, for type 3 messages. The left plot of the three figures shows a 3D comparison of the estimated position and the ground truth position reported by the motion capture cameras. DKAL algorithm achieved the best self localization estimation with an RMSE of $75cm$. The DKALarge and DOPT algorithms reported $89cm$ and $116cm$, respectively. The top left subfigures of the three figures show the 3D localization error. The localization errors in each axis are separately shown in the top right subfigure. Finally, the location of the mobile node in the network determines to some extent the accuracy with which it can be localized. A device whose location is more central to the network (i.e. closer to the centroid defined as the mean of all node positions) is more likely to be fully constrained in terms of its relative position. This correlation can be loosely seen in the bottom right of the three figures.

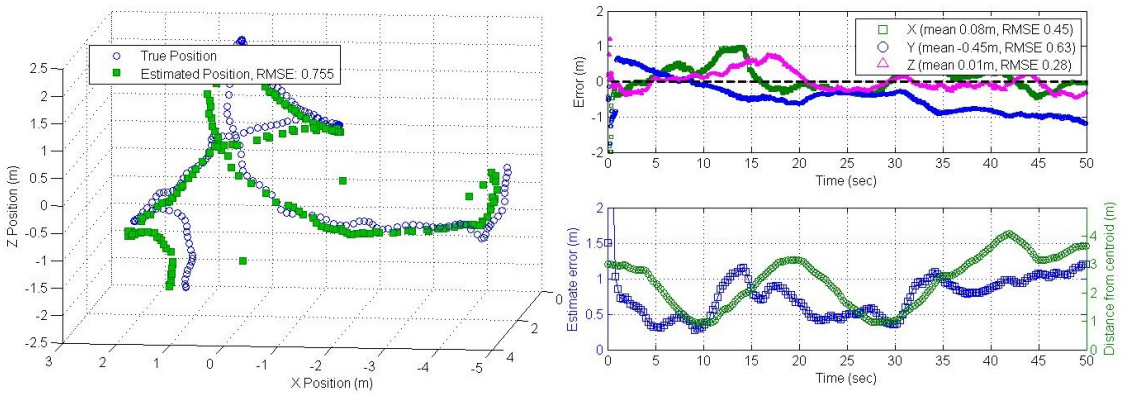



**Figure 3.12:** Localization errors for DKAL in 3D for a single mobile node. Spatial errors (left) are shown with corresponding per-axis errors by time (top right). Additionally, the error is plotted against the mobile node's distance from the network centroid (bottom right).

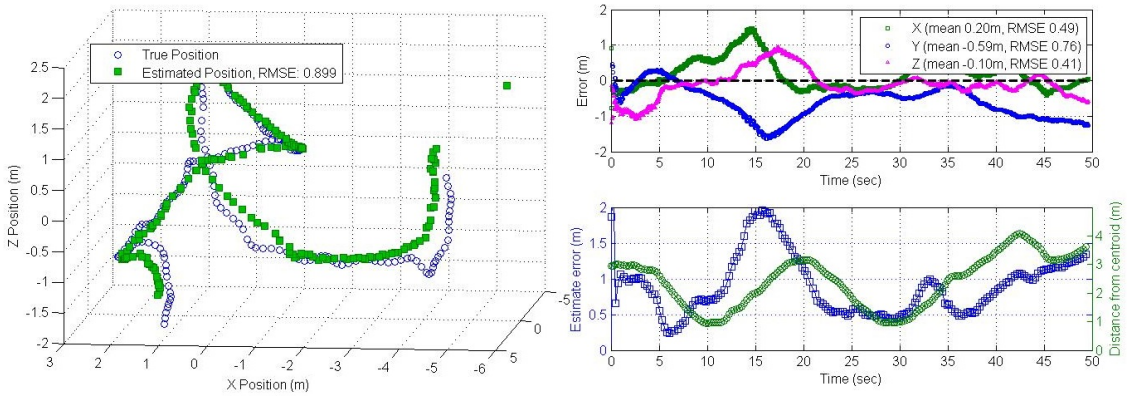



**Figure 3.13:** Localization errors for DKALarge in 3D for a single mobile node. Spatial errors (left) are shown with corresponding per-axis errors by time (top right). Additionally, the error is plotted against the mobile node's distance from the network centroid (bottom right).

## 3.6 Conclusions

This chapter describes and evaluates D-SLATS, an architecture for distributed simultaneous time synchronization and localization of static and mobile nodes in a network. Three different algorithms were proposed, namely DKAL, DKALarge, and DOPT, that perform distributed estimation in a scalable fashion. Several experiments using real, custom ultra-wideband wireless anchor nodes and mobile quadrotor nodes were conducted and they indicate that the proposed architecture is reliable in terms of performance, and efficient in the use of computational resources. D-SLATS is made possible by state-of-the-art advances in commercial ultra-wideband radios, and continued improvements to these devices will further underscore the importance of treating temporal and spatial

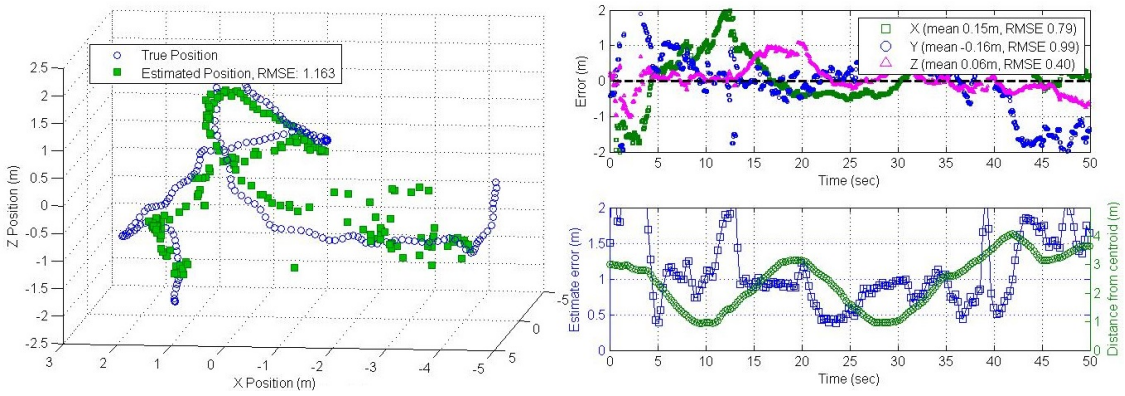



**Figure 3.14:** Localization errors for DOPT in 3D for a single mobile node. Spatial errors (left) are shown with corresponding per-axis errors by time (top right). Additionally, the error is plotted against the mobile node's distance from the network centroid (bottom right).

variables in a joint fashion. Future directions will deal with testing over a real large scale network by considering more nodes. Also, a secure estimation of nodes under malicious attacks is another direction.

# 4 Event-Triggered Diffusion Kalman Filter

State estimation is essential for habitat monitoring, emergency rescue, homeland security, military operations, and home automation services [113]. Besides ensuring the accuracy of the state estimation, one has to consider power constraints [13], limitations in terms of bandwidth [14], and limitations in computation [15] and communication [16]. One of the most popular estimation algorithms for sensor networks is the distributed Kalman filtering algorithm. Among distributed Kalman filters, diffusion algorithms, which have been utilized in the algorithms of Chapter 3, have favorable properties with respect to performance and robustness of node and link failures. The performance of distributed diffusion Kalman filtering [17] depends on frequent measurements and message exchange between nodes. However, the capabilities of individual nodes are very limited, and each node is often battery-powered. Thus, decreasing the communication overhead and the number of measurements is of great importance. There is no meaning of spending more resources, while the application need is much less. So the question should not be how much the algorithm could achieve - it should be - how well the algorithm is capable of satisfying the application needs while saving the resources?.

We propose an event-triggered diffusion Kalman filter algorithm that restricts the amount of processing, sensing, and communication-based on a local signal indicative of the estimation error. Thus, the number of transmission messages compared to the nominal distributed diffusion Kalman filter algorithm is significantly reduced. In particular, we characterize the trade-off between the spent resources and the corresponding estimation performance. A representative application of distributed state estimation is localization and time synchronization of sensor nodes. More specifically, we apply our event-triggered diffusion Kalman filer on D-SLATS presented in Chapter 3, which is a distributed simultaneous localization and time synchronization framework.

This chapter in based on our publication [25]. The rest of this chapter is organized as follows: Section 4.1 reviews related work. Section 4.2 gives the motivation behind our chosen triggering condition. We present our proposed algorithm and its theoretical analysis in Sections 4.3 and 4.4, respectively. Section 4.5 illustrates the application to localization and time synchronization, presents the experimental setup, and evaluates the proposed algorithm on static and mobile networks of nodes. Finally, Section 4.6 lists some concluding and discussion remarks.

## 4.1 Related Work

We first discuss general state estimation algorithms followed by centralized and distributed event-triggered estimators.

### 4.1.1 State Estimation Algorithms

Distributed estimation algorithms are widely used in wireless networks to reconstruct the system state from noisy measurements. Algorithms for diffusion least-mean squares [114], diffusion recursive least-squares [115], and diffusion Kalman filtering [116] have been proposed. Also, estimation algorithms based on average consensus have been analyzed in [117–119]. The proposed distributed estimation algorithm in [120] deals with extremely large-scale systems. The main idea is to approximate the inverse of the large covariance matrix $P$ by using the L-Banded inverse and DICI-OR method [121]; however, this requires a lot of resources. Due to limited resources in wireless sensor networks, many investigations have been made to decrease the communication and computation overheads, while preserving the performance. This leads to the work in the next categories.

### 4.1.2 Centralized Event-Triggered Estimation Algorithms

The event-triggered scheme has already been applied to network estimation problems. It was first proposed for centralized estimation problems. Send-on-delta is proposed for Kalman filters in [122], where sensor data values are transmitted only when encountering a user-defined change. An event-triggered sensor data scheduler has been proposed based on the minimum mean-squared error (MMSE) in [123]. The variance-based triggering scheme has been developed in [124], where each node runs a copy of the Kalman filter and transmits its measurement only if the associated measurement prediction variance exceeds a chosen threshold. The properties of set-valued Kalman filters with multiple sensor measurements have been analyzed in [125].

In general, the required communication can be reduced by the event-triggered scheme, when the sensor and the estimator are not on the same node, as in [126, 127]. Also, a discrete-time approach is proposed in [128] to address the same concern. The importance of including the effects of external disturbances and measurement noise in the analysis of event-triggered control systems is shown in [129]. Event-triggered centralized state estimation for linear Gaussian systems is proposed in [130]. Finally, the covariance intersection algorithm is investigated to get a centralized event-triggered estimator [131].

### 4.1.3 Distributed Event-Triggered Estimation Algorithms

As one of the main goals of sensor networks is to perform estimation distributively, event-triggered approaches are also applied in these scenarios, including Kalman filters with covariance intersection [132]. Interestingly, send-on-delta data transmission mechanisms are proposed in the event-triggered Kalman consensus filters [133]. Moreover,

event triggering on the sensor-to-estimator channel and estimator-to-estimator channel are investigated in distributed Kalman consensus [134]. Transmission delays and data drops in a distributed event-triggered control system are considered in [135]. Also, multiple distributed sensor nodes are considered in [136], where the sensors observe a dynamic process and sporadically exchange their measurements to estimate the full state of the dynamic system. Significant deviation from the information predicted from the last transmitted measurement is monitored to get a data-driven distributed Kalman filter [132]. For more related work in the domain, the reader is referred to [137].

When it comes to the herein considered event-triggered diffusion Kalman filters, we only found two previous works. The first one is a partial diffusion Kalman filter [138], which is mainly addressing the diffusion step. Every wireless node shares only a subset of its intermediate estimate vectors among its neighbors at each iteration. However, there is no saving at the measurement update step, which already includes high communication and measuring overheads. Also, it is not duty cycling the whole communication at the diffusion step. On the other hand, the concern of the other work in [139] is the measurement update step while neglecting the diffusion step, which also has significant overhead, as shown in [138]. We consider both the diffusion step and the measurement step; we temporarily shut-down the sensing and the communication between nodes. Also, we do not depend on monitoring the change between the expected state and the calculated one. To the best of our knowledge, our work is the first work in proposing event-triggering on the diffusion Kalman filter on both steps, based on an internal signal. Also, we are evaluating the mechanism on a real testbed for localization and time synchronization.

## 4.2 Triggering Logic Principle

One of the merits of the original centralized Kalman filter is the error covariance matrix. It is a perfect measure of the expected accuracy of the estimated state and can be utilized for regulating the resource consumption based on the application need. However, when it comes to the distributed diffusion Kalman filter, we do not have local access to the error covariance matrix [17]. Thus, we aim to obtain in this work the expected accuracy of the estimated state in the distributed diffusion Kalman filter, where the local estimators do not have access to all the measurements. Towards achieving our goal, let us start by a background example.

**Example 4.2.1.** *Let us introduce $\hat{x}_1$ as least-mean-squares estimator of $x$ given a zero-mean observation $y_1$, $\hat{x}_2$ as least-mean-squares estimator of $x$ given a zero-mean observation $y_2$, and $\hat{x}$ as least-mean-squares estimator of $x$ given all observations. As a consequence, we have two separate estimators for $x$ given two separate measurements and a global estimator given all the measurements. Let $P_1$, $P_2$ and $P$ denote the corresponding local and global error covariance matrices. We assume the measurement noises are uncorrelated and have zero-mean. It can be shown [140, p.89] that the global and local error covariance matrices are related via*

$$P^{-1} = P_1^{-1} + P_2^{-1} - R_x, \tag{4.1}$$

*where $R_x$ is the positive-definite covariance matrix of $x$.*

When it comes to the distributed Kalman filter, every node gets access to the measurements of its neighbors plus its local measurements. Thus, we need to introduce two terms: *individual* and *local* estimates. The individual estimate considers only the measurements at the node without its neighbors, while the local one considers the individual measurements plus the measurements of its neighbors. More specifically, we denote the individual estimate by $\hat{x}^{\text{ind}}_{k,i|i}$, which corresponds to the optimal linear estimate of $x_i$ given only the individual measurements at node $k$ without its neighbours, and the individual error covariance matrix by $P^{\text{ind}}_{k,i|i}$. The local estimate at node $k$ is denoted by $\hat{x}^{\text{loc}}_{k,i|i}$ and corresponds to the optimal linear estimate of $x_i$ given its individual measurements and measurements across the neighborhood of node $k$. The local error covariance matrix is denoted by $P^{\text{loc}}_{k,i|i}$. We also denote the global estimate by $\hat{x}_{i|i}$, which corresponds to the optimal linear estimate of $x_i$ given all observations across all nodes on the network and its error covariance matrix by $P_{i|i}$.

The global error covariance matrix $P_{i|i}$ provides the notion of the expected estimation error. Thus, we can duty cycle collecting measurements and messaging exchange processes based on a threshold on the trace of the global error covariance matrix $P_{i|i}$. However, every node has only access to its local matrix $P^{\text{loc}}_{k,i|i}$ in distributed Kalman filter algorithms and does not have access to $P_{i|i}$ locally. So let us find out if there is a direct relation between $P_{i|i}$ and $P^{\text{loc}}_{k,i|i}$. Instead of two nodes, we extend (4.1) to $N$ nodes, where every node only uses its individual measurements [116, p.14]:

$$P^{-1}_{i|i} = \sum_{k=1}^{N} (P^{\text{ind}}_{k,i|i})^{-1} - (N-1)\Pi_i^{-1}, \tag{4.2}$$

where $\Pi_i$ is the covariance matrix of $x_i$. The individual error covariance matrices $P^{\text{ind}}_{k,i|i}$ are expected to get smaller with time for observable systems. Therefore, their inverses in the first term $\sum_{k=1}^{N} (P^{\text{ind}}_{k,i|i})^{-1}$ in (4.2) become dominant. Thus, $P^{-1}_{i|i}$ can be approximated by

$$P^{-1}_{i|i} \approx \sum_{k=1}^{N} (P^{\text{ind}}_{k,i|i})^{-1}. \tag{4.3}$$

Now, let us apply (4.1) considering sharing the measurements between the neighbours, i.e, considering the local estimates. We can find that that the local error covariance $P^{\text{loc}}_{k,i|i}$ of each node $k$ depends on $P^{\text{ind}}_{l,i|i}$ of each neighbour $l$ as the local estimation process considers the neighbors measurements. Thus, we can relate $P^{\text{loc}}_{k,i|i}$ to $P^{\text{ind}}_{k,i|i}$ in (4.4) with the aid of the adjacency matrix $A$ which has unity entry if the corresponding nodes are neighbors, and zero otherwise. The element at row $l$ and column $k$ of matrix $A$ is denoted by $[A]_{l,k}$.

$$(P^{\text{loc}}_{k,i|i})^{-1} = \sum_{l=1}^{N} [A]_{l,k} (P^{\text{ind}}_{l,i|i})^{-1} - \Big( \sum_{l=1}^{N} [A]_{l,k} - 1 \Big) \Pi_i^{-1}, \tag{4.4}$$

where

$$[A]_{l,k} = \begin{cases} 1 & \text{if } l \in \mathcal{N}_k, \\ 0 & \text{otherwise.} \end{cases} \tag{4.5}$$

If we consider a set of real weights $\gamma_k$ in (4.4), we obtain the following combinations:

$$\begin{aligned} \sum_{k=1}^{N} \gamma_k (P^{\text{loc}}_{k,i|i})^{-1} \quad &= \quad \sum_{l=1}^{N} \sum_{k=1}^{N} \gamma_k [A]_{l,k} (P^{\text{ind}}_{l,i|i})^{-1} \\ &\quad - \Big( \sum_{l=1}^{N} \sum_{k=1}^{N} \gamma_k [A]_{l,k} - \sum_{k=1}^{N} \gamma_k \Big) \Pi_i^{-1}. \end{aligned} \tag{4.6}$$

Setting the weights in (4.6) such that $\sum_{k=1}^{N} \gamma_k [A]_{l,k} = 1$ for all $l$, and having the first term again on the right-hand side dominant, as the individual error covariance matrices are expected to get smaller with time, results in

$$\sum_{k=1}^{N} \gamma_k (P^{\text{loc}}_{k,i|i})^{-1} \approx \sum_{l=1}^{N} (P^{\text{ind}}_{l,i|i})^{-1}. \tag{4.7}$$

Equating the two approximations in (4.7) and (4.3) results in

$$P_{i|i}^{-1} \approx \sum_{k=1}^{N} \gamma_k (P^{\text{loc}}_{k,i|i})^{-1}. \tag{4.8}$$

We showed a direct approximation between the matrix $P_{i|i}$ and local available matrix $P^{\text{loc}}_{k,i|i}$. Therefore, we can trigger collecting measurements based on the trace of the local error covariance matrix $P^{\text{loc}}_{k,i|i}$, which is available in distributed Kalman filters. This is the motivation behind our triggering logic.

## 4.3 Event-Triggered Diffusion Extended Kalman Filter Algorithm

We consider the event-triggered distributed state estimation problem over a network of $N$ nodes distributed over some region in space indexed by $k \in \{0, \ldots, N-1\}$ as shown in Figure 4.1. Each node represents a sensor and an estimator. Also, we say that two nodes are connected if they can communicate directly with each other. Consider the following

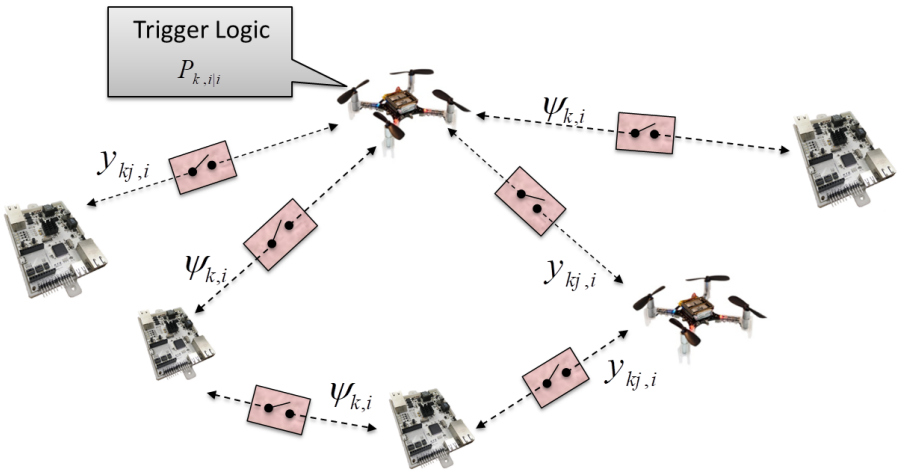



**Figure 4.1:** Every sensor node is running a distributed event-triggered state estimator to obtain the network state $x_{k,i|i}$. The trigger logic is based on monitoring local signal indicative of estimation error, thus linking the transmission and the sensing decisions to the estimation performance.

nonlinear time-varying system modeling our nonlinear application

$$\begin{aligned} x_{i+1} &= f_i(x_i) + G_i n_i \\ y_{kj,i} &= h_{k,i}(x_i) + v_{k,i} \end{aligned} \tag{4.9}$$

where $x_i \in \mathbb{R}^m$ is the state at time step $i$ and $y_{kj,i} \in \mathbb{R}^{L_k}$ is the measurement between node $k$ and its neighborhood node $j \in \mathcal{N}_k$ at time step $i$. Furthermore, the process noise $n_i$ and the measurement noise $v_{k,i}$ are assumed to be uncorrelated, and zero mean white Gaussian noises. The matrices $Q_i$ and $R_i$ are the process covariance and the measurement noise matrices at time step $i$, respectively. The state update and measurement functions are denoted by $f_i$ and $h_{k,i}$, respectively.

We denote the estimate of $x_i$ by $\hat{x}_{k,i|s}$ given the observations up to time $s$, where every node seeks to minimize the mean squared error $\mathbb{E}\|x_i - \hat{x}_{k,i|i}\|^2$. To handle the non-linearity in our model, we linearize (4.9) at a linearization point $z$, and apply the diffusion Kalman filtering algorithm [17, 141]. The resulting state update and measurement functions are shown in (4.10) and (4.11). The linearization clearly depends on $z$, and this point should be the best available local estimate of $x_i$.

$$\bar{F}_i(z) := \frac{\partial f_i(x)}{\partial x}\Big|_{x=z}, \tag{4.10}$$

$$\bar{H}_{k,i}(z) := \frac{\partial h_{k,i}(x)}{\partial x}\Big|_{x=z}. \tag{4.11}$$

Given a linearized model, we subsequently explain our event-triggered diffusion extended Kalman filter algorithm shown in Algorithm 4. One of the nodes is elected be-

**Algorithm 4:** Event-Triggered Diffusion Extended Kalman Filter

Start with $\hat{x}_{k,0|-1} = x_0$ and $P_{k,0|-1} = \Pi_0$ for all $k$, and at every time instant $i$, compute at every node $k$:

**if** $\mathsf{tr}(WP_{L,i|i-1}W^T) > \pi_{\max}$ **then**

**Step 1**: Measurement update:

$$\hat{H}_{kj,i} = \bar{H}_{j,i}(\hat{x}_{k,i|i-1}) \tag{4.12}$$

$$P^{-1}_{k,i|i} = P^{-1}_{k,i|i-1} + \sum_{j\in\mathcal{N}_k} \hat{H}^T_{kj,i} R^{-1}_i \hat{H}_{kj,i} \tag{4.13}$$

$$\Psi_{k,i} = \hat{x}_{k,i|i-1} + P_{k,i|i} \sum_{j\in\mathcal{N}_k} \hat{H}^T_{kj,i} R^{-1}_i [y_{kj,i} - h_{j,i}(\hat{x}_{k,i|i-1})] \tag{4.14}$$

**Step 2**: Diffusion update:

$$\hat{x}_{k,i|i} = \sum_{j\in\mathcal{N}_k} c_{kj} \Psi_{j,i} \tag{4.15}$$

**else**

**Step 3**: Propagation update:

$$\hat{x}_{k,i|i} = \hat{x}_{k,i|i-1} \tag{4.16}$$

$$P_{k,i|i} = P_{k,i|i-1} \tag{4.17}$$

**Step 4**: Time update:

$$\hat{x}_{k,i+1|i} = f_i(\hat{x}_{k,i|i}) \tag{4.18}$$

$$P_{k,i+1|i} = \bar{F}_i(\hat{x}_{k,i|i}) P_{k,i|i} \bar{F}^T_i(\hat{x}_{k,i|i}) + G_i Q_i G^T_i \tag{4.19}$$

forehand as a leader based on the accessibility of important measurements that facilitates reaching the best local estimate compared to the followers. We denote the leader node by subscript $L$. Choosing a good leader is crucial in saving energy; however, the election process based on the available estimates is out of the scope of this chapter. Algorithm 4 starts with the measurement update (step 1), where every node $k$ obtains a local estimate $\Psi_{k,i}$ at time step $i$. Next, information from the neighbors of node $k$ is diffused in a convex combination to produce a better new state estimate in step 2. The $c_{kj}$ elements represent the weights that are used by the diffusion algorithm to combine neighboring estimates. Step 1 and 2 are only executed if the trace of the required part of the leader matrix $P_{L,i|i-1}$ is more than the user-defined threshold $\pi_{\max}$. Explicitly, the triggering event is defined as $\mathsf{tr}(WP_{L,i|i-1}W^{\mathbb{T}}) > \pi_{\max}$ where $W$ is a weighting matrix to choose the required part of the $P_{L,i|i-1}$. If the triggering event is not satisfied, we do not take measurements $y_{kj,i}$, and we save the communication overheads in steps 1 and 2. Instead, we perform the propagation update (step 3), where every node considers the new estimates as the old available

ones $\hat{x}_{k,i|i} = \hat{x}_{k,i|i-1}$ and its corresponding local matrix $P_{k,i|i} = P_{k,i|i-1}$. Finally, every node performs the time update (step 4) in all cases.

We want to mention that the analysis in Section 4.2 shows the relationship of the global error covariance and the available local error covariance during the measurement update (step 1) in Algorithm 4. However, the diffusion update (step 2) does not take into account the recursions for these local error covariance matrices as it only combines the estimates of the neighbors without considering their local error covariance matrices. Also, exchanging the $P^{\text{loc}}_{k,i|i}$ between the neighbors to maintain the exact expected estimation error is a great overhead in sensor networks. Furthermore, the diffusion step decreases the estimation error, and it is important to have it included. Therefore, we continue with our modified version of $P^{\text{loc}}_{k,i|i}$, which we call *diffusion error covariance* matrix, and denote it by $P_{k,i|i}$ in Algorithm 4. Our event triggered algorithm holds also if there is no diffusion step, which is the case for the popular distributed Kalman filter. Removing the diffusion step is a special case of our algorithm with zeros in the diffusion weights $c_{kj}$.

## 4.4 Theoretical Analysis

Although the extended Kalman filter (EKF) has proven to work well in many nonlinear practical applications, its general convergence guarantees, even in the centralized version, can not be proved [142]. Thus, we limit our analysis to the linear case. We prove that event-triggered diffusion Kalman filter is an unbiased estimator. Next, we show the relationship between the diffusion error covariance matrix $P_{k,i|i}$ and the augmented error covariance. Consider the following linear time-varying system

$$x_{i+1} = F_i x_i + G_i n_i, \tag{4.20}$$
$$y_{kj,i} = H_{k,i} x_i + v_{k,i}. \tag{4.21}$$

**Lemma 4.4.1.** *The event-triggered diffusion Kalman filter is an unbiased estimator.*

*Proof.* The measurement update step in the linear case results in

$$P^{-1}_{k,i|i} = P^{-1}_{k,i|i-1} + \sum_{j \in \mathcal{N}_k} H^T_{j,i} R^{-1}_i H_{j,i}, \tag{4.22}$$
$$\Psi_{k,i} = \hat{x}_{k,i|i-1} + P_{k,i|i} \sum_{j \in \mathcal{N}_k} H^T_{j,i} R^{-1}_i [y_{kj,i} - H_{j,i} \hat{x}_{k,i|i-1}]. \tag{4.23}$$

The estimation error $\tilde{x}_{k,i|i-1}$ at the end of Algorithm 4 is defined and updated according to

$$\begin{aligned} \tilde{x}_{k,i|i-1} &:= x_i - \hat{x}_{k,i|i-1} \\ &= F_{i-1} \tilde{x}_{k,i-1|i-1} + G_{i-1} n_{i-1}. \end{aligned} \tag{4.24}$$

After defining the estimation error at the end of the measurement update by $\tilde{\Psi}_{k,i}$, we have

$$\begin{aligned}
\tilde{\Psi}_{k,i} &:= x_i - \Psi_{k,i} \\
&\overset{(4.23)}{=} \tilde{x}_{k,i|i-1} - P_{k,i|i} \sum_{j \in \mathcal{N}_k} H_{j,i}^T R_i^{-1} \Big[ y_{kj,i} - H_{j,i}(x_i - \tilde{x}_{k,i|i-1}) \Big] \\
&\overset{(4.21)}{=} \tilde{x}_{k,i|i-1} - P_{k,i|i} \sum_{j \in \mathcal{N}_k} H_{j,i}^T R_i^{-1} \Big[ H_{j,i} \tilde{x}_{k,i|i-1} + v_{j,i} \Big] \\
&= P_{k,i|i} \Big( P_{k,i|i}^{-1} - \sum_{j \in \mathcal{N}_k} H_{j,i}^T R_i^{-1} H_{j,i} \Big) \tilde{x}_{k,i|i-1} - P_{k,i|i} \sum_{j \in \mathcal{N}_k} H_{j,i}^T R_i^{-1} v_{j,i}.
\end{aligned} \tag{4.25}$$

Using (4.22) to simplify (4.25) results in

$$\tilde{\Psi}_{k,i} \overset{(4.22)}{=} P_{k,i|i} P_{k,i|i-1}^{-1} \tilde{x}_{k,i|i-1} - P_{k,i|i} \sum_{j \in \mathcal{N}_k} H_{j,i}^T R_i^{-1} v_{j,i}. \tag{4.26}$$

Applying the diffusion step (4.15) results in

$$\begin{aligned}
\tilde{x}_{k,i|i} &= \sum_{l \in \mathcal{N}_k} c_{lk} \tilde{\Psi}_{l,i} \\
&\overset{(4.26)}{=} \sum_{l \in \mathcal{N}_k} c_{lk} \Big[ P_{l,i|i} P_{l,i|i-1}^{-1} \tilde{x}_{l,i|i-1} - P_{l,i|i} \sum_{j \in \mathcal{N}_l} H_{j,i}^T R_i^{-1} v_{j,i} \Big].
\end{aligned} \tag{4.27}$$

Executing $M$ time updates and propagation updates before the $\mathsf{tr}(W P_{L,i|i} W^T)$ exceeds the threshold $\pi_{\max}$ results in

$$\begin{aligned}
\tilde{x}_{k,i+M+1|i+M} &\overset{(4.24)}{=} F_{i+M} \tilde{x}_{k,i+M|i+M} + G_{i+M} n_{i+M} \\
&\overset{(4.16)}{=} F_{i+M} \tilde{x}_{k,i+M|i+M-1} + G_{i+M} n_{i+M} \\
&\overset{(4.24)}{=} F_{i+M} \Big( F_{i+M-1} \tilde{x}_{k,i+M-1|i+M-1} + G_{i+M-1} n_{i+M-1} \Big) + G_{i+M} n_{i+M} \\
&= \prod_{j=0}^{M} F_{i+M-j} \tilde{x}_{k,i|i} + \sum_{l=1}^{M} \prod_{j=0}^{M-l} F_{i+M-j} G_{i+l-1} n_{i+l-1} + G_{i+M} n_{i+M}.
\end{aligned} \tag{4.28}$$

Inserting (4.27) in (4.28) results in

$$\begin{aligned}
\tilde{x}_{k,i+M+1|i+M} = &\prod_{j=0}^{M} F_{i+M-j} \left[ \sum_{l \in \mathcal{N}_k} c_{lk} \left[ P_{l,i|i} P_{l,i|i-1}^{-1} \tilde{x}_{l,i|i-1} - P_{l,i|i} \sum_{j \in \mathcal{N}_l} H_{j,i}^T R_i^{-1} v_{j,i} \right] \right] \\
&+ \sum_{l=1}^{M} \prod_{j=0}^{M-l} F_{i+M-j} G_{i+l-1} n_{i+l-1} + G_{i+M} n_{i+M}.
\end{aligned} \tag{4.29}$$

Taking the expectations of both sides of (4.29) results in the following recursion given that we have zero mean noises:

$$\mathbb{E}\tilde{x}_{k,i+M+1|i+M} = \prod_{j=0}^{M} F_{i+M-j} \sum_{l\in\mathcal{N}_k} c_{lk} P_{l,i|i} P_{l,i|i-1}^{-1} \mathbb{E}\tilde{x}_{l,i|i-1}. \tag{4.30}$$

Since $\mathbb{E}\tilde{x}_{l,0|-1} = 0$ as $\hat{x}_{l,0|-1} = 0$ and $\mathbb{E}x_0 = 0$ [141], we conclude that the event-triggered diffusion Kalman filter is an unbiased estimator. □

The Kronecker product is denoted by $\boxtimes$. The diffusion step (4.15) in Algorithm 4 combines the intermediate estimates from neighbors without combining the corresponding error covariance matrices, as we mentioned before. Thus, we need to find the new relationship between $P_{k,i|i}$, and the augmented error covariance. We define the augmented state-error vector $\tilde{\mathcal{X}}_{i|i}$ for the whole network as

$$\tilde{\mathcal{X}}_{i|i} := \left[\tilde{x}_{1,i|i}, \ldots, \tilde{x}_{N,i|i}\right]^T. \tag{4.31}$$

We further introduce the following block-diagonal matrices and $v_i$:

$$\begin{aligned}
\mathcal{H}_i &:= \text{diag}(H_{1,i}, \ldots, H_{N,i}),\\
\mathcal{P}_{i|i} &:= \text{diag}(P_{1,i|i}, \ldots, P_{N,i|i}),\\
\mathcal{P}_{i|i-1} &:= \text{diag}(P_{1,i|i-1}, \ldots, P_{N,i|i-1}),\\
v_i &:= \left[v_{1,i}, \ldots, v_{N,i}\right]^T.
\end{aligned}$$

**Lemma 4.4.2.** *The relationship between the error covariance $\mathcal{P}_{\tilde{x}|i} = \mathbb{E}\{\tilde{\mathcal{X}}_{i|i}\tilde{\mathcal{X}}_{i|i}^T\}$ of the augmented state and the diffusion error covariance $P_{k,i|i}$ is:*

$$\begin{aligned}
\mathcal{P}_{\tilde{x}|i+M+1} = {} & A_i \mathcal{P}_{\tilde{x}|i} A_i^T + B_i\big(\mathbb{1}\mathbb{1}^T \boxtimes Q_{i+M}\big) B_i^T \\
& + \sum_{l=1}^{M} D_{i,l}\big(\mathbb{1}\mathbb{1}^T \boxtimes Q_{i+l-1}\big) D_{i,l}^T + E_i R_{i+M+1} E_i^T,
\end{aligned} \tag{4.32}$$

*where*

$$\begin{aligned}
Z_i &:= \mathcal{C}^T \mathcal{P}_{i+M+1|i+M+1} \mathcal{P}_{i+M+1|i+M}^{-1},\\
A_i &:= Z_i\big(I_N \boxtimes \prod_{j=0}^{M} F_{i+M-j}\big),\\
B_i &:= Z_i\big(I_N \boxtimes G_{i+M}\big),\\
D_{i,l} &:= Z_i\big(I_N \boxtimes \prod_{j=0}^{M-l} F_{i+M-j}\big)\big(I_N \boxtimes G_{i+l-1}\big),\\
E_i &:= \mathcal{C}^T \mathcal{P}_{i+M+1|i+M+1} \mathcal{A}^T \mathcal{H}_{i+M+1}^T \mathcal{R}_{i+M+1}^{-1}.
\end{aligned} \tag{4.33}$$

*Proof.* Extending (4.27) to the augmented version results in

$$\begin{aligned}\tilde{\mathcal{X}}_{i+M+1|i+M+1} &\overset{(4.31)}{=} \left[\tilde{x}_{1,i+M+1|i+M+1}, \ldots, \tilde{x}_{N,i+M+1|i+M+1}\right]^T \\ &\overset{(4.27)}{=} \mathcal{C}^T \begin{bmatrix} P_{1,i+M+1|i+M+1} P^{-1}_{1,i+M+1|i+M} \tilde{x}_{1,i+M+1|i+M} \\ \vdots \\ P_{N,i+M+1|i+M+1} P^{-1}_{N,i+M+1|i+M} \tilde{x}_{N,i+M+1|i+M} \end{bmatrix} \\ &\quad - \mathcal{C}^T \mathcal{P}_{i|i} \mathcal{A}^T \begin{bmatrix} H_{1,i+M+1} R^{-1}_{1,i+M+1} v_{1,i+M+1} \\ \vdots \\ H_{N,i+M+1} R^{-1}_{1,i+M+1} v_{N,i+M+1} \end{bmatrix},\end{aligned}$$

or equivalently

$$\tilde{\mathcal{X}}_{i+M+1|i+M+1} = \mathcal{C}^T \mathcal{P}_{i+M+1|i+M+1} \Big( \mathcal{P}^{-1}_{i+M+1|i+M} \tilde{\mathcal{X}}_{i+M+1|i+M} - \mathcal{A}^T \mathcal{H}^T_{i+M+1} \mathcal{R}^{-1}_{i+M+1} v_{i+M+1} \Big), \tag{4.34}$$

with

$$\mathcal{C} := C \boxtimes I_m \qquad \mathcal{A} := A \boxtimes I_m, \tag{4.35}$$

where the element at row $l$ and column $k$ of diffusion matrix $C$ is $c_{lk}$ in (4.15). The $A$ is the adjacency matrix in (4.5). The size of $x_i$ in (4.9) is $m$. $I_m$ is the identity matrix with size $m \times m$. Similarly, extending (4.28) to the augmented version results in

$$\begin{aligned}\tilde{\mathcal{X}}_{i+M+1|i+M} &= \big(I_N \boxtimes \prod_{j=0}^{M} F_{i+M-j}\big) \tilde{\mathcal{X}}_{i|i} + \big(I_N \boxtimes G_{i+M}\big)\big(\mathbb{1} \boxtimes n_{i+M}\big) \\ &\quad + \sum_{l=1}^{M} \big(I_N \boxtimes \prod_{j=0}^{M-l} F_{i+M-j}\big)\big(I_N \boxtimes G_{i+l-1}\big)\big(\mathbb{1} \boxtimes n_{i+l-1}\big).\end{aligned} \tag{4.36}$$

Inserting (4.36) into (4.34) results in

$$\begin{aligned}\tilde{\mathcal{X}}_{i+M+1|i+M+1} &= A_i \tilde{\mathcal{X}}_{i|i} + B_i\big(\mathbb{1} \boxtimes n_{i+M}\big) \\ &\quad + \sum_{l=1}^{M} D_{i,l}\big(\mathbb{1} \boxtimes n_{i+l-1}\big) - E_i v_{i+M+1}.\end{aligned} \tag{4.37}$$

Taking the expectation of both sides of (4.37) results in (4.38) with the assumption that the state error $\tilde{\mathcal{X}}$, the time instances of modeling noise $n_i$, and the time instances of measurements noise $v_i$ are mutually independent [141].

$$\begin{aligned}\mathcal{P}_{\tilde{\mathcal{X}}|i+M+1} &= \mathbb{E}\Big\{\tilde{\mathcal{X}}_{i+M+1|i+M+1}\tilde{\mathcal{X}}^T_{i+M+1|i+M+1}\Big\} \\ &\overset{(4.37)}{=} A_i\mathbb{E}\Big\{\tilde{\mathcal{X}}_{i|i}\tilde{\mathcal{X}}^T_{i|i}\Big\}A_i^T + B_i\mathbb{E}\Big\{\big(\mathbb{1}\boxtimes n_{i+M}\big)\big(\mathbb{1}\boxtimes n_{i+M}\big)^T\Big\}B_i^T \\ &\quad + \sum_{l=1}^{M} D_{i,l}\mathbb{E}\Big\{\big(\mathbb{1}\boxtimes n_{i+l-1}\big)\big(\mathbb{1}\boxtimes n_{i+l-1}\big)^T\Big\}D_{i,l}^T \\ &\quad + E_i\mathbb{E}\Big\{v_{i+M+1}v_{i+M+1}^T\Big\}E_i^T. \end{aligned} \tag{4.38}$$

Applying the property of Kronecker products that $(A\boxtimes B)(C\boxtimes D)^T = (AC^T\boxtimes BD^T)$ results in

$$\begin{aligned}\mathcal{P}_{\tilde{\mathcal{X}}|i+M+1} =& A_i\mathcal{P}_{\tilde{\mathcal{X}}|i}A_i^T + B_i\big(\mathbb{1}\mathbb{1}^T\boxtimes Q_{i+M}\big)B_i^T \\ &+ \sum_{l=1}^{M} D_{i,l}\big(\mathbb{1}\mathbb{1}^T\boxtimes Q_{i+l-1}\big)D_{i,l}^T + E_iR_{i+M+1}E_i^T. \end{aligned} \tag{4.39}$$

The relation (4.39) relates the error covariance $\mathcal{P}_{\tilde{\mathcal{X}}|i}$ of the augmented state and the diffusion error covariance matrix $P_{k,i|i}$ and this concludes the proof. □

## 4.5 Evaluation

We start by describing our application to the considered localization and time synchronization problem. Next, we preform case studies to have a satisfying evaluation on the same testbed used in Section 3.5.

### 4.5.1 Application to Localization and Time Synchronization

One of the illustrative application to the event-triggering body of work is the distributed localization and time synchronization problem due to its need for excessive communication and computation overhead. Therefore, we pick this application to show the practicality of our proposed algorithm in real life. Our state vector consists of three-dimensional position vector $\boldsymbol{p}_{k,i}$, a clock time offset $o_{k,i}$, and a clock frequency bias $b_{k,i}$ for all nodes. We adopt a convention where both $o_{k,i}$ and $b_{k,i}$ are described with respect to the global time clock which is usually the clock of a leader node. Every node is interested in the state of the whole network. Thus, our state vector is $x_{k,i} = [\bar{x}_{1,i}, ..., \bar{x}_{N,i}]^T$, where $\bar{x}_{k,i} = \big[\boldsymbol{p}_{k,i}^T,\ o_{k,i},\ b_{k,i}\big]^T$.

The clock parameters evolve according to the first-order affine approximation of the dynamics $o_{k,i+1} = o_{k,i} + b_{k,i}\delta_i$ and $b_{k,i+1} = b_{k,i}$, where $\delta_i := t_{L,i+1} - t_{L,i}$ given that $t_{L,i}$ is the time according to the leader node, which is the global time. Therefore, we can write the update function as:

$$f_i(\bar{x}_{k,i}) \quad = \quad \begin{bmatrix} \boldsymbol{p}_{k,i} \\ o_{k,i} + b_{k,i}\delta_i \\ b_{k,i} \end{bmatrix}. \tag{4.40}$$

Our framework supports three types of measurements which are distinguished by the number of messages exchanged between a pair of nodes. The measurement vector sent from node $j \in \mathcal{N}_k$ to node $k$ has the form $y_{kj,i} = [d_{kj,i}, r_{kj,i}, \Gamma_{kj,i}]^T$, where, $d_{kj,i}$, represents the counter difference at time step $i$ which is the difference between the clock offsets of the two nodes $k$ and $j$. On the other hand, $r_{kj,i}$ represents a noisy measurement due to frequency bias discrepancies between $k$ and $j$ which is formally represented by single-sided two-way range. Finally, $\Gamma_{kj,i}$ is another distance measurement between nodes $k$ and $j$ based on a trio of messages between nodes at time index $i$. This is a more accurate estimate than $r_{kj,t}$ due to mitigation of frequency bias errors from the additional message. It is formally called double-sided two-way range. For more details about the three types of measurements, we refer the reader to Section 3.3.1.

We want to note that the subset of these measurements may be used rather than the full set, i.e., we can have experiments involving just $r_{kj,i}$, $\Gamma_{kj,i}$, or $d_{kj,i}$. The response time duration between the first pair of timestamps is denoted by $T_{RSP1}$. Our measurement function is:

$$h_{k,i}(\bar{x}_{j,i}) = \begin{bmatrix} (o_{j,i} - o_{k,i}) + \frac{1}{c}\left\|\boldsymbol{p}_{j,i} - \boldsymbol{p}_{k,i}\right\|_2 \\ \left\|\boldsymbol{p}_{j,i} - \boldsymbol{p}_{k,i}\right\|_2 + \frac{c}{2}\left(b_{j,i} - b_{k,i}\right) T_{RSP1} \\ \left\|\boldsymbol{p}_{j,i} - \boldsymbol{p}_{k,i}\right\|_2 + c\tilde{\Gamma}_{kj,i} \end{bmatrix} \tag{4.41}$$

## 4.5.2 Experiments

We consider mainly the communication overhead and its associated accuracy on different network topologies to show the effectiveness of our proposed algorithm. We are concerned in applying the triggering logic based on the expected estimation error of the location of the mobile node which is a CrazyFlie. The mobile node is elected as the leader and its diffusion error covariance matrix is $P_{L,i|i}$ in Algorithm 4.

To give the reader an intuition of how we are going to evaluate our algorithm, we show the results of running a portion of the experiments in Figure 4.2. The mobile node is flying at different speeds in our lab while trying to save computation and communication resources. The threshold $\pi_{\max}$ is set to be $4m$. The second sub-figure in Figure 4.2 shows the behavior of $\mathsf{tr}(WP_{L,i|i}W^T)$. All nodes are only executing the time update step when $\mathsf{tr}(WP_{L,i|i}W^T)$ is less than $\pi_{\max}$. As a consequence, we are decreasing the spent power in measurements, message exchange, and computation overheads. The effect on the estimated localization error of the mobile node can be seen clearly in the first sub-figure of Figure 4.2. Once, $\mathsf{tr}(WP_{L,i|i}W^T)$ reaches the threshold $\pi_{\max}$, all nodes are triggered to start measuring and to exchange messages to decrease $\mathsf{tr}(WP_{L,i|i}W^T)$ back to the allowed range. The third sub-figure in Figure 4.2 demonstrates the time where the

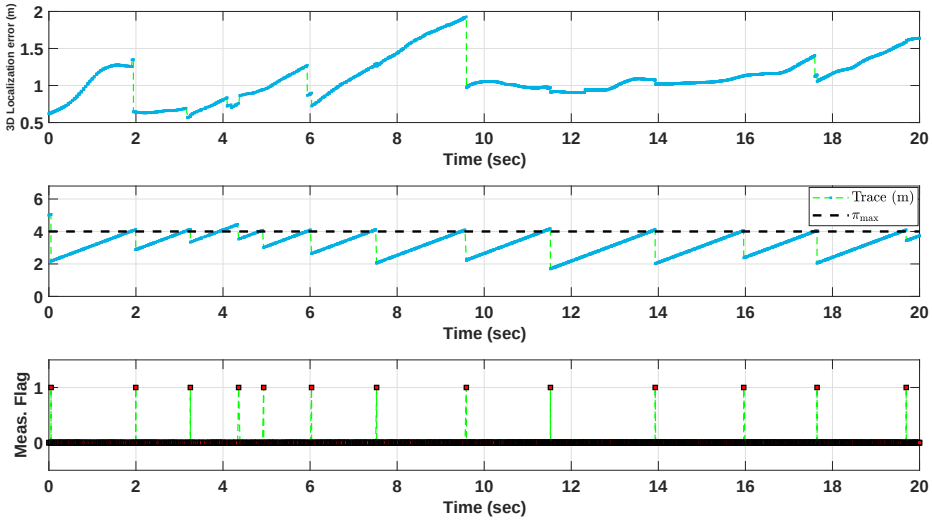



**Figure 4.2:** A snapshot of $20$ seconds of our experiments. The threshold is set to $4m$. The 3D localization error and trace value $\mathtt{tr}(WP_{L,i|i}W^T)$ are shown in the first and second sub-figures, respectively. The measurement and diffusion flags are the same and shown in the third sub-figure where, a value of $1$ indicates of executing the step, while $0$ means skipping the step at the corresponding time instance. Time Update step is happening all the time.

measurement update step is executed at all nodes. For instance, we can see that the measurement update step happens at the time instances $0.1, 2.7, 4.4, 5.1, 7.0, 10.1, 12.9, 16.4$ so that we can notice the decrease in the localization error in the first sub-figure at the same time instances. Similarly, the diffusion update step happens just after the measurement update step. The time update step is happening all the time. The CrazyFlie is flown through our lab over four different sessions. Each session was conducted on a different day with a different number of students in the room. Also, the path of the CrazyFlie was a random walk for each day. We repeated the experiments while setting a different threshold, then calculate the number of shared messages between nodes and the localization error reported by the motion capture system.

#### 4.5.2.1 Fully-Connected Network Case Study

We are going to illustrate the effectiveness of our triggering algorithm by showing the amount of communication-saving and the associated localization error by applying the algorithm over a fully-connected network.

**Communication Analysis:** Figure 4.4a shows the effect of changing the threshold value $\pi_{\max}$ on the percentage of the saved message for a fully connected network. Zero threshold refers to the case of sending all the messages and running the three steps of the algorithm in a normal fashion. In other words, it corresponds to sending $1,975,632$

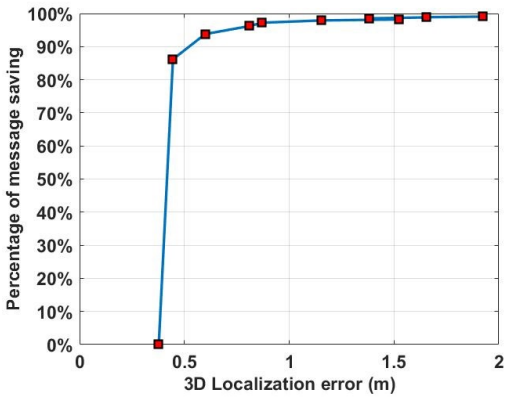
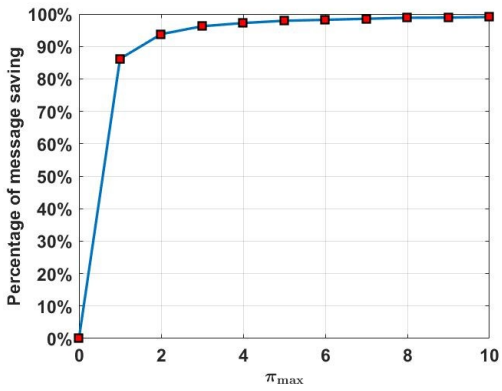



**Figure 4.3:** The trade-off between the communication overhead saving and the mean 3D localization error of the CrazyFlie for a fully connected network.

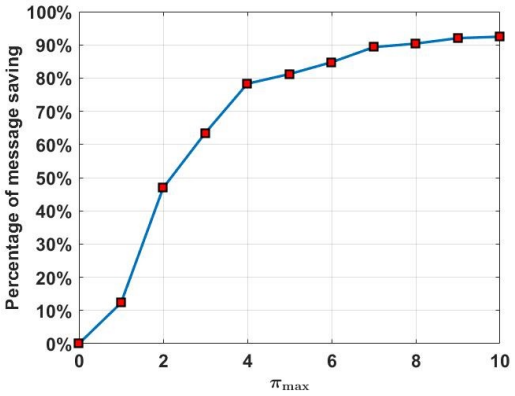
messages. Remember that every message already corresponds to single-sided or double-sided two-way range measurements. Thus, the reported number of messages should be multiplied by two or three, depending on the message ranging type. Setting the threshold to $1m$ leads to saving about $86.2\%$ of the overall number of messages. Namely, saving $1,702,797$ messages. Also, a threshold of $5m$ ends up with saving $98\%$ of the total number of messages.

**Accuracy Analysis:** We conduct the following study to see what is the trade-off between the 3D-localization error with each threshold value $\pi_{\max}$. The error plot in Figure 4.4c summarizes our results of this case study. The red rectangles correspond to the mean value of the localization error, while the vertical lines represent the standard deviation around that mean value. At threshold $\pi_{\max} = 0$, we are not saving any resources, and we achieve $0.377m$ mean localization error with a standard deviation of $0.195m$. While, $\pi_{\max} = 5m$ results in a $1.1545m$ mean error with a standard deviation of $0.71m$. Finally, $\pi_{\max} = 10m$ achieves a $1.923m$ mean error with a standard deviation of $0.790m$. It is up to the user to set the appropriate threshold based on his need.

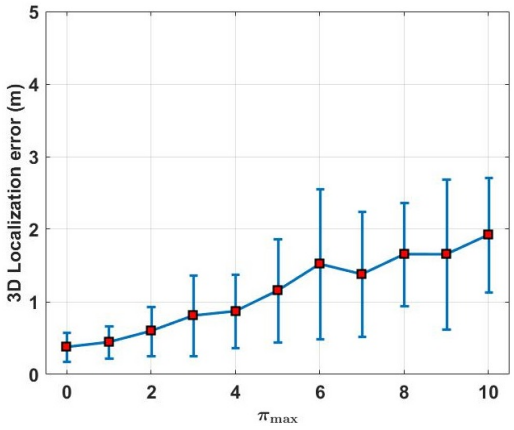
Our algorithm restricts the amount of processing, sensing, and communication. Such a chosen restriction dramatically reduces the amount of communication overhead in the network, but potentially results in reduced network performance. We analyze the trade-off between the number of messages sent in the wireless network and the estimation algorithms performance. Figure 4.3 shows the trade-off between the communication overhead and the mean 3D localization error. Interestingly, saving $86.2\%$ of communication overhead leads to $16.57\%$ increase in the localization error. This has been calculated by considering the mean localization error plus the standard deviation at a threshold $0$ and $1$.



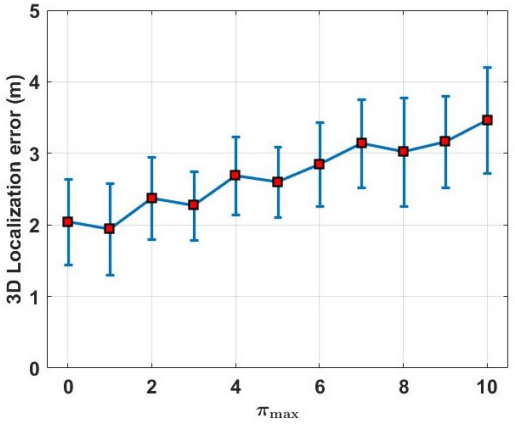

**(a)** Effect of changing the threshold value $\pi_{\max}$ on the percentage of the saved message for a fully connected network.




**(b)** Effect of changing the threshold value $\pi_{\max}$ on the percentage of the saved message for a partially connected network.




**(c)** Effect of changing the threshold value $\pi_{\max}$ on the 3D localization error of the CrazyFlie for a fully connected network.




**(d)** Effect of changing the threshold value $\pi_{\max}$ on the 3D localization error of the CrazyFlie for a partially connected network.

**Figure 4.4:** Effect of changing the threshold value $\pi_{\max}$ on a fully connected network and partially connected one.

#### 4.5.2.2 Partially Connected Network Case Study

We considered another case study where every node of the nine nodes is connected to only four neighbors instead of eight neighbors in the previous case study. Again, we are going to analyze the communication saving and the associated localization error. Figure 4.4b summarizes the results. Interestingly, setting the threshold to $5m$ leads to saving about $81.2\%$ of the overall number of messages. Also, a threshold of $\pi_{\max} = 9m$ ends up saving $92\%$ of the total number of messages. The error plot in Figure 4.4d summarizes our results of this case study. Again, the red rectangles correspond to the mean value of the localization error, while the vertical lines represent the standard deviation around that mean value. At threshold $\pi_{\max} = 0m$, we are not saving any resources, and we could achieve $2m$ mean localization error with a standard deviation of $0.6m$, where the network is partially connected as described before. While $\pi_{\max} = 5m$ results in a $2.6m$

mean error with a standard deviation of $0.49m$. Finally, $\pi_{\max} = 10m$ achieves a $3.46m$ mean error with a standard deviation of $0.74m$.

## 4.6 Conclusions

We investigated the energy aware-aspect of the distributed estimation problem for a multi-sensor system with event-triggered processing schedules. More specifically, we propose the event-triggered diffusion distributed Kalman filter for wireless networks. Our algorithm is the first algorithm that temporarily shuts down the measurement and diffusion steps in the diffusion Kalman filter. We have demonstrated our new algorithm on the distributed localization and time synchronization application. Several experiments using real, custom ultra-wideband wireless anchor nodes and mobile quadrotor node were conducted, and they indicate that the proposed algorithm is reliable in terms of performance, and efficient in the use of computational and communication resources. Future directions will deal with testing over a large-scale system.

# 5 Distributed Secure State Estimation Using Reachability Analysis

The Office of Science and Technology Policy (OSTP) assigns a high priority to cyber-physical systems (CPS) security since the recent attacks launched in the cyber domain led to calamitous consensuses during the past decades [143]. Therefore, secure state estimation has attracted attention due to the rise of new security vulnerabilities and attacks at the sensor level with potentially life-threatening consequences in the last decade. For instance, the Maroochy Water Breach [4] made it possible to attack the underlying infrastructure at Maroochy Water Services in Queensland. Also, one popular attack is the Stuxnet attack on Supervisory Control and Data Acquisition (SCADA) systems, which are used in industrial process control [5, 6].

After proposing distributed estimation algorithms in Chapters 3 and 4, we attach the utmost importance to the security of the distributed estimation algorithms. We provide a solution for attacks on the sensor. More specifically, we propose an approach for distributed linear secure state estimation in the presence of measurement noise and modeling errors. By combining the diffusion Kalman filter [33] with reachability analysis [34], we provide a new algorithm for distributed secure state estimation between a network of nodes. We apply the proposed algorithm to a localization problem of a rotating target. This chapter is based on our publication [24, 32] and is organized as follows. We review the related work in Section 5.1. After we introduce the problem and the proposed solution in Section 5.2, the secure measurement update is presented in Section 5.3 and secure diffusion in Section 5.4. The applicability of the algorithm is demonstrated in Section 5.5. This is followed by a discussion of the algorithm and a conclusion in Section 5.6.

## 5.1 Related Work

We review the different techniques that have addressed the problem of secure state estimation against sensor attacks in centralized dynamical systems. Fawzi et al. show the impossibility of accurately reconstructing the state of a system if more than half of the sensors are attacked [144]. The presence of process and measurements noise offers attackers an additional possibility to tamper with CPS sensors, thereby making the detection task more challenging. Another work in [145] uses brute force search for studying the observability of linear systems under adversarial attacks; however, this approach is

not applicable to large-scale systems. A practical solution is proposed in [146] that considers jitter, latency and synchronization errors. Graph-theoretic conditions for the detectability of attacks for a noiseless system are shown in [147]. Also, a measure of the stealthiness of attacks in stochastic control systems is proposed in [148, 149].

The replay attack is commonly defined as observing and recording sensor readings and replying them afterward while carrying out an attack. This kind of attack is considered in [150] where all sensors were attacked and the attacker does not have any model knowledge. Another work considered a stochastic game for detecting replay attacks [151]. Also, a solution for denial of service attacks under Gaussian noise is proposed in [152]. Furthermore, false data injection is solved by proposing an ellipsoidal algorithm where the strategy of the attacker is formulated as a constrained control problem [153].

Next, we review the related work in the distributed setting. Distributed processing mitigates the computation load by getting rid of the fusion center in centralized fusion and estimation. Pasqualetti et al. [154] propose a fully decentralized solution for attack identification. However, they only consider noiseless systems. Distributed secure controllers based on a virtual fractional dynamic surface are designed in [155]. Also, a consensus-based protocol is utilized for distributed secure state estimation in [156].

## 5.2 Distributed Secure State Estimation and Proposed Solution

We target to estimate the full state vector of a system in a distributed fashion by observing physical signals through sensory devices which are under attack. The distributed processing aims to get rid of the required fusion center in the centralized version. In order to deviate the system from its correct operation, an attacker endeavors to either a) physically attack the sensor environment, b) attack the sensor hardware, c) break the communication links in the CPS, or d) modify the sensor readings (e.g., by delaying packets in time-of-flight based localization). We first discuss our threat model and preliminaries followed by a mathematical formulation of the distributed secure state estimation problem.

### 5.2.1 Threat Model

We consider attackers that directly compromise the readings of various sensor groups and man-in-the-middle attackers that endeavor to modify the data transfer between sensors, as shown in Figure 5.1.a. Our assumptions are:

- The adversary can commit to unbounded attack values.
- The adversary additionally has no prior knowledge of the system parameters.
- The adversary can corrupt all sensors and has unlimited computational power.
- The selection of attacked sensors is unknown to the system and can change dynamically over time.

### 5.2.2 Preliminaries

We state some preliminaries around the proposed solution.

**Definition 5.2.1.** ***(Zonotope)*** *A zonotope $\mathcal{Z} = \langle c, G \rangle \subset \mathbb{R}^n$ consists of a center $c \in \mathbb{R}^n$ and generator matrix $G \in \mathbb{R}^{n \times e}$. We define $G$ as $e$ generators $g^{(i)} \in \mathbb{R}^n$ $(i = \{1, .., e\})$, where $G = [g^1, ..., g^e]$ [34]. A zonotope is a set*

$$\mathcal{Z} = \Big\{ c + \sum_{i=1}^{e} \beta_i g^{(i)} \Big| -1 \leq \beta_i \leq 1 \Big\}. \tag{5.1}$$

□

Given two zonotopes $\mathcal{Z}_1 = \langle c_1, G_1 \rangle$ and $\mathcal{Z}_2 = \langle c_2, G_2 \rangle$, we define [34]:

1. Minkowski sum:
$$\mathcal{Z}_1 \boxplus \mathcal{Z}_2 = \Big\langle c_1 + c_2, [G_1, G_2] \Big\rangle \tag{5.2}$$
2. Linear map:
$$L\mathcal{Z}_1 = \Big\langle Lc_1, LG_1 \Big\rangle \tag{5.3}$$

We define the reachable set as the set of possible solution $x_i$ which can be reached at each time step. In this work, reachable sets are represented by zonotopes due to their favorable computational complexity as discussed in [34].

### 5.2.3 System Model

We consider a set of $N$ nodes indexed by $k \in \{0, \ldots, N-1\}$ distributed geographically over some region. The neighborhood of a node $k$ is denoted by the set $\mathcal{N}_k$ which contains the nodes connected to node $k$; the size of $\mathcal{N}_k$ is $m_k$. Every node is interested in estimating the state $\tilde{x}$ of the network securely. We assume that network connectivity is fixed with time and the measurements trace follows a predefined sequence. We consider a discrete-time, linear system model with pairwise measurements taken per time step $i$.

$$\begin{aligned} \tilde{x}_{k,i+1} &= \tilde{F}_i \tilde{x}_{k,i} + \tilde{n}_{k,i} \\ y_{kj,i} &= \tilde{H}_{kj,i} \tilde{x}_{k,i} + \tilde{v}_{k,i} + a_{kj,i}\,, \end{aligned} \tag{5.4}$$

where $\tilde{x}_{k,i} \in \mathbb{R}^{n_k}$ is the state of node $k$ at time $i \in \mathbb{N}$ and $y_{kj,i} \in \mathbb{R}^{m_k}$ is the measurement sent to node $k$ from the neighboring node $j \in \mathcal{N}_k$. The process and measurement noises are denoted by $\tilde{n}_{k,i}$ and $\tilde{v}_{k,i}$, respectively. All vectors and matrices are real-valued and have proper dimensions. The attack vector $a_{kj,i}$ is a vector which models how an attacker corrupts the sensor measurements between node $k$ and node $j$ at time $i$. A non-zero element in the vector $a$ corresponds to the attacked values on the corresponding sensor, otherwise the measurement is not attacked. Thereby, the additive attack vector $a_{kj,i}$ can account for both a malicious node $k$ and a corrupted link $kj$. Moreover, the attack values can be constant or time-varying. The modeling noise $\tilde{n}_{k,i}$ and measurement noise $\tilde{v}_{k,i}$

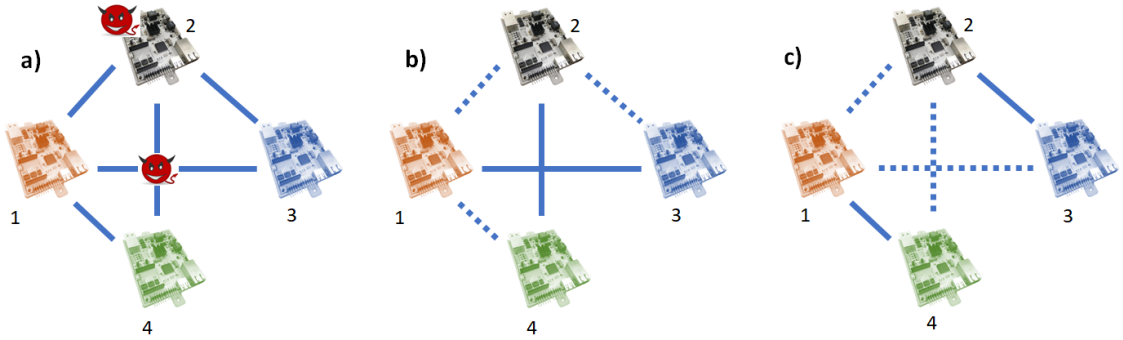



**Figure 5.1:** Attacks are on links and sensors as shown in sub-figure a. Links are divided into passive (dashed) and active (bold) links at each time step. Active links carry a measurement between two nodes. The two sub-figures b and c show the active and passive links at two time steps.

are assumed to be unknown but bounded by zonotopes: $\tilde{n}_{k,i} \in I_{Q_i} = \langle 0, Q_i \rangle$ and $\tilde{v}_{k,i} \in I_{R_i} = \langle 0, R_i \rangle$.

### 5.2.4 Proposed Solution

Our proposed solution is based mainly on a diffusion Kalman filter [157] - which was studied in Chapters 3 and 4 - integrated within a secure state estimation concept [27] and combined with reachability analysis [34]. The original non-secure diffusion Kalman filter consists of three main steps, namely, measurement update, time update, and diffusion update. By ensuring the security of all the steps of the diffusion Kalman filter, we can obtain a secure distributed state estimator. Thus, our solution consists of:

1. Secure measurement update: Every node shares its measurements with its neighbors and does some internal processing. Protecting the measurement update is achieved by extending secure state estimation in [27] for the distributed case.

2. Secure diffusion: Every node shares its network state estimate with its neighbors and combines the estimates in a convex way. We protect the diffusion step by accepting the shared estimate if and only if it is within the accepted region of the reachability analysis.

3. Time update: Every node updates its state, which is trivially protected as no data is exchanged.

We will describe the protection of the measurement update and secure diffusion in more detail in the following sections.

## 5.3 Secure Measurement Update

Typically, sensor nodes have one radio for one type of measurement. Thus, one link is activated per node at each time step for performing measurements (like calculating the pairwise distances between nodes). For instance, node$_1$ performs measurement with node$_3$ at $t_1$ in Figure 5.1.b. Then, it will perform measurement with node$_4$ at time $t_2$ in Figure 5.1.c. At the same time, node$_2$ performs measurements with node$_4$ then with node$_3$, as shown in Figures 5.1.b and 5.1.c. We have five links with five attacks to be mitigated in Figure 5.1 according to (5.4). Related work typically considers collecting the measurements from all the links and performs the estimation at once. This results in a very complex problem with the number of unknown attacks to be equal to the number of links, which is usually much more than the number of nodes.

In contrast, we propose solving the problem from another perspective: We define the links between node$_2$ and node$_4$ and between node$_1$ and node$_3$ in Figure 5.1.b as "active" links (bold) which carry measurements. The other links (dashed) in Figure 5.1.b are called "passive" links. The time horizon of the measurements is divided into time steps where each node performs a measurement at each time step. We do not estimate the attack on passive links at the corresponding time steps. So, for example at time $t_1$ in Figure 5.1.b, why do we trouble ourselves to estimate $a_{23,1}$? At each time instant $i$, each node performs measurement with one neighbor. Note that due to this principle, it is possible to denote the attack $a_{kj,i}$ only by $a_{k,i}$, i.e., neighbor $j$ is uniquely defined by time step $i$ and node $k$. With this idea, the number of attacks at each time instant equals the number of nodes, instead of the number of links. Therefore, we drastically simplify the secure state estimation problem. All the links for each node are modeled using one variable at different time steps. This even works while attacking all the links with time-varying attacks as we will show in Section 5.5. This concept is repeated at each node $k$ as every node is interested in estimating the network state. In short, we consider the attack variables attached to nodes instead of links at different time steps.

More formally, we utilize our idea to change the general model in (5.4) by including the attack value in the state of the node initiating the measurement. Following this procedure, the state of node $k$ is extended to $x_{k,i} = [\tilde{x}_{k,i}^{\mathrm{T}}, a_{kj,i}^{\mathrm{T}}]^{\mathrm{T}}$ yielding a modified system model

$$\begin{aligned} x_{k,i+1} &= F_i x_{k,i} + n_{k,i}, \\ y_{kj,i} &= H_{kj,i} x_{k,i} + v_{k,i}. \end{aligned} \tag{5.5}$$

**Proposition 5.3.1.** *System model* (5.4) *is equivalent to system model* (5.5) *under the assumption that the measurements processing is done for each node with one neighbor at each time step for a network with pairwise measurements.*

**Proof:** We get rid of $j$ in $a_{kj,i}$ in (5.4) by choosing $x_{k,i} = [\tilde{x}_{k,i}^{\mathrm{T}}, a_{kj,i}^{\mathrm{T}}]^{\mathrm{T}}$ where the variations in $j$ would be presented by changing the time step $i$, i.e., the couple $i$ and $k$ uniquely defines the neighbor $j$ because node $k$ can only communicate with one node at time $i$. The matrices $\tilde{F}_i$ and $\tilde{H}_{kj,i}$ are changed accordingly to matrices $F_i$ and $H_{kj,i}$, re-

spectively. We assume that the network connectivity is fixed with time and the sequence in the measurement trace is predefined. □

A valid question would be how to model the time-evolution for time-varying attacks $a_{k,i}$? We choose to set $a_{k,i+1} = a_{k,i} + n_{k,a_i}$, and the variance of the modeling noise of the attack entries $n_{k,a_i}$ accounts for the time-varying aspect of the attack and for changing the links between time steps. This lets us move from specifying specific dynamics for the attacks. Another question would be how to obtain bounds on $n_{k,a_i}$. Our proposed solution is to use reachability analysis [34] and only accept the measurements that let the state stay inside the expected reachable set. Also, it should be noted that, since high attack values can be easily detected by threshold methods (e.g., if the reported distance is far beyond the range of the area where the network is employed), critical attacks can occur within a limited interval, which can be represented by the modeling covariance. This concept - of modeling a time-varying signal using the modeling noise - has been applied in localization in Section 3.3 where we use a stationary model for process updates of a flying quadrotor. As long as the quadrotor moves in the range of the modeling noise, we would be able to localize it correctly.

To move forward with utilizing reachability analysis in our solution, we need to define the following sets:

**Definition 5.3.2.** ***(Predicted State Set)*** *Given system* (5.5) *with initial state* $x_0 \in \langle c_0, G_0 \rangle$*, the reachable state set* $\mathcal{Z}_{k,i}$ *of node* $k$ *is defined as the set of all possible solutions* $x_i$ *which can be reached given* $x_{i-1}$*.* $I_{Q_i}$ *is the zonotope which bounds modeling noise [158, p.4]*

$$\mathcal{Z}_{k,i} = F_i \mathcal{Z}_{k,i-1} \boxplus I_{Q_i}. \tag{5.6}$$

□

**Definition 5.3.3.** ***(Measurement State Set)*** *Given system* (5.5)*, the measurement state set* $\mathcal{S}_{kj,i}$ *of node* $k$ *is defined as the set of all possible solutions* $x_i$ *which can be reached given* $y_{kj,i}$ *and* $v_i$*. This measurement set is a strip [158, p.4]:*

$$\mathcal{S}_{kj,i} = \left\{ x_i \middle| |H_{kj,i} x_i - y_{kj,i}| \leq R_{j,i} \right\}. \tag{5.7}$$

□

**Definition 5.3.4.** ***(Corrected State Set)*** *Given system* (5.5) *with initial state* $x_0 \in \langle c_0, G_0 \rangle$*, the reachable corrected state set* $\mathcal{Z}_{k,\psi_i}$ *of node* $k$ *is defined as the intersection between* $\mathcal{Z}_{k,i}$ *and* $\mathcal{S}_{kj,i}$ *[158, p.4]:*

$$\mathcal{Z}_{k,\psi_i} = \mathcal{Z}_{k,i} \cap \mathcal{S}_{kj,i}. \tag{5.8}$$

□

We denote the predicted and the filtered estimates of $x_i$ at time step $i$ obtained by node $k$ as $\hat{x}_{k,i+1|i}$ and $\hat{x}_{k,i|i}$, respectively. The main algorithm is summarized in Algorithm 5. We start with zonotope $\mathcal{Z}_0 = \langle c_0, G_0 \rangle$ where the center $c_0$ equals the expected initial estimates $x_0$.

Every measurement $y_{kj,i}$ restricts the state to be in a strip $\mathcal{S}_{kj,i}$ as shown in (5.7). Every node corrects the reachable set (zonotope $\mathcal{Z}_{k,i|i-1}$) by determining the set of consistent states with the model and the measurements received from each neighbor. Therefore, we need to find the intersection between the family of strips in (5.7) and the zonotope $\mathcal{Z}_{k,i|i-1}$. This results in calculating the corrected over-approximated zonotope $\mathcal{Z}_{k,\psi_i}$ for node $k$. We extend the work [159] in the following theorem to find the required intersection.

**Theorem 5.3.5.** *The zonotope $\mathcal{Z} = \langle c_0, G_0 \rangle$, the family of $m$ strips $\mathcal{S}_{kj,i}$ (5.7), and the vectors $\lambda_{kj,i} \in \mathbb{R}^n$ are given. The intersection between the zonotope and the strips can be over-approximated by a zonotope $\mathcal{Z}_{k,\psi_i} = \langle c(\lambda), G(\lambda) \rangle$, where*

$$c(\lambda) = c_0 + \sum_{j \in \mathcal{N}_k} \lambda_{kj,i}(y_j - H_{kj,i}c_0) \tag{5.9}$$

$$G(\lambda) = \Big[(I - \sum_{j \in \mathcal{N}_k} \lambda_{kj,i} H_{kj,i})G_0,\ \lambda_{k1,i}R_{1,i}, ..., \lambda_{km_k,i}R_{m_k,i}\Big]. \tag{5.10}$$

**Proof:** Let $x \in (\mathcal{Z} \cap \mathcal{S}_{k1} \cap ... \cap \mathcal{S}_{km})$, then there is a $z$, where

$$x = c_0 + G_0 z, \tag{5.11}$$

where $G_0$ has full rank. We would like to highlight that the over-approximation comes from choosing $x \in (\mathcal{Z} \cap \mathcal{S}_{k1,i} \cap ... \cap \mathcal{S}_{km,i})$. Then, adding and subtracting $\sum_{j \in \mathcal{N}_k} \lambda_{kj,i} H_{kj,i} G_0 z$ results in

$$x = c_0 + \sum_{j \in \mathcal{N}_k} \lambda_{kj,i} H_{kj,i} G_0 z + (I - \sum_{j \in \mathcal{N}_k} \lambda_{kj,i} H_{kj,i}) G_0 z. \tag{5.12}$$

Given that $x$ is inside the intersection of the zonotope $\mathcal{Z}$ and the family of strips, then $x \in \mathcal{S}_{kj,i}, \forall j \in \mathcal{N}_k$, i.e., there exists a $b_j \in [-1, 1]$ in (5.7) for the $j^{th}$ strip so that:

$$H_{kj,i}x - y_j = R_{j,i}b_j. \tag{5.13}$$

Inserting (5.11) in (5.13) results in

$$H_{kj,i}G_0 z = y_j - H_{kj,i}c_0 + R_{j,i}b_j. \tag{5.14}$$

Inserting (5.14) in (5.12) results in

$$
\begin{aligned}
x = \; & c_0 + \sum_{j \in \mathcal{N}_k} \lambda_{kj,i}(y_j - H_{kj,i}c_0 + R_{j,i}b_j) + (I - \sum_{j \in \mathcal{N}_k} \lambda_{kj,i}H_{kj,i})G_0 z \\
= \; & c_0 + \sum_{j \in \mathcal{N}_k} \lambda_{kj,i}(y_j - H_{kj,i}c_0) + (I - \sum_{j \in \mathcal{N}_k} \lambda_{kj,i}H_{kj,i})G_0 z + \sum_{j \in \mathcal{N}_k} \lambda_{kj,i}R_{j,i}b_j \\
= \; & \underbrace{c_0 + \sum_{j \in \mathcal{N}_k} \lambda_{kj,i}(y_j - H_{kj,i}c_0)}_{c(\lambda)} \\
& + \underbrace{\Big[(I - \sum_{j \in \mathcal{N}_k} \lambda_{kj,i}H_{kj,i})G_0, \lambda_{k1,i}R_{1,i}, ..., \lambda_{km_k,i}R_{m_k,i}\Big]}_{G(\lambda)} \begin{bmatrix} z \\ b_1 \\ .. \\ b_{m_k} \end{bmatrix} \\
= \; & c(\lambda) + G(\lambda) \begin{bmatrix} z \\ b_1 \\ .. \\ b_{m_k} \end{bmatrix} \qquad \square
\end{aligned}
$$

**Choosing an appropriate $\lambda$:** In order to find an appropriate over-approximation for the intersection of a zonotope with a family of strips, $\lambda$ is typically chosen to minimize an approximation criterion. The authors in [159] proposed two approaches for intersecting a zonotope with a strip. The first approach is a segment-minimization approach which has a low computational complexity by minimizing the Frobenius norm of $G(\lambda)$. The second approach provides a better approximation; it is a volume-minimizing approach and requires solving a convex optimization problem.

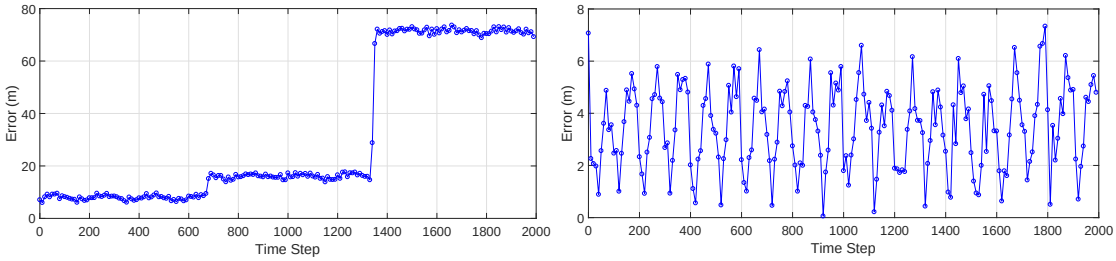
However, if we take a careful look at (5.15), which is the measurement update equation that propagates the measurement effect into the estimates in the diffusion Kalman filter [157], we can find that the structure is very similar to our formula (5.9) which finds the new center of the intersection of a zonotope with a family of strips.

$$
\psi_{k,i} \;=\; \hat{x}_{k,i|i-1} + P_{k,i|i} \sum_{j \in \mathcal{N}_k} H_{kj,i}^T {R_{j,i}}^{-1} [y_{kj,i} - H_{kj,i}\hat{x}_{k,i|i-1}] \tag{5.15}
$$

Thus, we choose to use the $\lambda$ that is aligned with diffusion Kalman filter theory at each node $k$ as shown in step 1 of Algorithm 5:

$$
\lambda_{kj,i} = P_{k,i|i} H_{kj,i}^T {R_{j,i}}^{-1} \tag{5.16}
$$

We choose the center of the reachable set on every node $k$ as the estimate $\psi_{k,i}$. Also, as the size of the generators is increasing in each step by doing the previous measurement




**(a)** Insecure **(b)** Secure

**Figure 5.2:** Localization error at one node of the rotating target where all the measurements are under attack. The measurements only are under attack while the diffusion step is not under attack. Attacks are generated from uniform, normal and Pareto pseudo-random distributions, as shown in (5.13). Y-scales are different in Figures (a) and (b).

update, we reduce the order of the corrected zonotope $\mathcal{Z}_{k,\psi_i} = \langle \psi_{k,i}, G_{k,\psi_i} \rangle$ order by the method from [160, p.7]. This illustrates Step 1 of Algorithm 5.

## 5.4 Secure Diffusion

Every node shares its own local estimate $\psi_{k,i}$ with its neighbors in the diffusion step [157]. Then every node averages the shared estimates $\psi_{j,i}$ from the neighbors to achieve a better estimate of the system state. The averaging is based on some weights $w_{kj,i}$ for each neighbor $j$ [157]. However, these shares may be under attack. Thus, we make use of reachability analysis to protect against attacks during the diffusion step.

We propose to let every node compute the next corrected state set $\mathcal{Z}_{k,\psi_i}$. Then, the combination in the diffusion step [157] is executed over shares $\psi_{j,i}$ inside the corrected reachable set $\mathcal{Z}_{k,\psi_i}$ of the node. If the share $\psi_{j,i}$ is outside its corrected set, it would be marked as "attacked share" and thus excluded. Thus, we can limit the effect of the attack on the diffusion shares $\psi_{j,i}$. More specifically, we assign the weight to zero if the share is outside the reachable set $\mathcal{Z}_{k,\psi_i}$. Shares inside $\mathcal{Z}_{k,\psi_i}$ take new weights $\hat{w}_{kj,i}$ where $\sum_{j \in \mathcal{N}_k} \hat{w}_{kj,i} = 1$.

$$w_{kj,i} = \begin{cases} 0 & \text{if } \psi_{j,i} \notin \mathcal{Z}_{k,\psi_i} \\ \hat{w}_{kj,i} & \text{else} \end{cases} \tag{5.17}$$

This illustrates Step 2 of Algorithm 5.

## 5.5 Evaluation

Our proposed algorithm is implemented in Matlab 2017 on a similar example to the one presented in [157], where a network of eight nodes attempts to track the position of a ro-

**Algorithm 5:** Secure Diffusion Kalman Filter

Start with $\hat{x}^k_{0|-1} = x_0$, $P_{k,0|-1} = \Pi_0$ and zonotope $\mathcal{Z}_{k,0|-1} = \langle \hat{x}_{k,0|-1}, G_{k,0|-1} \rangle$.
For all $k$, and at every time instant $i$, compute at every node $k$:
**Step 1**: Measurement update (Section 5.3):

$$\begin{aligned} P^{-1}_{k,i|i} &= P^{-1}_{k,i|i-1} + \sum_{j \in \mathcal{N}_k} H^T_{kj,i} R^{-1}_{j,i} H_{kj,i} \\ \lambda_{kj,i} &= P_{k,i|i} H^T_{kj,i} R^{-1}_{j,i} \\ \psi_{k,i} &= \hat{x}_{k,i|i-1} + \sum_{j \in \mathcal{N}_k} \lambda_{kj,i} (y_{kj,i} - H_{kj,i} \hat{x}_{k,i|i-1}) \\ G_{\psi_{k,i|i}} &= [(I - \sum_{j \in \mathcal{N}_k} \lambda_{kj,i} H_{kj,i}) G_{k,i|i-1}, \;\; \lambda_{k1,i} R_{1,i}, \\ &\quad \ldots, \lambda_{km_k,i} R_{m_k,i}] \end{aligned}$$

Reduce the order of the corrected zonotope $\mathcal{Z}_{k,\psi_i} = \langle \psi_{k,i}, G_{k,\psi_i} \rangle$ order by Girard method [160, p.7].
**Step 2**: Diffusion update (Section 5.4): Filter $\psi_{j,i}$ based on the reachability analysis, i.e. average $\psi_{j,i}$ from neighbors if they are within the expected reachable set and assign the weights $w_{kj,i}$ accordingly. ($w_{kj,i} = 0$ if $\psi_{j,i} \notin \mathcal{Z}_{k,\psi_i}$).

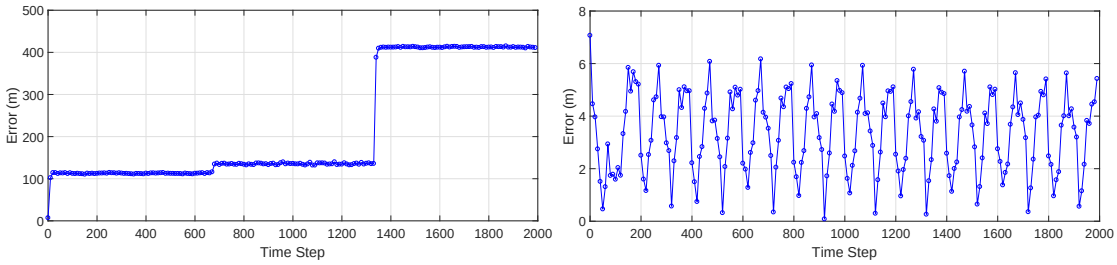
$$\begin{aligned} \hat{x}_{k,i|i} &= \sum_{j \in \mathcal{N}_k} w_{kj,i} \psi_{j,i} \\ G_{k,i|i} &= G_{k,\psi_i} \end{aligned}$$

**Step 3**: Time update:

$$\begin{aligned} \hat{x}_{k,i+1|i} &= F_i \hat{x}_{k,i|i} \\ P_{k,i+1|i} &= F_i P_{k,i|i} F_i^T + Q_i \\ G_{k,i+1|i} &= [F_i G_{k,i|i}, Q_i] \end{aligned}$$

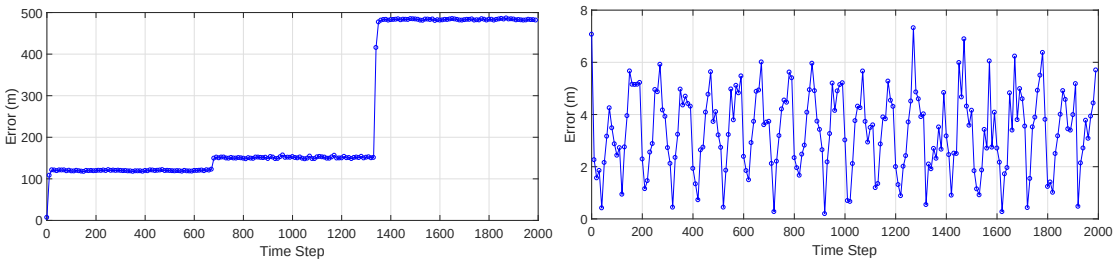
tating object. All computations run on a single thread of an Intel(R) Core(TM) i7-8750 with 16 GB RAM. We made use of Cora [161–163] for zonotope operations. Our example is quite representative for secure state estimation, since it includes modeling noise and measurements noise. The state of each node consists of the unknown 2-dimensional position of the object combined with the attack on the measurements. The state matrix in (5.5) is

$$F = \begin{bmatrix} 0.992 & -0.1247 & 0 \\ 0.1247 & 0.992 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{5.12}$$




**(a)** Insecure **(b)** Secure

**Figure 5.3:** Localization error at one node of the rotating target where all the diffusion shares are only under attack. Attacks are generated from uniform, normal and Pareto pseudo-random distributions. Y-scales are different in Figures (a) and (b).




**(a)** Insecure **(b)** Secure

**Figure 5.4:** Localization error at one node of the rotating target where all the measurements and the diffusion shares are under attack with time varying values. Attacks are generated from uniform, normal and Pareto pseudo-random distributions as shown in (5.13). Y-scales are different in Figures (a) and (b).

and the measurement matrix $H_{kj,i}$ is [0 1 1] or [1 0 1] in the sequence of the taken measurements. This means that the nodes take measurements of the unknown position of the object either in the x or y direction and the measurements are under attack $a_{k,i}$. We generate the attacks $a_{k,i}$ as following:

$$a_{k,i} = \begin{cases} 2\,\texttt{rand} + 4 & \text{if } t < 1/3T_{\text{sim}}, \\ \texttt{randp}(3,2) + 8 & \text{if } 1/3T_{\text{sim}} < t < 2/3T_{\text{sim}}, \\ 2\,\texttt{randn} + 50 & \text{if } t > 2/3T_{\text{sim}}, \end{cases} \tag{5.13}$$

where `rand` and `randn` return pseudo-random values drawn from the standard uniform distribution on the open interval (0,1) and the standard normal distribution, respectively. On the other hand, $\texttt{randp}(3,2)$ generates values form the Pareto distribution, where the shape equals 3, and the scale equals 2.The total simulation time is denoted by $T_{\text{sim}}$.

The implemented random attacks have time-varying statistics, as shown in (5.13). The simulations without protection and with protection are shown in Figures 5.2a and 5.2b,

**Table 5.1:** The mean and standard deviation of the localization error (m) of the rotating target at one node with and without the proposed protection algorithm.

| | Insecure | | Secure | |
|---|---|---|---|---|
| **Steps under attack** | **mean** | **std** | **mean** | **std** |
| Measurement step only | 31.558 | 28.119 | 3.306 | 1.523 |
| Diffusion step only | 219.325 | 136.567 | 3.194 | 1.474 |
| Measurement and diffusion steps | 250.161 | 164.489 | 3.220 | 1.518 |

respectively, by attacking the measurement step only. Then, we attack the diffusion shares $\psi_{k,i}$ using the three presented random generators. The insecure and secure versions run on the same values of random numbers for a fair comparison and the diffusion step only is under attack. The outputs are shown in Figures 5.3a and 5.3b. Finally, we attack both the measurement and diffusion steps and report the output in Figures 5.4a and 5.4b. We obtain the reported means and standard deviations in Table 5.1 with and without the proposed protection on the same set of attacks. The mean is around $3$m for the secure version with a standard deviation around $1.5$ regardless of the attack type.

## 5.6 Conclusions

We combine reachability analysis with secure state estimation to obtain a secure and fully distributed estimator. Our approach works for discrete-time, linear systems affected by disturbances, and measurement noises. Our proposed solution is fully distributed and does not require a fusion center. Our algorithm is the first algorithm that combines reachability analysis with secure state estimation. We consider attacks on the sensor levels as well as the communication links. At each time step, estimation output is supervised by reachability analysis to provide secure estimation shares. Reachability analysis allows us to have secure diffusion in distributed secure state estimation. We demonstrate the applicability of our approach with a simulation of a rotating target where the measurements and the diffusion shares are under attack.

# 6 Localization for Enabling IoT Device Selection and Control

There is widespread interest in having smart devices in the last decade. These devices penetrate every aspect of our daily lives in many forms including mobile phones, smartwatches, thermostats, and door locks. Also, smart home controllers, e.g., Amazon Echo[1] and Google Home[2], represent another recent wave of smart devices that are catching a lot of attention and gaining remarkable popularity. The development of smart devices has been fueled by research progress in several directions, such as improved network connectivity, reliability, and availability. The advancement in machine learning and data analysis also allows us to make semantics inferences from the sensory data streamed from these devices. However, making an easy and natural control interface for many surrounding devices at home remains an unsolved problem. Human interaction with machines in daily use should be intuitive and simple. Thus, much effort has been invested in the Human-Computer Interface (HCI) domain to enable more natural forms of interactions between humans and devices.

Different forms of interaction have been proposed, such as speech recognition [164], face recognition [165], gaze/eye-tracking [166], and hand gesture tracking [167]. However, despite this tremendous effort, we are still far from having a natural way to control and interact with devices. Vision-based methods (e.g., [168]) present a serious invasion of the user's privacy and they work only when sufficient lighting is provided in the room

[1]http://amazon.com/echo
[2]https://madeby.google.com/home/

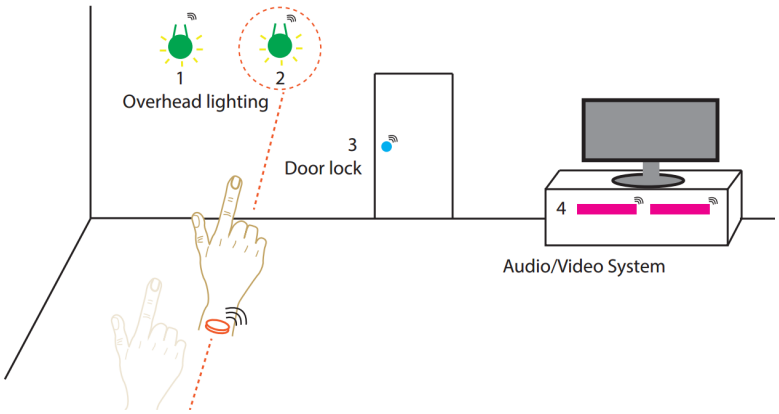



**Figure 6.1:** Gesture based IoT device selection using wearable devices.

assuming all objects are in the view without obstruction. Similarly, approaches based on speech recognition also invade privacy because they contentiously record audio and release it to remote cloud servers to interpret users' commands.

The hand gesture is a natural and effective communication method. Hand gesture recognition has received much attention, especially in the HCI domain. Different sensing modalities have been proposed to recognize hand gestures, including cameras [168], depth sensors, Wi-Fi signals [169, 170], and body-worn inertial sensors [171–173]. The last approach, in particular, fits well into the smart home scenario because of the wide adoption of smartwatches and other wearable devices that are equipped with inertial sensors. However, only a few existing gesture-based control systems have reached end users because there is no scalable and practical solution that fits into everyday life, yet. For example, a typical smart home may have tens of devices connected to each other, including lights, thermostats, locks, and other appliances. Current smart devices typically require every single family member to install applications for controlling these devices. It might additionally burden the users to assign semantic labels for each device such as "living room light 2" or "northeast door." With the increasing number of devices in users' surroundings, this process becomes cumbersome.

Existing hand gesture recognition methods do not well address the device selection problem. Although a body of literature has proposed different gesture recognition solutions [167, 170, 172–182], none of these techniques can *select* a device and *control* it without increasing the appliance installation overhead. For example, if a user wants to turn on the light in the living room, how does a smart home system know which device is intended? In fact, without augmenting the previous gesture recognition techniques with a position estimation technique coupled with predetermined locations of all smart devices, none of the existing techniques can be used to directly control a specific device based on human gestures unless a special gesture is assigned to each individual device. This motivates us to develop a new technology to enable accurate and scalable IoT device selection and control.

Towards achieving a scalable and practical architecture for selecting and giving commands to smart home devices, we design SeleCon, a gesture-based system that aims to provide a natural device selection and control method for users to interact with smart IoT devices. A user can simply point his arm towards the target device to select it, as shown in Figure 6.1. SeleCon stands for devise selection and control. SeleCon is able to identify which IoT device is selected by monitoring the direction of the wrist movement. The user then draws a gesture in the air to give a command to the selected device.

As smart devices can be placed in arbitrary locations and users can move over time, inertial sensors alone are not sufficient to identify the intended target. Therefore, we designed and implemented a smartwatch prototype equipped with an ultra-wideband (UWB) transceiver, and we use pair-wise ranging measurements between the smartwatch and the IoT devices to identify the target. The intuition behind our device selection process is that when a user points towards a given device, the smartwatch attached to the wrist of the user will get closer to the chosen device after the transition of the pointing event. We use different machine learning algorithms to verify our hypothesis. We also develop machine learning models for recognizing hand gestures for giving commands to

**Table 6.1:** Summary of related work.

| Research | Pointing Detection | Gesture Recognition | Requirements |
|---|---|---|---|
| SeleCon | ✓ | ✓ | Smart watch. |
| WristQue [182] | ✓ | ✓ | Pre-calibration of magnetic fields. and full localization using UWB |
| WiTrack [169] | ✓ | ✗ | Does not require the user to carry any device. |
| Inertial Gestures [183] | ✗ | ✓ | Smart watch. |
| Kinect Pointing [184] | ✓ | ✗ | Kinect within 3.5m. |

target devices. One major challenge is that UWB is known to be power-hungry compared to inertial sensors. To address this problem, we use the low power inertial sensors to implement a motion-based triggering module so that UWB ranging is turned on only after potential pointing actions are detected. Therefore, SeleCon can effectively reduce the operating time of the UWB transceiver to save energy.

This chapter is based on our publication in [23]. The rest of the chapter is organized as follows: Section 6.2 provides an overview of the proposed SeleCon system architecture. Section 6.1 summarizes the related work. We then go through SeleCon module by module. Pointing event detector is illustrated in Section 6.3. Section 6.4 introduces a simplified formulation for pointing gesture recognition problem, highlights the challenges in pointing gesture recognition, and introduces device selection algorithm using pattern matching. The language of gestures is introduced in Section 6.5. Section 6.6 evaluates SeleCon. Current limitations and future work are shown in Section 6.7. Finally, Section 6.8 concludes this chapter.

## 6.1 Related Work

To examine various interaction modalities that are used to communicate with IoT devices, including device selection and device control, we broadly partition prior work into three groups and compare the most related work in Table 6.1.

### 6.1.1 Inertial-based interaction

Previous research showed that inertial sensors are good at capturing local movements. For instance, it has been demonstrated that inertial sensors can be used for full body posture [172]. As wristbands become more and more popular, a body of literature explores the sensing boundary in this form factor. Xu et al. [185] point out that it is possible to track arm-level [173], hand-level [167, 179], and finger-level gestures [171, 186]. Shen et al. also show that since wrist position is determined by both the shoulder and the elbow motions [187], the full arm posture can be sensed even by a wristband. Based on

this work, several interesting sensing applications such as driving [180, 188], whiteboard writing [189], gaming control[3] [190], and writing or drawing in the air [181, 183, 185] have been developed. Researchers found inertial sensors are capable of capturing subtle hand movements, and sensitive information can be leaked when a user types [191–193]. Interestingly, WristQue [182] combines environmental and inertial sensing with precise indoor localization, using UWB for pointing and gestures recognition. However, WristQue needs magnetic field pre-calibration and full localization information, which are critical limitations in that work. SeleCon aligns with all these works and employs inertial sensors for two use cases: Accurate pointing action detection to save energy and hand gesture recognition for controlling devices.

### 6.1.2 Wireless-based interaction

Prior work attempts to use RFID [175, 176] and audio [177] for gesture recognition. However, they are not feasible because an RFID reader typically has a coverage limit of $10m^2$. Also, ambient sound may add noises and decrease the accuracy. Perhaps WiFi is a better medium to "observe" interactions between humans and devices as several works have demonstrated WiFi can mimic human sensations as in WiSee [170], WiHear [194], and WiFinger [195]. One popular technique is leveraging the Doppler effect to detect moving objects [174] which allows the system to "see through the wall" [169]; Kim et al. also exploit this phenomenon to monitor human activities [196]. Extracting information Channel State Information (CSI) from the Network Interface Card (NIC) is another technique to sense the environment, such as occupancy detection [197], typing [178], falling detection [198], and activities involving body displacements in general [199, 200]. Though gesture recognition via WiFi has been proved possible, none of this previous work is capable of identifying which device is pointed at by a user. This is because WiFi cannot give good range resolution. In contrast, UWB can provide high resolution of ranging measurement, which makes it a promising technology for indoor localization [201]. In SeleCon, we choose UWB for device selection because pointing is instant and the distance between wrist start and stop positions is short.

### 6.1.3 Vision-based interaction

Microsoft Kinect[4] and other commercial products employ both an RGB and a depth camera to track the human skeleton. Sharing the same idea, Digiteyes [202] reports 27 degrees of freedom (DOF) hand model at a speed of 10 Hz. This technique has also been applied on detecting human gestures [168] and sign language [203]. Jing et al. [184] recognize pointing events using Kinect. Perhaps intuitive to humans, vision-based approaches, however, suffer from requiring a line of sight and good lighting conditions, and also impose severe privacy invasion. HeatWave [204] and HeatProbe [205] use thermal cameras to detect and track how users interact with appliances.

[3] Nintendo wii. http://www.nintendo.com/wii

[4] https://developer.microsoft.com/en-us/windows/kinect

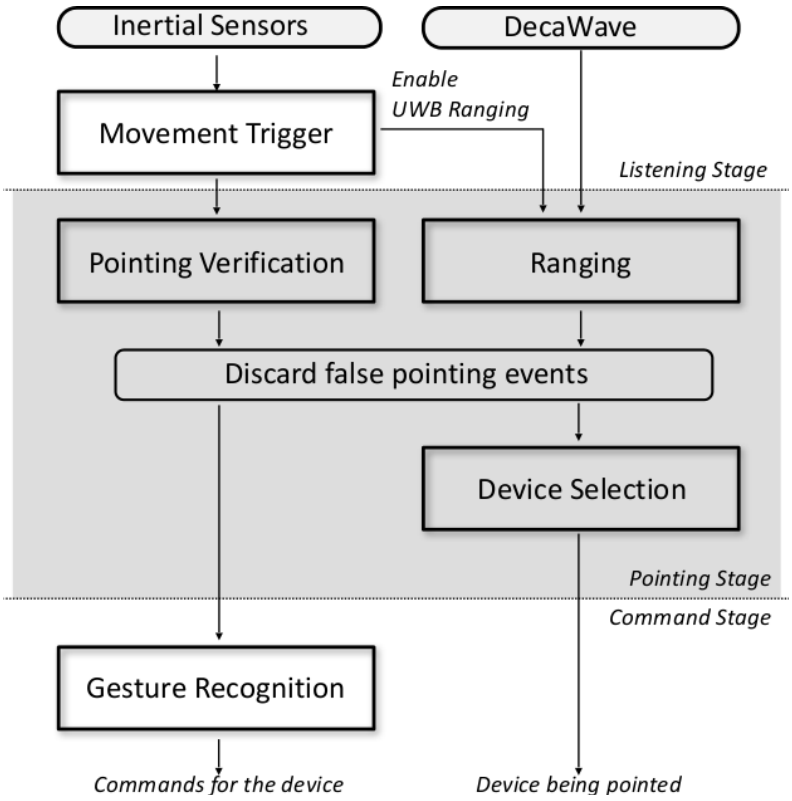



**Figure 6.2:** SeleCon system overview.

## 6.2 System Overview

We present SeleCon for device selection and control. The different stages in the SeleCon processing pipeline is presented in Figure 6.2. Broadly speaking, our system can be divided into three different stages in the following chronological order: the *listening stage*, the *pointing stage*, and the *command stage*. In *listening stage*, SeleCon detects all potential wrist motion events using the low-power inertial sensors. After a hand motion event is detected, SeleCon moves into the *pointing stage*. Our system uses the inertial sensor to verify whether the user has started pointing to a target device. Concurrently, SeleCon turns on the UWB transceiver to perform distance ranging between the smartwatch and the surrounding smart devices for a predefined time window. If the pointing action is verified by our system, SeleCon performs machine learning algorithms over UWB time series to determine which device is selected. The last step is the *command stage*. SeleCon again leverages the inertial sensors, exploits machine learning algorithms to classify the hand gestures. Figure 6.8 shows the supported gestures in our system. Figure 6.3 shows an example of inertial sensor data over a valid device interaction session. A session includes two parts, a pointing event followed by a gesture to give the command to a device. Note that the entire process is finished within 2 seconds.

## 6.3 Pointing Event Detection

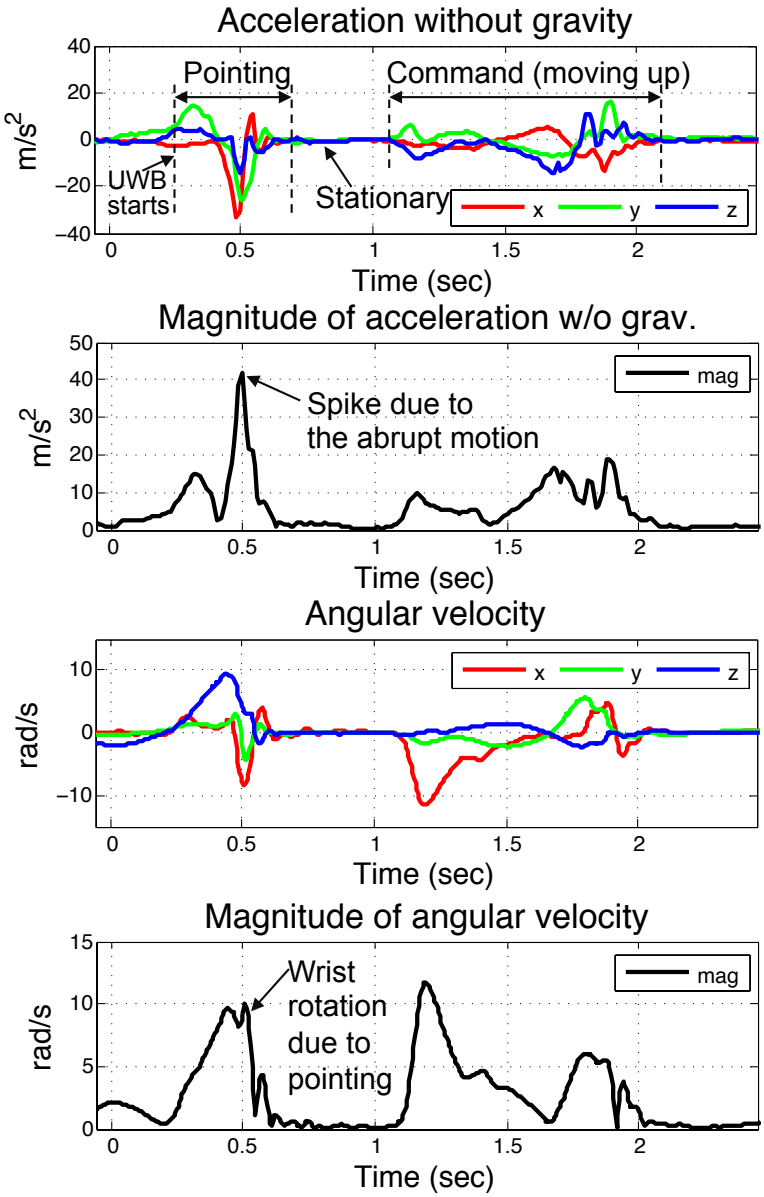



**Figure 6.3:** An example of inertial data: A user points to a device (TV) and conducts a moving up gesture (raise volume).

We aim to save energy by using a pointing gesture as the preamble of one device interaction session. Since UWB is $10x$ more power consuming than inertial sensors, it is infeasible to turn on the UWB for device selection for a long time. To cope with this issue, we use wrist motion as a trigger and enables the UWB only when a user tries to select a device. The *movement trigger* module in the *listening stage* aims at using low power inertial sensors to detect local wrist movements. As demonstrated in Figure 6.3,

the motion of pointing is fast and the duration is no longer than $0.5$ seconds. Thus, as long as a user *starts* moving her wrist, SeleCon enters the *pointing stage* and turns on the UWB transceiver. All the wrist movements are then passed to the *pointing verification* module, which checks whether the motion is a pointing gesture based on the inertial time series data. Below we provide the details of the *movement trigger* module and *pointing verification* module.

**Movement Trigger Module**: It aims to identify all the possible pointing events based on inertial sensors. However, if we make the module over sensitive, the UWB transceiver will wake up more frequently, causing the above-mentioned energy concern. In our design, we make the *movement trigger* module responsive enough so that even slow pointing gestures can be captured. Whenever a possible pointing event is captured, the *movement trigger* mobile turns on the UWB transceiver, records both inertial data and UWB ranging data for a couple seconds, and then passes the collected data to the *pointing verification* module which verifies whether the detected motion is a pointing gesture.

**Pointing Verification Module**: Many false pointing events are likely to be included since the *movement trigger* module is set to capture small wrist movements. The goal of the *pointing verification* module is to keep those events that are intended to point to objects. The *pointing verification* module combines a number of heuristics to verify the pointing gestures from inertial measurements. Figure 6.3 shows both accelerometer and gyroscope data collected within a valid pointing event. Note that the gravity has been removed from the accelerometer data, processed in the hardware level. Since a pointing gesture is a fast action, it will cause high acceleration showing as a spike in accelerometer data (e.g., at $t = 0.5$). On the other hand, though people have different habits to point towards objects, people naturally rotate their elbow joints and fully stretch their arms, making the smartwatch mounted on the wirst facing a different orientation. The orientation difference is reflected on gyroscope data, which measures the angular velocity, demonstrated in Figure 6.3. We pick an acceleration and an angular velocity magnitude threshold such that true pointing events can be distinguished from incorrect ones. Though the magnitude of inertial data is a good indicator of pointing gestures, it is still not robust enough to remove all false events. Users usually hold their arms for a short period (i.e., a couple hundred milliseconds) after pointing to an object. Therefore, if inertial sensors are stable for a predefined length, we consider this as a pointing event.

## 6.4 IoT Device Selection

After the system has detected a pointing event, it is time to determine which device is selected by processing the UWB ranging measurements in the *device selection* module. Assume that we have a network of $N$ IoT devices and a single user in a room. We denote the position of each device $\bar{n}_i$ for $i \in 1, ..., N$ by $p_i \in \mathbb{R}^3$. Similarly, the position of the smartwatch $\bar{n}_u$ (i.e., user's wrist) is denoted by $p_u(t)$ at time $t$. SeleCon gets the ranging measurements between the smartwatch $\bar{n}_u$ and any smart device $\bar{n}_i$ from the UWB transceiver, and we denote the measurement by $r_i(t)$ at time $t$. Consider the user is pointing to a target IoT device whose index is denoted by $i^*$. Since pointing

is a process to move the wrist closer to the target device, mathematically speaking, the distance between $\bar{n}_{i^*}$ and $\bar{n}_u$ should decrease the most comparing with any other $\bar{n}_i$ where $i \neq i^*$. By collecting the pairwise ranging measurements between the smartwatch and every smart device, we can find out $i^*$ by solving the following minimization problem:

$$\begin{aligned} i^* &= \arg\min_i \quad \Big( \|p_i - p_u(t_f)\|_2 - \|p_i - p_u(t_s)\|_2 \Big) \\ &= \arg\min_i \quad \Big( r_i(t_f) - r_i(t_s) \Big) \end{aligned} \tag{6.1}$$

where $t_s$ and $t_f$ are the start and finish time of the pointing gesture, respectively. In order to measure the distance between the user and any IoT device, we consider both *single-sided* and *double-sided* two-way ranging techniques, which are described in Section 3.3.1. In the *single-sided* two-way ranging, two devices $i$ and $j$ take turns to send message to the other ones. The distance is derived by the round trip time of the message transfer. *Double-sided* two-way ranging can be seen as an extension of the *single-sided* technique, and the difference is that the first device $i$ sends the third message to $j$ so that the round trip time can be also acquired from the second device. This approach usually gets a better ranging accuracy because the third message compensates for the clock drift. However, both aforementioned ranging techniques have a range estimation error of around $10cm$ to $30cm$, which can impact the device selection accuracy. The possible reasons of the large ranging errors are the range noise, the pointing length, and the spatial diversity. In the following subsections, we will present the challenges to distinguish the target device from the incorrect ones, and then provide the features that have potential to identify the correct device.

### 6.4.1 Spatial Resolution and Gesture Length

When a user points to an IoT device, the length of that gesture inherently defines a signal-to-noise ratio (SNR). We denote this length $\ell$, and we assume that the range error follows a non-zero mean Gaussian distribution $e \sim N(\mu_r, \sigma_r^2)$. This error is not necessarily i.i.d, but for the sake of simplicity, we assume that the range error is independent across time. Ideally, the start wrist position $p_u(t_s)$, the end wrist position $p_u(t_f)$, and the selected device $p_{i^*}$ should fall on a straight line. However, when a user points to a device, the eyes, the wrist, and the device are not co-linear, causing the pointing angular error $\tilde{\theta}$. Additionally, if there is a device close to the true device, our system may be mistaken and select the wrong one.

We define the angle formed by the true device $\bar{n}_{i^*}$, the user $\bar{n}_u$, and the cloest device $\bar{n}_j$ in terms of angle as $\theta_{min}$, illustrated in Figure 6.4. In the case of high spatial diversity, i.e., when $\theta_{min}$ is quite large and when $\ell$ is large with respect to $\sigma_r^2$, it is relatively easy to distinguish the correct device $\bar{n}_{i^*}$ that a user is pointing to from any other devices $\bar{n}_j$ for $j \neq i^*$. Figure 6.5 shows the ranging difference measured from both target device $\bar{n}_{i^*}$ and other devices $\bar{n}_j$ over time in a high spatial diversity area. The range difference at the $k^{th}$ sample is defined as the delta of current range value and the start range value, or $\triangle R_i(t) = r_i(t_f) - r_i(t_s)$. Each trace in Figure 6.5 represents a pointing gesure measured

by a device. The true traces which are plotted in blue correspond to the distance estimates between the user and $\bar{n}_{i^*}$, and the false traces plotted in red are those corresponding to any other device. As we can see in the figure, the true traces show the range differences are negative, as the user moves her wrist closer to the target device. In contrast, most red traces measured from other devices are positive. These two kinds of traces can be easily separated.

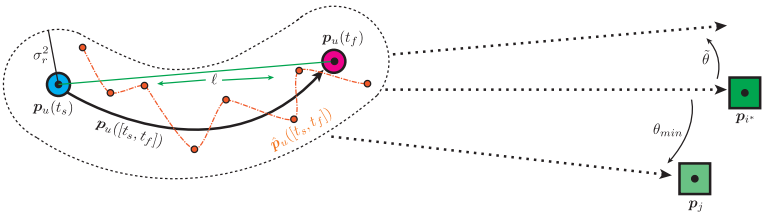



**Figure 6.4:** Ranging errors and angular (spatial) resolution in gesture-based IoT device selection.

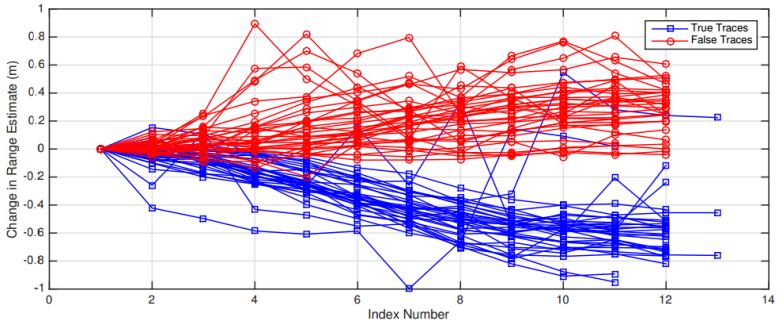



**Figure 6.5:** Example ranging traces during pointing. These examples are in an environment with high spatial diversity where $\triangle R$ is sufficient to identify the selected device

In the case of low spatial diversity, the simple metric $\triangle R$ no longer suffices as a reliable metric for discerning the desired IoT device from other IoT devices. For instance, Figure 6.6 shows estimated range differences for a low spatial diversity environment, which we deployed the IoT devices as illustrated in Figure 3.5. The result shows that true range measurement traces have similar a pattern in high spatial diversity case (i.e., blue traces in Figure 6.5), but the false traces from some other devices overlap with the true traces. Figure 6.7 plots the distribution of ranging differences $\triangle R$. As we can see, there is an overlap between true and false range differences within $-0.5m$ to $-0.8m$. Thus, in order to provide higher device selection accuracy, we need to explore other metrics. In the following subsection, we explore a more principled approach to the device selection problem, keeping device power and computational overheads in mind.

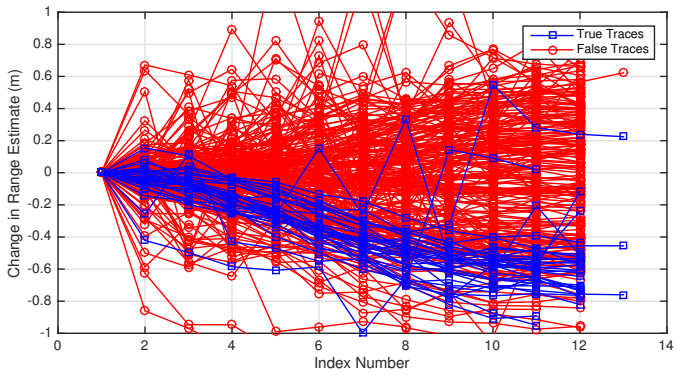



**Figure 6.6:** Example ranging traces during pointing. These examples are in an environment with low spatial diversity where $\triangle R$ is not sufficient to identify the selected device

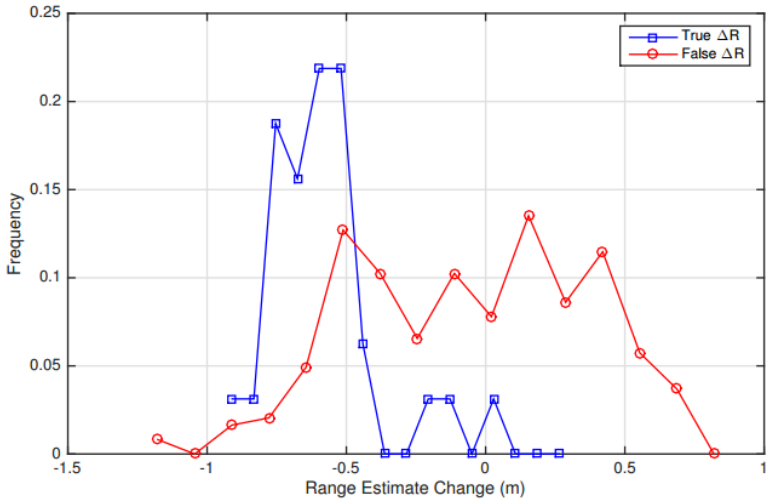



**Figure 6.7:** Range difference, $\triangle R$, for true and false ($\bar{n}_{i^*}$ and $\bar{n}_{i \neq i^*}$) as a probability density function.

## 6.4.2 Device Selection by Pattern Matching

We have demonstrated that the estimated distance change, $\triangle R$, is insufficient to reliably identify the target device to which the user is pointing. Additional features have to be explored to find correct devices. We first present features considered in our system, and explain how we determine the selected devices by the proposed classification algorithm.

### 6.4.2.1 Relevant Ranging Features

SeleCon considers the following features:

**Linear Fit:** Although the wrist path of a point gesture is curved as illustrated in Figure 6.4, in reality the path is close to a straight line. Any divergence from this fit could indicate that the range estimated trace could belong to an undesired device. More specifically, if we estimate a linear fit describing the range estimate between $\bar{n}_u$ and $\bar{n}_i$ for a given device $i$ and noise $v$ at sample index $k$, we describe the fitting error as the mean squared residual $MSR = \frac{1}{K}\sum_k v_k^2$, where $K$ is the number of ranging samples when pointing. Intuitively, a low $MSR$ indicates a good linear fit and consequently more likely being the true trace. We use this feature to enhance the pointing recognition process.

**First Path Loss:** Path loss is defined as power density reduction from a transmitter to a receiver. The accurate timing provided by UWB radios is enabled by measuring the energy in the radios accumulator corresponding to the communication along the first path. One consequence of this accurate timing is that the same energy can be used to estimate the power of the communication along the first path of communication typically the line-of-sight path. Since people rarely point to an object which is out of their sight, the chosen device is likely to report low first path loss, denoted by $fploss$.

**Range Data:** If there are two devices which align along the direction that a user points towards, naturally we consider that the user is interacting with the closer one. Thus, the distance between a device and a user also plays an important role. To estimate this distance, we use the raw range result as a feature.

**Angular Divergence:** As a user points from $p_u(t_s)$ to $p_u(t_f)$, she is attempting to point as closely towards a device as possible. In an ideal case, the line between start and finish is perfectly co-linear with the coordinates of the IoT device itself. In practice, however, there is some angular divergence $\tilde{\theta}$ as shown in Figure 6.4. In order to calculate this angle, we must know the device position $p_i$ as well as the start and stop position of the user's hand. This requires a collaborative position estimate among multiple IoT devices, which assumes a certain device density and additionally consumes more power due to added communication costs. When possible, however, $\tilde{\theta}$ can be used to increase the accuracy of gesture-based IoT device selection.

We combine the previous features into an aggregate feature vector, defined as

$$f = [\triangle R, MSR, fploss, \hat{r}(t_s), \tilde{\theta}]^T \tag{6.2}$$

For each pointing event, there are $N$ different data points corresponding to the feature vectors computed from the ranging measurements between the user's smartwatch and each of the $N$ surrounding devices. Only one among these data points corresponds to the true target device $\bar{n}_{i^*}$ while the remaining $N-1$ are not selected. Thus, we label a feature $f_i$ as 1 for the true device $\bar{n}_{i^*}$ and 0 otherwise. Although we have included the expensive feature $\tilde{\theta}$ in our feature vector $f$ definition, we report the system accuracy based on different experiments that uses a feature vector with and without this feature

being included. We believe that these experiments mimic diverse settings and provide more realistic results for sparse deployments and restricted energy reserves.

#### 6.4.2.2 Classification Methods

From each feature vector we can estimate whether a particular IoT device is selected or not, but this approach ignores any potential communication or collaboration between connected IoT devices. Another option in the classical classification methodology is to have a single centralized server classifies based on the feature vectors from IoT devices. However, this collaborative classification is not scalable and has a high communication cost. On the other hand, in non-collaborative classification, each IoT device operates independently, attempting to classify itself as selected or unselected. Other than the communication with $\bar{n}_u$ required for range estimates, no additional communication is performed. This saves power, but it comes at the cost of very high selection errors. We do not want multiple devices to be selected at the same time. Therefore, we choose to do collaborative classification; devices are allowed to communicate. Rather than sending entire feature vectors, however, each device will send only a summary of their classification results and indicate the certainty with which the classification was made. This allows the network to arrive at a consensus of maximum certainty of which device was selected, enabling collaboration without prohibitively high communication overheads.

## 6.5 Hand Gesture Recognition

The second half of a pointing session is a hand gesture to control the target device. Individual devices can be configured to execute different actions in response to the different gestures. Hand gesture is arguably one of the intuitive ways for describing actions and does not require moving closer to the target device. Currently, SeleCon supports 11 different gestures which can be assigned to up to 11 different operations for each individual device. Our system can be easily extended to support additional gestures, but we believe this number is satisfactory for the needs of most devices. The list of supported gestures is shown in Figure 6.8.

Our gesture recognition module relies on the measurements from the inertial sensors in our smartwatch. Namely, we use three axes of both accelerometer and gyroscope measurements. According to our language definition, a user should point to the device first then draw a gesture command in the air. Therefore, after the end of the pointing action, we collect 3 seconds of inertial measurements. From these samples, we compute a feature vector consisting of the following features along every axis of both accelerometer and gyroscope:

- The three quartiles (25%, 50%, 75%).
- Standard deviation $\sigma$

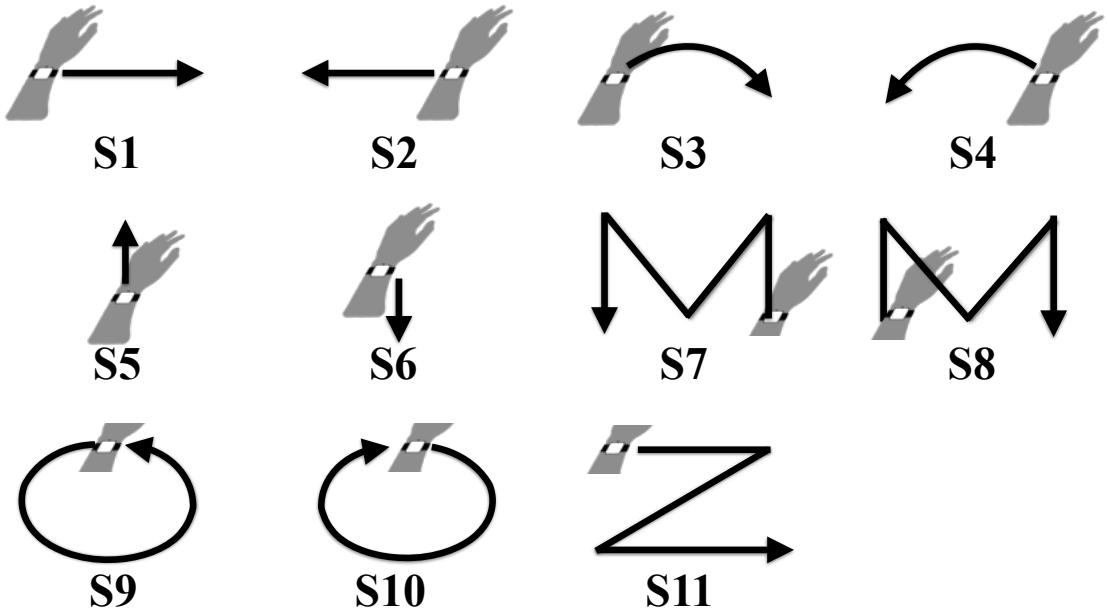



**Figure 6.8:** List gestures supported by SeleCon.

**Table 6.2:** Classification results for gesture-based IoT device selection, using collaborative technique. Angle is not part of the feature vector.

| **Collaborative Classifier** | **Single-sided ranging accuracy** (%) | **Double-sided ranging accuracy** (%) |
|---|---|---|
| Voting on SVM (linear) | 41.46 | 43.29 |
| Voting on SVM (quadratic) | 48.78 | 50.00 |
| Voting on SVM (rbf) | 50.60 | 55.48 |
| Voting on RF | 78.04 | 80.48 |

- Skewness:

$$skewness = E\left[\left(\frac{X-\mu}{\sigma}\right)^3\right]$$

- Kurtosis:

$$Kurtosis = \frac{E\left[(X-\mu)^4\right]}{\left(E\left[(X-\mu)^2\right]\right)^2}$$

As a result, in total we have 36 features. We use these features to train a machine learning classifier to recognize different gestures.

**Table 6.3:** Classification results for gesture-based IoT device selection, using collaborative techniques. Angle is part of the feature vector.

| Collaborative Classifier | Single-sided ranging accuracy (%) | Double-sided ranging accuracy (%) |
|---|---|---|
| Voting on SVM (linear) | 45.12 | 46.34 |
| Voting on SVM (quadratic) | 50.00 | 56.70 |
| Voting on SVM (RBF) | 64.63 | 66.46 |
| Voting on RF | 81.09 | 84.14 |


**Figure 6.9:** Hardware prototype of UWB-equipped smartwatch

## 6.6 Evaluation

We deployed a custom ultra-wideband RF testbed based on the DecaWave DW1000 IR-UWB radio[5] for evaluation of SeleCon. The whole experimental setup overview is shown in Figure 3.5. Our testbed consists of smart devices anchor nodes, an UWB-equipped smartwatch (Figure 6.9) and a motion capture system as described in Section 3.5

### 6.6.1 Pointing Event Detection

To evaluate the accuracy of pointing event detection, we conduct the following two experiments. The first experiment evaluates how well our system can capture pointing events. We ask three participants to wear our smartwatch and perform at least 50 pointing events. The participants are instructed to naturally point to any device (e.g., an anchor node) without restrictions. During the data collection sessions, participants could move freely within the testbed area. We used the OptiTrack motion capture system[6] to collect smartwatch location traces and post-processed the pointing events. The detection rate is 91.9%.

With such a promising detection rate, one might naturally ask for the false alarm rate. This has to be broken down to two further questions: (1) How frequent does the UWB radio have to be active? (2) What is the false alarm rate in reporting pointing events? To answer these questions, we conduct a second experiment to collect non-pointing inertial

[5]http://www.decawave.com/products/dw1000
[6]https://www.optitrack.com

data from five students. Participants are allowed to do any activity they want as long as no pointing gestures are involved. We collect 11.6 hours of data in total. Our results show that SeleCon only has to enable the UWB ranging for only 8.0% of the time. SeleCon reports 73 false events in total, with an average of 6.29 false events per hour.

To further reduce the false event rate, SeleCon considers the result from the gesture recognition module. Since a valid command should include a gesture, if the gesture recognition module returns none of these 11 predefined gestures, we discard the pointing event. This further step decreases the false alarm rate to 2.5 false events per hour.

### 6.6.2 Device Selection

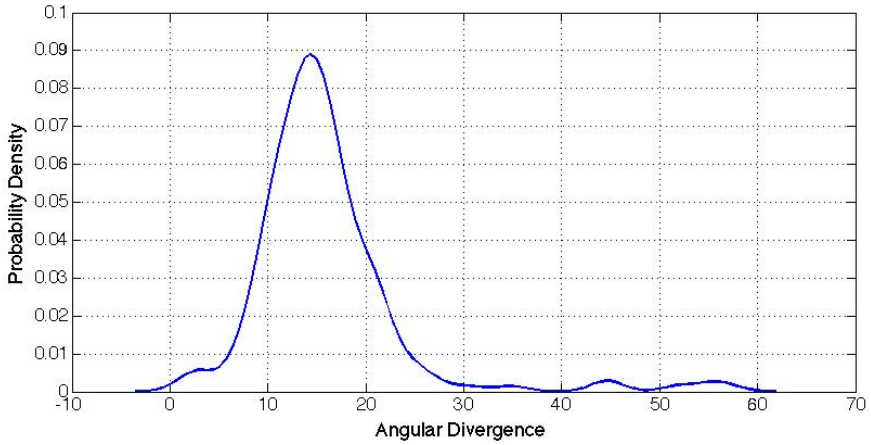



**Figure 6.10:** Probability density function of the angular divergence of the pointing events.

In order to evaluate the pointing-based IoT selection system, we perform a series of pointing events using our prototype of the UWB-enabled smartwatch. For groundtruth collection of the users' wrist motion, we attach a set of infrared reflectors to the smartwatch prototype. We use the OptiTrack motion capture system to track the realtime positions. We collected 200 pointing events to various devices in total performed by various users. We evaluated the accuracy of different classification techniques for device selection. The results of collaborative classification schemes using support vector machine (SVM) and random forest (RF) are shown in Table 6.2 where the angle $\tilde{\theta}$ is not part of the feature vector. Table 6.3 shows the result of different classification schemes under collaborative schemes in which the angle $\tilde{\theta}$ is part of the feature vector. The used angular divergence in pointing has the shown probability density function in Figure 6.10. The ground truth angular divergence is calculated by processing the motion capture logs while pointing.

Collaborative classification achieves a good accuracy. This is due to the voting scheme among all the $N$ IoT devices, which follows non-collaborative classification. We communicate classification certainties between all devices, arriving at a maximally certain

positive label. In particular, in the case of SVM we communicate the margin between a datapoint and its n-dimensional polytope. In the case of RF, we communicate the numerical average of the ensemble prediction. We report the results with and without the expensive feature $\tilde{\theta}$ in order to provide more realistic results for sparse deployments and restricted energy reserves. We will also show the results for both single-sided and double-sided ranging.

In doing so, we achieve for single-sided ranging without the feature $\tilde{\theta}$ an accuracy of $50.6\%$ for collaborative RBF SVM, and $78.04\%$ for collaborative RF. Collaborative linear and quadratic SVM show $41.46\%$ and $48.78\%$ accuracy, respectively. On the other hand, using double-sided ranging messages achieves $80.48\%$ accuracy for collaborative RF. Other classifiers results are shown in Table 6.2. Introducing the feature $\tilde{\theta}$ enhanced the accuracy. Collaborative RF achieves the best accuracy of $84.14\%$ and $81.09\%$ for double-sided ranging and single-sided ranging, respectively. Collaborative RBF stands in rank two position with accuracy $64.63\%$ and $66.46\%$ for single-sided ranging and double-sided ranging, respectively. We report only accuracy, because at any given pointing event we know that only a single IoT device out of N possible is selected. The accuracy is the percentage of times we choose the correct device. Recall that our testbed size is $9m$-by-$10m$, where on average every deployed device is $3.5m$ apart. We should mention that the reported accuracy is the average accuracy when standing in different positions in the room. This accuracy depends on the co-linearity between the devices with respect to the user's position. We will analyze the effect of co-linearity between devices and the distance between them regarding accuracy in the next subsections.

#### 6.6.2.1 Analysis of Distance Between two Devices

We study the effect of the distance between two devices on the accuracy of selecting one of them. We conducted an experiment where the user points to one of two devices. We change the distance between the two devices from $30cm$ to $2m$ and compute the accuracy of device selection at each distance. Figure 6.11 shows how the selection accuracy changes with the change of the distance between devices. We emphasis that the reported accuracy in Tables 6.2 and 6.3 is the average accuracy across different positions. On the other hand, the reported accuracy in Figure 6.11 is the result of standing in one position in the middle of the testing environment. Therefore, the reported accuracy in Figure 6.11 is better than the previous results.

#### 6.6.2.2 Co-linearity Analysis

We also study the effect of co-linearity between different devices and the user's position on the reported accuracy. The co-linearity is defined in terms of the angle $\alpha$ in the x-z plane as shown in Figure 6.12. Two devices are co-linear if $\alpha = 180$. In order to analyze the co-linearity effect on the reported accuracy, we conducted two sets of experiments. Two devices are places at a height of $75cm$, i.e, their position is $75cm$ in y direction. In the first set of experiments a user is asked to keep pointing to the two devices while standing, i.e, from height $140cm$ in the y direction. On the other hand, the second set of

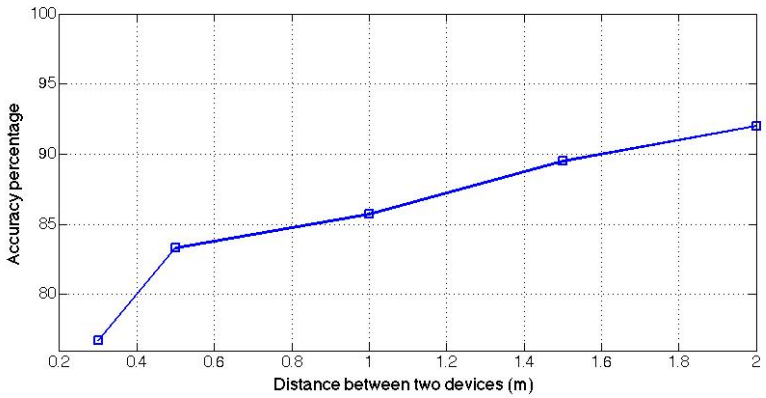



**Figure 6.11:** The effect of distance between two devices on SeleCon accuracy

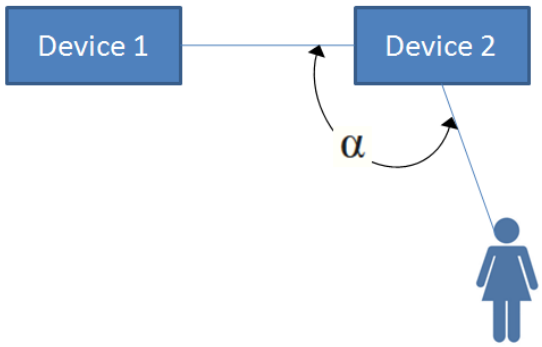



**Figure 6.12:** Co-linearity effect.

experiments is conducted while sitting, i.e, from height $75cm$ in the y direction. Figure 6.13 shows the co-linearity effect while pointing from height $75cm$ and $140cm$ on devices at $75cm$ in the y direction. Again, we emphasize that the conducted experiments were done while standing roughly in the middle of the $9x10m^2$ room. Therefore, the reported accuracy is much higher than the average accuracy in Tables 6.2 and 6.3.

#### 6.6.2.3 Power Analysis

Our device selection technique targets IoT devices that are potentially battery powered. Because of this, care has to be taken to ensure that power consumption is minimized. Here we briefly analyze the power consumption for just the mobile (e.g. smartwatch) device that the user wears. Note that in practice, some of the stationary (anchor) IoT devices may be battery powered as well, and in these cases a low power *sniffing* strategy should be employed for listening to messages from the mobile device. For the mobile case, we will ignore the energy cost of computation for classification, as this is dwarfed by the energy required to transmit and receive using the DW1000 UWB transceivers.

In the idle case (when the smart watch is not ranging and is idle, listening for a pointing gesture), $6\mu W$ of power are consumed. When transmitting or receiving UWB frames,

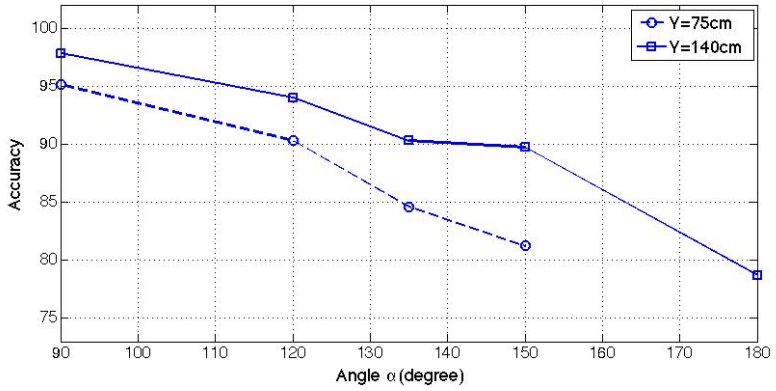



**Figure 6.13:** Co-linearity effect while pointing from height 75 cm and 140 cm at devices at 75cm in y direction.

there is a $5ms$ wakeup period during which $3mW$ of power are consumed, followed by a $260mW$ power demand for $200\mu s$ for TX and a $370mW$ power demand for RX. We will ignore the sleep power consumption (6 $\mu W$) for now, as it will be shadowed by the processing consumption and analog conversions for gesture detection on the smart watch. These power numbers are derived from radio datasheets[7]. For $N$ IoT devices, energy consumption during each pointing session is calculated as follows:

$$\begin{aligned} E &= K \cdot N^*(E_{wake} + 2 \cdot (E_{TX} + T_{RX}P_{RX})) \\ &= K \cdot N^*(15\mu J + 2 \cdot (52\mu J + 370\mu J)) \end{aligned} \tag{6.3}$$

with a listen period of $T_{RX} = 1ms$ following each transmission (and expected response). Here $N^*$ represents all nodes within communication range, rather than the full number of networked devices, and $K$ is the number of range measurements calculated per node—roughly 20, in the above experiments. For $N = 8$, as is the case in the results shown here, we have $E = 137.4mJ$ per point. For today's smart wearables, a battery a capacity with between 750 and 1400 mWh is common—this corresponds to a range between $2.7kJ$ and $5kJ$ in each battery, meaning that if, for example, 100 pointing gestures are made each day, there is a 0.27% to 0.5% overhead in battery life. This is a very reasonable price to pay for the added convenience of high fidelity gesture-based IoT device selection. With improvements in commodity UWB hardware, however, it is likely that this overhead will decrease further.

### 6.6.3 Gesture Recognition

To evaluate our hand gesture recognition module, we collected 1290 gestures samples from volunteers who were given the freedom to wear the watch on their left hand or right hand. On average, we collected 117 samples for each gesture from the supported

[7] http://www.decawave.com/products/dw1000

**Table 6.4:** The accuracy of different classifier for gesture recognition.

| Classifier | Accuracy |
|---|---|
| SVM (Linear) | 97.03% |
| SVM (quadratic) | 96.50% |
| SVM (RBF) | 93.12% |
| RF | 95.48% |

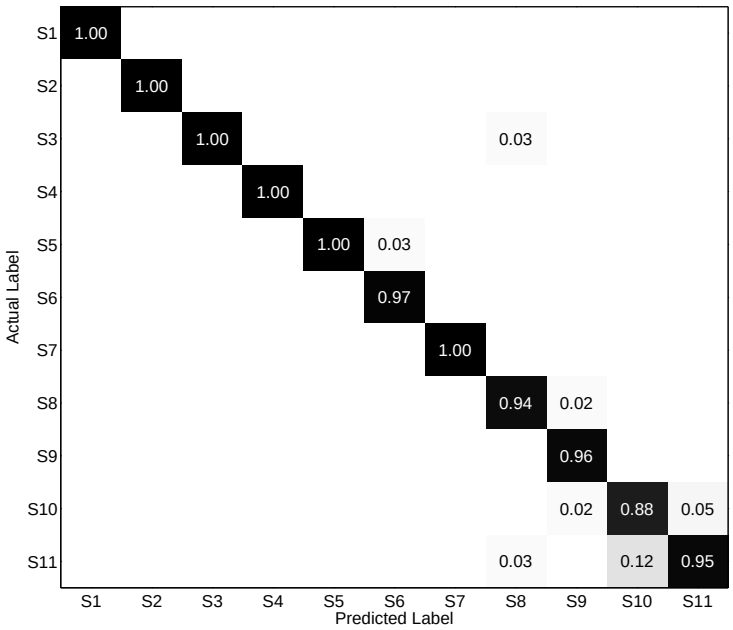



**Figure 6.14:** Confusion matrix of the gesture recognition classifier.

gestures shown in Table 6.8. We used the collected measurements to evaluate the performance of different classification algorithms at the *gesture recognition* module. Table 6.4 shows the accuracy of the Gesture Recognizer module. Linear SVM achieves about $97\%$. Quadratic SVM achieves an accuracy of $96.5\%$. Then, Random Forest comes the third with $95.48\%$ accuracy. Finally, SVM with RBF kernel achieves $93.12\%$. Figure 6.14 shows the confusion matrix of the gesture recognition. In summary, SeleCon has a robust gesture recognizer module with 11 gestures which should be sufficient to control any IoT device.

## 6.7 Limitations and Future Work

While we are very positive about SeleCon's current capabilities, we admit the following limitations in our architecture:

- SeleCon requires the user to wear a custom smartwatch equipped with both an inertial measurement unit (IMU) and an ultra-wideband radio.
- Currently, SeleCon cannot be used by more than one user simultaneously.
- We assume that smart devices around the user are also equipped with UWB radio. Although one might argue that these assumptions are too strict, we anticipate that UWB radios will permeate the IoT scene in the next few years, given their success and growing adoption rate.

We believe that the next step is to integrate SeleCon in the real-life deployment of many smart homes, collect user reviews and enhance the system architecture. Also, enhancing the pointing recognition accuracy by considering more features is another feasible future work. Supporting multiple users at the same time is a key challenge in the current SeleCon implementation.

## 6.8 Conclusions

In this chapter, we have described and evaluated SeleCon which is a system for IoT device selection and control. SeleCon provides a simple and intuitive interface to interact with a myriad of smart devices through pointing actions and hand gestures. Our approach is the first approach that does pointing detection and gesture recognition without magnetic pre-calibration. All smart devices have to be UWB enabled, and users only need to wear a custom smartwatch equipped with inertial sensors and UWB transceiver. We have designed and implemented hardware prototypes of both the custom smartwatch and the smart devices anchor nodes. SeleCon employs different machine learning classifiers to accurately identify the selected target device from the UWB ranging measurements. In addition, SeleCon supports a language of 11 different gestures to provide control of the selected device. We also presented an energy-saving approach which uses the low-power inertial sensors to trigger UWB such that UWB can be in sleep mode in 92% of the time. Our experimental results demonstrate that SeleCon achieves $84\%$ accuracy for device selection even with a high device deployment density, and our system achieves $97\%$ accuracy for hand gesture recognition.

# 7 Conclusion

To conclude this thesis, we have presented PrOLoc in Chapter 2, a set of algorithms to achieve localization while preserving the privacy of the observers. We have built PrOLoc around the Pallier additive homomorphic cryptosystem, redesigning traditional localization algorithms to benefit from the privacy guarantees of partial homomorphism. Our experiments and timing analyses over both simulated and real, custom range measurement hardware demonstrate that PrOLoc can accurately and efficiently provide localization estimates comparable to traditional, unsecured methods. Our experiments on real hardware demonstrate that PrOLoc yields accurate location estimate at least $500x$ faster than state-of-art secure function evaluation techniques. We then touch upon the problem of distributed localization and its correlated time synchronization problem in Chapter 3. We proposed the algorithms namely, DKAL, DKALarge, and DOPT, that perform distributed estimation in a scalable fashion. Several experiments using real, custom ultra-wideband wireless anchor nodes and mobile quadrotor nodes were conducted and they indicate that the proposed architecture is reliable in localizing static and mobile nodes. Next, we work on the energy-aware aspect of the distributed estimation problem for a multi-sensor system. More specifically, we propose an event-triggered diffusion Kalman filter and apply it to the problem of distributed localization and time synchronization in Chapter 4. Our event-triggered algorithm is the first algorithm that temporarily shuts down the measurement and diffusion steps in the diffusion Kalman filter. Furthermore, we tackle the problem of attacks on the sensor level by proposing an approach for distributed linear secure state estimation in the presence of measurement noise and modeling errors. By combining the diffusion Kalman filter with reachability analysis, we propose a new algorithm for distributed secure state estimation between a network of nodes and we apply the proposed algorithm on the problem of the localization of a rotating target in Chapter 5. Finally, we show an application to accurate localization. More specifically, we tackle the problem of device selection and control in Chapter 6 with the aid of machine learning algorithms and accurate localization. Our approach is the first approach that does pointing detection and gesture recognition without magnetic pre-calibration.

Future work towards realizing the goals laid out in this thesis can be divided into the following thrusts:

- Enhancing the privacy guarantees of the proposed privacy-aware algorithms in Chapter 2.

- Extending the linear distributed secure state estimator in Chapter 5 to the nonlinear case while considering the introduced error due to linearization in the reachability analysis.

- Consider a more powerful attacker who knows the system parameter in Chapter 5.
- Generalize the diffusion strategy to traditional set-membership estimation in Chapter 5.
- Extending the device selection and control in Chapter 6 to multi-users pointing and controlling devices at the same time.
- Fusing inertial-measurement-based path estimates with sparsely available UWB range estimates given only a single anchored device in an indoor localization setting.

# Bibliography


[1] T. Shu, Y. Chen, J. Yang, and A. Williams. Multi-lateral privacy-preserving localization in pervasive environments. In *International Conference on Computer Communications*, pages 2319–2327, April 2014.

[2] J. R. Lowell. Military applications of localization, tracking, and targeting. In *IEEE Wireless Communications*, volume 2, pages 60–65, 2011.

[3] K. Lingemann, H. Surmann, A. Nuchter, and J. Hertzberg. Indoor and outdoor localization for fast mobile robots. In *International Conference on Intelligent Robots and Systems*, volume 3, pages 2185–2190. IEEE, 2004.

[4] J. Slay and M. Miller. Lessons learned from the Maroochy water breach. In *International Conference on Critical Infrastructure Protection*, pages 73–82, 2007.

[5] D. Kushner. The real story of Stuxnet. In *IEEE Spectrum*, volume 50, pages 48–53. Institute of Electrical and Electronics Engineers, 2013.

[6] R. Langner. Stuxnet: Dissecting a cyberwarfare weapon. In *IEEE Security & Privacy*, volume 9, pages 49–51, 2011.

[7] V. M. Igure, S. A. Laughter, and R. D. Williams. Security issues in SCADA networks. In *Computers & Security*, volume 25, pages 498–506, 2006.

[8] D. F. Kune, J. Backes, S. S. Clark, D. Kramer, M. Reynolds, K. Fu, Y. Kim, and W. Xu. Ghost talk: Mitigating EMI signal injection attacks against analog sensors. In *Symposium on Security and Privacy*, pages 145–159. IEEE, 2013.

[9] M. Mohan. *Cybersecurity in drones*. PhD thesis, Utica College, 2016.

[10] N. Shachtman. Computer virus hits US drone fleet. In *CNN.com*, 2011.

[11] A. H. Rutkin. Spoofers use fake GPS signals to knock a yacht off course. In *MIT Technology Review*, 2013.

[12] A. J. Kerns, D. P. Shepard, J. A. Bhatti, and T. E. Humphreys. Unmanned aircraft capture and control via gps spoofing. In *Journal of Field Robotics*, volume 31, pages 617–636. Wiley Online Library, 2014.

[13] A. J. Goldsmith and S. B. Wicker. Design challenges for energy-constrained ad hoc wireless networks. In *IEEE wireless communications*, volume 9, pages 8–27, 2002.

[14] G. J. Pottie and W. J. Kaiser. Wireless integrated network sensors. In *Communications of the ACM*, volume 43, pages 51–58, 2000.

[15] K. Kumar, J. Liu, Y.-H. Lu, and B. Bhargava. A survey of computation offloading for mobile systems. In *Mobile Networks and Applications*, volume 18, pages 129–140, 2013.

[16] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-efficient communication protocol for wireless microsensor networks. In *Proceedings of the 33rd Annual Hawaii International Conference on System Sciences*, volume 2, pages 1–10, 2000.

[17] F. S. Cattivelli, C. G. Lopes, and A. H. Sayed. Diffusion strategies for distributed Kalman filtering: formulation and performance analysis. In *Proceedings Cognitive Information Processing*, pages 36–41, 2008.

[18] A. Alanwar, Y. Shoukry, S. Chakraborty, B. Balaji, P. Martin, P. Tabuada, and M. Srivastava. Demo abstract: Proloc: Resilient localization with private observers using partial homomorphic encryption. In *16th ACM/IEEE International Conference on Information Processing in Sensor Networks*, pages 257–258, April 2017.

[19] A. Alanwar, Y. Shoukry, S. Chakraborty, P. Martin, P. Tabuada, and M. Srivastava. Proloc: Resilient localization with private observers using partial homomorphic encryption. In *16th ACM/IEEE International Conference on Information Processing in Sensor Networks*, pages 41–52, April 2017.

[20] Y. Shoukry, K. Gatsis, A. Alanwar, G. J. Pappas, S. A. Seshia, M. Srivastava, and P. Tabuada. Privacy-aware quadratic optimization using partially homomorphic encryption. In *IEEE 55th Conference on Decision and Control*, pages 5053–5058, Dec 2016.

[21] H. Ferraz, A. Alanwar, M. Srivastava, and J. P. Hespanha. Node localization based on distributed constrained optimization using jacobi's method. In *2017 IEEE 56th Annual Conference on Decision and Control*, pages 3380–3385, Dec 2017.

[22] A. Alanwar, H. Ferraz, K. Hsieh, R. Thazhath, P. Martin, J. Hespanha, and M. Srivastava. D-SLATS: Distributed simultaneous localization and time synchronization. In *Proceedings of the 18th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, page 14. ACM, 2017.

[23] A. Alanwar, M. Alzantot, B. Ho, P. Martin, and M. Srivastava. Selecon: Scalable iot device selection and control using hand gestures. In *IEEE/ACM Second International Conference on Internet-of-Things Design and Implementation*, pages 47–58, April 2017.

[24] A. Alanwar, H. Said, and M. Althoff. Distributed secure state estimation using diffusion kalman filters and reachability analysis. In *2019 IEEE 58th Conference on Decision and Control*, pages 4133–4139, Dec 2019.

[25] A. Alanwar, H. Siad, A. Mehta, and M. Althoff. Event triggered diffusion Kalman filter. *arXiv preprint arXiv:1711.00493*, 2019.

[26] M. T. I. Ziad, A. Alanwar, M. Alzantot, and M. Srivastava. Cryptoimg: Privacy preserving processing over encrypted images. In *Conference on Communications and Network Security*, pages 570–575. IEEE, 2016.

[27] A. Alanwar, B. Etzlinger, H. Ferraz, J. Hespanha, and M. Srivastava. Secsens: Secure state estimation with application to localization and time synchronization. *arXiv preprint arXiv:1801.07132*, 2018.

[28] A. Alanwar, F. Anwar, J. P. Hespanha, and M. Srivastava. Realizing uncertainty-aware timing stack in embedded operating system. In *Processding of the Embedded Operating Systems Workshop*, October 2016. [available online].

[29] A. Alanwar, F. M. Anwar, Y.-F. Zhang, J. Pearson, J. Hespanha, and M. B. Srivastava. Cyclops: PRU programming framework for precise timing applications. In *IEEE International Symposium on Precision Clock Synchronization for Measurement, Control, and Communication*, pages 1–6, 2017.

[30] A. Alanwar, F. Miao, O. Ploder, J. Hespanha, and M. Srivastava. ReCaP: Resilient cyber physical systems against adversarial sensor attacks. [available online].

[31] F. M. Anwar, A. Alanwar, and M. B. Srivastava. OpenClock: A testbed for clock synchronization research. In *2018 IEEE International Symposium on Precision Clock Synchronization for Measurement, Control, and Communication*, pages 1–6, 2018.

[32] A. Alanwar, J. J. Rath, H. Said, and M. Althoff. Distributed set-based observers using diffusion strategy, 2020. `arXiv:2003.10347`.

[33] F. S. Cattivelli and A. H. Sayed. Distributed nonlinear Kalman filtering with applications to wireless localization. In *International Conference on Acoustics, Speech and Signal Processing*, pages 3522–3525. IEEE, 2010.

[34] M. Althoff. *Reachability analysis and its application to the safety assessment of autonomous cars*. PhD thesis, Technische Universität München, 2010.

[35] R. Shokri, G. Theodorakopoulos, J.-Y. Le Boudec, and J.-P. Hubaux. Quantifying location privacy. In *Proceedings of the IEEE Symposium on Security and Privacy*, pages 247–262, 2011.

[36] A. R. Beresford and F. Stajano. Mix zones: User privacy in location-aware services. In *IEEE Annual Conference on Pervasive Computing and Communications*, pages 127–131, 2004.

[37] V. Rastogi and S. Nath. Differentially private aggregation of distributed time-series with transformation and encryption. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data*, pages 735–746, 2010.

[38] E. Shi, R. Chow, T. h. Hubert Chan, D. Song, and E. Rieffel. Privacy-preserving aggregation of time-series data. In *Proceedings of the Network and Distributed System Security Symposium*, 2011. [available online].

[39] M. Naehrig, K. Lauter, and V. Vaikuntanathan. Can homomorphic encryption be practical? In *Proceedings of the 3rd ACM Workshop on Cloud Computing Security Workshop*, pages 113–124, 2011.

[40] P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Proceedings of the 17th International Conference on Theory and Application of Cryptographic Techniques*, pages 223–238, 1999.

[41] C. Gentry. *A Fully Homomorphic Encryption Scheme*. PhD thesis, 2009.

[42] Y. Huang, C.-h. Shen, D. Evans, J. Katz, and A. Shelat. Efficient secure computation with garbled circuits. In S. Jajodia and C. Mazumdar, editors, *Information Systems Security*, volume 7093 of *Lecture Notes in Computer Science*, pages 28–48. Springer Berlin Heidelberg, 2011.

[43] W. Henecka, A.-R. Sadeghi, T. Schneider, I. Wehrenberg, et al. Tasty: tool for automating secure two-party computations. In *Proceedings of the 17th ACM conference on Computer and communications security*, pages 451–462, 2010.

[44] O. Goldreich. *Foundations of cryptography: volume 2, basic applications*. Cambridge university press, 2009.

[45] D. Bogdanov, R. Talviste, and J. Willemson. Deploying secure multi-party computation for financial data analysis. In *International Conference on Financial Cryptography and Data Security*, pages 57–64, 2012.

[46] C. Park, K. Lahiri, and A. Raghunathan. Battery discharge characteristics of wireless sensor nodes: An experimental analysis. In *Second Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks*, pages 430–440, 2005.

[47] C. Dwork. Differential privacy. In *33rd International Colloquium on Automata, Languages and Programming, part II*, volume 4052 of *Lecture Notes in Computer Science*, pages 1–12. Springer Verlag, July 2006.

[48] C. Dwork. Differential privacy: A survey of results. In *Proceedings of the 5th International Conference on Theory and Applications of Models of Computation*, pages 1–19, 2008.

[49] J. C. Duchi, M. I. Jordan, and M. J. Wainwright. Privacy aware learning. In *Journal of ACM*, volume 61, pages 38:1–38:57, December 2014.

[50] N. E. Bordenabe, K. Chatzikokolakis, and C. Palamidessi. Optimal geo-indistinguishable mechanisms for location privacy. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*, pages 251–262, 2014.

[51] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor. Our data, ourselves: Privacy via distributed noise generation. In *Proceedings of the 24th Annual International Conference on The Theory and Applications of Cryptographic Techniques*, pages 486–503, 2006.

[52] A. C. Yao. Protocols for secure computations. In *Proceedings of the 23rd Annual Symposium on Foundations of Computer Science*, pages 160–164, 1982.

[53] Y. Lindell and B. Pinkas. Secure multiparty computation for privacy-preserving data mining. In *Journal of Privacy and Confidentiality*, volume 1, 2009.

[54] S. Goryczka, L. Xiong, and V. Sunderam. Secure multiparty aggregation with differential privacy: A comparative study. In *Proceedings of the Joint EDBT/ICDT Workshops*, pages 155–163, 2013.

[55] R. A. Popa, H. Balakrishnan, and A. J. Blumberg. VPriv: Protecting privacy in location-based vehicular services. In *Proceedings of the 18th Conference on USENIX Security Symposium*, pages 335–350, 2009.

[56] C. Gentry and S. Halevi. Implementing Gentry's fully-homomorphic encryption scheme. In *Proceedings of the 30th Annual International Conference on Theory and Applications of Cryptographic Techniques: Advances in Cryptology*, pages 129–148, 2011.

[57] K. Lauter, A. López-Alt, and M. Naehrig. Private computation on encrypted genomic data. In *International Conference on Cryptology and Information Security in Latin America*, pages 3–27, 2014.

[58] R. Bost, R. A. Popa, S. Tu, and S. Goldwasser. Machine learning classification over encrypted data. In *22nd Annual Network and Distributed System Security Symposium*, 2015.

[59] W. Wang, Y. Hu, L. Chen, X. Huang, and B. Sunar. Exploring the feasibility of fully homomorphic encryption. In *IEEE Transactions on Computers*, volume 64, pages 698–706, 2015.

[60] S. Capkun and J. pierre Hubaux. Secure positioning of wireless devices with application to sensor networks. In *IEEE International Conference on Computer Communications*, page 12, 2005.

[61] S. Capkun, K. Bonne Rasmussen, M. Cagalj, and M. Srivastava. Secure location verification with hidden and mobile base stations. In *IEEE Transactions on Mobile Computing*, volume 7, pages 470–483, 2008.

[62] Q. Mi, J. A. Stankovic, and R. Stoleru. Secure walking gps: A secure localization and key distribution scheme for wireless sensor networks. In *Proceedings of the third ACM Conference on Wireless Network Security*, pages 163–168, 2010.

[63] C. Gentry and S. Halevi. Implementing gentry's fully-homomorphic encryption scheme. In *Proceedings of the 30th Annual International Conference on Theory and Applications of Cryptographic Techniques: Advances in Cryptology*, pages 129–148, 2011.

[64] R. B. Langley. Dilution of precision. In *GPS world*, volume 10, pages 52–59, 1999.

[65] J. Von Neumann. *Functional Operators: The Geometry of Orthogonal Spaces*. Annals of Mathematics Studies. Princeton University Press, 1950.

[66] J. P. Boyle and R. L. Dykstra. A method for finding projections onto the intersection of convex sets in hilbert spaces. In *Advances in order restricted statistical inference*, pages 28–47. Springer, 1986.

[67] S. Mishra, Y. Shoukry, N. Karamchandani, S. N. Diggavi, and P. Tabuada. Secure state estimation against sensor attacks in the presence of noise. In *IEEE Transactions on Control of Network Systems*, volume 4, pages 49–59, 2016.

[68] W. Du and M. J. Atallah. Privacy-preserving cooperative scientific computations. In *IEEE Computer Security Foundations Workshop*, pages 273–282, 2001.

[69] H. Fan, J. Sun, M. Gu, and K.-Y. Lam. Overlap-free Karatsuba-Ofman polynomial multiplication algorithms. In *Information Security*, volume 4, pages 8–14, 2010.

[70] M. Lipiński, T. Włostowski, J. Serrano, and P. Alvarez. White rabbit: A PTP application for robust sub-nanosecond synchronization. In *International IEEE Symposium on Precision Clock Synchronization for Measurement Control and Communication*, pages 25–30, 2011.

[71] G. Regula and B. Lantos. Formation control of a large group of UAVs with safe path planning. In *21st Mediterranean Conference on Control & Automation*, pages 987–993. IEEE, 2013.

[72] P. V. Estrela and L. Bonebakker. Challenges deploying PTPv2 in a global financial company. In *International Symposium on Precision Clock Synchronization for Measurement, Control and Communication*, pages 1–6. IEEE, 2012.

[73] S. Natarajan and A. Ganz. Surgnet: an integrated surgical data transmission system for telesurgery. In *International journal of telemedicine and applications*, volume 2009, page 3. Hindawi Publishing Corp., 2009.

[74] J. C. Corbett, J. Dean, M. Epstein, A. Fikes, C. Frost, J. J. Furman, S. Ghemawat, A. Gubarev, C. Heiser, P. Hochschild, et al. Spanner: Google's globally distributed database. In *ACM Transactions on Computer Systems*, volume 31, page 8, 2013.

[75] Y.-T. Chan, W.-Y. Tsui, H.-C. So, and P.-c. Ching. Time-of-arrival based localization under NLOS conditions. In *IEEE Transactions on Vehicular Technology*, volume 55, pages 17–24, 2006.

[76] C. Zhang, M. Kuhn, B. Merkl, A. E. Fathy, and M. Mahfouz. Accurate UWB indoor localization system utilizing time difference of arrival approach. In *IEEE Radio and Wireless Symposium*, 2006.

[77] F. S. Cattivelli and A. H. Sayed. Distributed nonlinear Kalman filtering with applications to wireless localization. In *International Conference on Acoustics, Speech and Signal Processing*, pages 3522–3525. IEEE, 2010.

[78] U. A. Khan and J. M. Moura. Distributing the Kalman filter for large-scale systems. In *Transactions on Signal Processing*, volume 56, pages 4919–4935, 2008.

[79] J. Medvesek, A. Symington, A. Trost, and S. Hailes. Gaussian process inference approximation for indoor pedestrian localisation. In *Electronics Letters*, volume 51, pages 417–419, 2015.

[80] M. R. Gholami, S. Gezici, and E. G. Strom. TDOA based positioning in the presence of unknown clock skew. In *IEEE Transactions on Communications*, volume 61, pages 2522–2534, 2013.

[81] B. Denis, J.-B. Pierrot, and C. Abou-Rjeily. Joint distributed synchronization and positioning in UWB ad hoc networks using TOA. In *IEEE Transactions on Microwave Theory and Techniques*, volume 54, pages 1896–1911, 2006.

[82] J. Zheng and Y.-C. Wu. Joint time synchronization and localization of an unknown node in wireless sensor networks. In *IEEE Transactions on Signal Processing*, volume 58, pages 1309–1320, 2010.

[83] S. Zhu and Z. Ding. Joint synchronization and localization using toas: A linearization based wls solution. In *IEEE Journal On Selected areas in communications*, volume 28, pages 1017–1025, 2010.

[84] A. W. Weiser, Y. Orchan, R. Nathan, M. Charter, A. J. Weiss, and S. Toledo. Characterizing the accuracy of a self-synchronized reverse-GPS wildlife localization system. In *15th ACM/IEEE International Conference on Information Processing in Sensor Networks*, pages 1–12, 2016.

[85] F. Gustafsson and F. Gunnarsson. Positioning using time-difference of arrival measurements. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 6, pages VI–553, 2003.

[86] Y. Wang, X. Ma, and G. Leus. Robust time-based localization for asynchronous networks. In *IEEE Transactions on Signal Processing*, volume 59, pages 4397–4410, 2011.

[87] D. Dardari, A. Conti, U. Ferner, A. Giorgetti, and M. Win. Ranging with ultra-wide bandwidth signals in multipath environments. In *Proceedings of the IEEE*, volume 97, pages 404–426, Feb 2009.

[88] B. Kempke, P. Pannuto, and P. Dutta. Polypoint: Guiding indoor quadrotors with ultra-wideband localization. In *Proceedings of the 2nd International Workshop on Hot Topics in Wireless*, pages 16–20. ACM, 2015.

[89] A. Ledergerber, M. Hamer, and R. D'Andrea. A robot self-localization system using one-way ultra-wideband communication. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3131–3137, Sept 2015.

[90] J. Tiemann, F. Schweikowski, and C. Wietfeld. Design of an UWB indoor-positioning system for uav navigation in gnss-denied environments. In *International Conference Indoor Positioning and Indoor Navigation*, pages 1–7, Oct 2015.

[91] M. Vossiek, L. Wiebking, P. Gulden, J. Weighardt, and C. Hoffmann. Wireless local positioning-concepts, solutions, applications. In *Radio and Wireless Conference, 2003. RAWCON'03. Proceedings*, pages 219–224. IEEE, 2003.

[92] B. Etzlinger, F. Meyer, A. Springer, F. Hlawatsch, and H. Wymeersch. Cooperative simultaneous localization and synchronization: A distributed hybrid message passing algorithm. In *Asilomar Conference on Signals, Systems and Computers*, pages 1978–1982. IEEE, 2013.

[93] A. D'Amico, L. Taponecco, and U. Mengali. Ultra-wideband TOA estimation in the presence of clock frequency offset. In *IEEE Transactions on Wireless Communications*, volume 12, pages 1606–1616, April 2013.

[94] L. A. Villas, D. L. Guidoni, G. Maia, R. W. Pazzi, J. Ueyama, and A. A. Loureiro. An energy efficient joint localization and synchronization solution for wireless sensor networks using unmanned aerial vehicle. In *Wirel. Netw.*, volume 21, pages 485–498, February 2015.

[95] B. Etzlinger, F. Meyer, H. Wymeersch, F. Hlawatsch, G. Muller, and A. Springer. Cooperative simultaneous localization and synchronization: Toward a low-cost hardware implementation. In *IEEE Sensor Array and Multichannel Signal Processing Workshop*, pages 33–36, 2014.

[96] S. P. Chepuri, G. Leus, and A. van der Veen. Joint localization and clock synchronization for wireless sensor networks. In *Conference Record of the Forty Sixth Asilomar Conference on Signals, Systems and Computers*, pages 1432–1436. IEEE, 2012.

[97] S. Dwivedi, A. D. Angelis, D. Zachariah, and P. Händel. Joint ranging and clock parameter estimation by wireless round trip time measurements. In *CoRR*, volume abs/1501.05450, 2015.

[98] N. Patwari, J. Ash, S. Kyperountas, A. Hero, R. Moses, and N. Correal. Locating the nodes: cooperative localization in wireless sensor networks. In *Signal Processing Magazine, IEEE*, volume 22, pages 54–69, July 2005.

[99] F. Sivrikaya and B. Yener. Time synchronization in sensor networks: a survey. In *Network, IEEE*, volume 18, pages 45–50, July 2004.

[100] P. Martin, A. Symington, and M. Srivastava. Slats: Simultaneous localization and time synchronization. In *ACM Transactions Cyber-Physical Systems*, volume 2, pages 19:1–19:25, June 2018.

[101] Y.-S. Kim and K.-S. Hong. Federated information mode-matched filters in ACC environment. In *International Journal of Control Automation and Systems*, volume 3, pages 173–182, 2005.

[102] G. Bierman. An application of the square root information filter to large-scale linear interconnected systems. In *IEEE Transactions on Automatic Control*, volume 22, pages 989–991, 1977.

[103] T. M. Chin, W. C. Karl, and A. S. Willsky. Sequential filtering for multi-frame visual reconstruction. In *Signal Processing*, volume 28, pages 311–333. Elsevier, 1992.

[104] A. Kavcic and J. M. Moura. Matrices with banded inverses: Inversion algorithms and factorization of gauss-markov processes. In *IEEE Transactions on Information Theory*, volume 46, pages 1495–1509, 2000.

[105] A. B. Frakt, H. Lev-Ari, and A. S. Willsky. A generalized levinson algorithm for covariance extension with application to multiscale autoregressive modeling. In *IEEE Transactions on Information Theory*, volume 49, pages 411–424, 2003.

[106] R. Grone, C. R. Johnson, E. M. Sá, and H. Wolkowicz. Positive definite completions of partial hermitian matrices. In *Linear algebra and its applications*, volume 58, pages 109–124, 1984.

[107] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng. ROS: an open-source robot operating system. In *International Conference on Robotics and Automation, Workshop on Open Source Software*, 2009.

[108] J. Aspnes, D. Goldenberg, and Y. R. Yang. On the computational complexity of sensor network localization. In *Algorithmic aspects of wireless sensor networks*, pages 32–44. Springer, 2004.

[109] B. Jackson and T. Jordán. Connected rigidity matroids and unique realizations of graphs. In *Journal of Combinatorial Theory, Series B*, volume 94, pages 1–29, 2005.

[110] P. Zhang and Q. Wang. On using the relative configuration to explore cooperative localization. In *IEEE Transactions on Signal Processing*, volume 62, pages 968–980, Feb 2014.

[111] M. Maróti, B. Kusy, G. Simon, and A. Lédeczi. The flooding time synchronization protocol. In *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems*, pages 39–49. ACM, 2004.

[112] C. Lenzen, P. Sommer, and R. Wattenhofer. PulseSync: An efficient and scalable clock synchronization protocol. In *IEEE/ACM Transactions on Networking*, volume 23, pages 717–727, 2015.

[113] J. Yick, B. Mukherjee, and D. Ghosal. Wireless sensor network survey. In *Computer networks*, volume 52, pages 2292–2330. Elsevier, 2008.

[114] C. G. Lopes and A. H. Sayed. Diffusion least-mean squares over adaptive networks: Formulation and performance analysis. In *IEEE Transactions on Signal Processing*, volume 56, pages 3122–3136, 2008.

[115] F. S. Cattivelli, C. G. Lopes, and A. H. Sayed. Diffusion recursive least-squares for distributed estimation over adaptive networks. In *IEEE Transactions on Signal Processing*, volume 56, pages 1865–1877, 2008.

[116] F. S. Cattivelli and A. H. Sayed. Diffusion strategies for distributed Kalman filtering and smoothing. In *IEEE Transactions on Automatic Control*, volume 55, pages 2069–2084, 2010.

[117] L. Xiao, S. Boyd, and S. Lall. A space-time diffusion scheme for peer-to-peer least-squares estimation. In *Proceedings of the 5th international conference on Information Processing in Sensor Networks*, pages 168–176. ACM, 2006.

[118] I. D. Schizas, G. B. Giannakis, S. I. Roumeliotis, and A. Ribeiro. Consensus in ad hoc WSNs with noisy links—part ii: Distributed estimation and smoothing of random signals. In *IEEE Transactions on Signal Processing*, volume 56, pages 1650–1666, 2008.

[119] R. Olfati-Saber. Distributed Kalman filtering for sensor networks. In *Conference on Decision and Control*, pages 5492–5498. IEEE, 2007.

[120] U. A. Khan and J. M. Moura. Distributing the Kalman filter for large-scale systems. In *IEEE Transactions on Signal Processing*, volume 56, pages 4919–4935, 2008.

[121] R. Grone, C. R. Johnson, E. M. Sá, and H. Wolkowicz. Positive definite completions of partial hermitian matrices. In *Linear algebra and its applications*, volume 58, pages 109–124. Elsevier, 1984.

[122] Y. S. Suh, V. H. Nguyen, and Y. S. Ro. Modified Kalman filter for networked monitoring systems employing a send-on-delta method. volume 43, pages 332 – 338, 2007.

[123] J. Wu, Q.-S. Jia, K. H. Johansson, and L. Shi. Event-based sensor data scheduling: Trade-off between communication rate and estimation quality. In *IEEE Transactions on Automatic Control*, volume 58, pages 1041–1046, 2013.

[124] S. Trimpe and R. D'Andrea. Event-based state estimation with variance-based triggering. In *IEEE Transactions on Automatic Control*, volume 59, pages 3266–3281, 2014.

[125] D. Shi, T. Chen, and L. Shi. On set-valued Kalman filtering and its application to event-based state estimation. In *IEEE Transactions on Automatic Control*, volume 60, pages 1275–1290, 2015.

[126] L. Li, M. Lemmon, and X. Wang. Event-triggered state estimation in vector linear processes. In *American Control Conference*, pages 2138–2143. IEEE, 2010.

[127] L. Li, Z. Wang, and M. Lemmon. Polynomial approximation of optimal event triggers for state estimation problems using sostools. In *American Control Conference*, pages 2699–2704. IEEE, 2013.

[128] L. Groff, L. Moreira, J. G. da Silva, and D. Sbarbaro. Observer-based event-triggered control: A discrete-time approach. In *American Control Conference*, pages 4245–4250. IEEE, 2016.

[129] D. N. Borgers and W. M. Heemels. Event-separation properties of event-triggered control systems. In *IEEE Transactions on Automatic Control*, volume 59, pages 2644–2656, 2014.

[130] D. Shi, L. Shi, and T. Chen. Linear gaussian systems and event-based state estimation. In *Event-Based State Estimation*, pages 33–46. Springer, 2016.

[131] A. Molin, H. Sandberg, and K. H. Johansson. Consistency-preserving event-triggered estimation in sensor networks. In *Conference on Decision and Control*, pages 7494–7501. IEEE, 2015.

[132] G. Battistelli, L. Chisci, and D. Selvi. Distributed Kalman filtering with data-driven communication. In *19th International Conference on Information Fusion*, pages 1042–1048. IEEE, 2016.

[133] W. Li, Y. Jia, and J. Du. Event-triggered Kalman consensus filter over sensor networks. In *IET Control Theory & Applications*, volume 10, pages 103–110, 2016.

[134] C. Zhang and Y. Jia. Distributed Kalman consensus filter with event-triggered communication: Formulation and stability analysis. In *Journal of the Franklin Institute*, volume 354, pages 5486–5502, 2017.

[135] X. Wang and M. D. Lemmon. Event-triggering in distributed networked control systems. volume 56, pages 586–601. IEEE, 2010.

[136] S. Trimpe. Distributed event-based state estimation. In *arXiv preprint arXiv:1511.05223*, 2015.

[137] L. Zou, Z.-D. Wang, and D.-H. Zhou. Event-based control and filtering of networked systems: A survey. In *International Journal of Automation and Computing*, pages 1–15. Springer, 2017.

[138] V. Vahidpour, A. Rastegarnia, A. Khalili, W. Bazzi, and S. Sanei. Partial diffusion Kalman filtering. In *arXiv preprint arXiv:1705.08920*, 2017.

[139] Y. Chen, G. Qi, Y. Li, and A. Sheng. Diffusion Kalman filtering with multi-channel decoupled event-triggered strategy and its application to the optic-electric sensor network. In *Information Fusion*, volume 36, pages 233–242. Elsevier, 2017.

[140] A. H. Sayed. *Fundamentals of adaptive filtering*. John Wiley & Sons, 2003.

[141] F. S. Cattivelli and A. H. Sayed. Diffusion strategies for distributed Kalman filtering and smoothing. In *IEEE Transactions on Automatic Control*, volume 55, pages 2069–2084, 2010.

[142] L. Ljung. Asymptotic behavior of the extended Kalman filter as a parameter estimator for linear systems. In *IEEE Transactions on Automatic Control*, volume 24, pages 36–50, 1979.

[143] M. Uma and G. Padmavathi. A survey on various cyber attacks and their classification. In *I. J. Network Security*, volume 15, pages 390–396, 2013.

[144] H. Fawzi, P. Tabuada, and S. Diggavi. Secure estimation and control for cyber-physical systems under adversarial attacks. In *IEEE Transactions on Automatic Control*, volume 59, pages 1454–1467, 2014.

[145] M. S. Chong, M. Wakaiki, and J. P. Hespanha. Observability of linear systems under adversarial attacks. In *American Control Conference*, pages 2439–2444. IEEE, 2015.

[146] M. Pajic, J. Weimer, N. Bezzo, P. Tabuada, O. Sokolsky, I. Lee, and G. J. Pappas. Robustness of attack-resilient state estimators. In *ACM/IEEE 5th International Conference on Cyber-Physical Systems*, pages 163–174, 2014.

[147] F. Pasqualetti, F. Dorfler, and F. Bullo. Control-theoretic methods for cyberphysical security: Geometric principles for optimal cross-layer resilient control systems. In *IEEE Control Systems*, volume 35, pages 110–127, 2015.

[148] C.-Z. Bai, V. Gupta, and F. Pasqualetti. On Kalman filtering with compromised sensors: Attack stealthiness and performance bounds. In *IEEE Transactions on Automatic Control*, volume 62, pages 6641–6648, 2017.

[149] C.-Z. Bai and V. Gupta. On Kalman filtering in the presence of a compromised sensor: Fundamental performance bounds. In *American Control Conference, 2014*, pages 3029–3034. IEEE, 2014.

[150] Y. Mo and B. Sinopoli. Secure control against replay attacks. In *47th annual Allerton conference on communication, control, and computing*, pages 911–918. IEEE, 2009.

[151] F. Miao, M. Pajic, and G. J. Pappas. Stochastic game approach for replay attack detection. In *52nd Conference on Decision and Control*, pages 1854–1859. IEEE, 2013.

[152] S. Amin, A. A. Cárdenas, and S. Sastry. Safe and secure networked control systems under denial-of-service attacks. In *ACM International Conference on Hybrid Systems: Computation and Control*, pages 31–45, 2009.

[153] Y. Mo, E. Garone, A. Casavola, and B. Sinopoli. False data injection attacks against state estimation in wireless sensor networks. In *49th Conference on Decision and Control*, pages 5967–5972. IEEE, 2010.

[154] F. Pasqualetti, F. Dörfler, and F. Bullo. A divide-and-conquer approach to distributed attack identification. In *54th IEEE Conference on Decision and Control*, pages 5801–5807, 2015.

[155] W. Ao, Y. Song, and C. Wen. Distributed secure state estimation and control for CPSs under sensor attacks. In *IEEE Transactions on Cybernetics*, pages 1–11. IEEE, 2018. [available online].

[156] L. An and G.-H. Yang. Distributed secure state estimation for cyber–physical systems under sensor attacks. In *Automatica*, volume 107, pages 526 – 538, 2019.

[157] F. S. Cattivelli, C. G. Lopes, and A. H. Sayed. Diffusion strategies for distributed Kalman filtering: formulation and performance analysis. In *Proceedings Cognitive Information Processing*, pages 36–41, 2008.

[158] Y. Wang, V. Puig, and G. Cembrano. Set-membership approach and Kalman observer based on zonotopes for discrete-time descriptor systems. In *Automatica*, volume 93, pages 435–443. Elsevier, 2018.

[159] T. Alamo, J. M. Bravo, and E. F. Camacho. Guaranteed state estimation by zonotopes. In *Automatica*, volume 41, pages 1035–1043. Elsevier, 2005.

[160] A. Girard. Reachability of uncertain linear systems using zonotopes. In *International Workshop on Hybrid Systems: Computation and Control*, pages 291–305, 2005.

[161] M. Althoff. An introduction to CORA 2015. In *Proceedings of the Workshop on Applied Verification for Continuous and Hybrid Systems*, pages 120–151, 2015.

[162] M. Althoff and D. Grebenyuk. Implementation of interval arithmetic in CORA 2016. In *Proceedings of the 3rd International Workshop on Applied Verification for Continuous and Hybrid Systems*, pages 145–173, 2016.

[163] M. Althoff, D. Grebenyuk, and N. Kochdumper. Implementation of taylor models in CORA 2018. In *Proceedings of the 5th International Workshop on Applied Verification for Continuous and Hybrid Systems*, pages 91–105, 2018.

[164] F. Alonso-Martín, A. Castro-González, F. J. F. d. G. Luengo, and M. Á. Salichs. Augmented robotics dialog system for enhancing human–robot interaction. In *Sensors*, volume 15, pages 15799–15829. Multidisciplinary Digital Publishing Institute, 2015.

[165] W. O. Lee, Y. G. Kim, H. G. Hong, and K. R. Park. Face recognition system for set-top box-based intelligent tv. In *Sensors*, volume 14, pages 21726–21749. Multidisciplinary Digital Publishing Institute, 2014.

[166] A. Lopez-Basterretxea, A. Mendez-Zorrilla, and B. Garcia-Zapirain. Eye/head tracking technology to improve hci with ipad applications. In *Sensors*, volume 15, pages 2244–2264. Multidisciplinary Digital Publishing Institute, 2015.

[167] F.-S. Chen, C.-M. Fu, and C.-L. Huang. Hand gesture recognition using a real-time tracking method and hidden markov models. In *Image and vision computing*, volume 21, pages 745–758. Elsevier, 2003.

[168] B. Michoud, E. Guillou, and S. Bouakaz. Real-time and markerless 3D human motion capture using multiple views. In *Human Motion–Understanding, Modeling, Capture and Animation*, pages 88–103. Springer, 2007.

[169] F. Adib, Z. Kabelac, D. Katabi, and R. C. Miller. 3D tracking via body radio reflections. In *Proceedings of the 11th USENIX Conference on Networked Systems Design and Implementation*, pages 317–329, 2014.

[170] Q. Pu, S. Gupta, S. Gollakota, and S. Patel. Whole-home gesture recognition using wireless signals. In *Proceedings of the 19th annual international conference on Mobile computing & networking*, pages 27–38. ACM, 2013.

[171] H. Wen, J. Ramos Rojas, and A. K. Dey. Serendipity: Finger gesture recognition using an off-the-shelf smartwatch. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pages 3847–3851. ACM, 2016.

[172] G. Cohn, D. Morris, S. Patel, and D. Tan. Humantenna: using the body as an antenna for real-time whole-body interaction. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1901–1910. ACM, 2012.

[173] H. Junker, P. Lukowicz, and G. Tröster. Continuous recognition of arm activities with body-worn inertial sensors. In *International Semantic Web Conference*, pages 188–189, 2004.

[174] F. Adib and D. Katabi. See through walls with WiFi! In *Conference on Special Interest Group on Data Communication*, pages 75–86. ACM, 2013.

[175] K. Bouchard, A. Bouzouane, and B. Bouchard. Gesture recognition in smart home using passive rfid technology. In *Proceedings of the 7th International Conference on PErvasive Technologies Related to Assistive Environments*, pages 12:1–12:8. ACM, 2014.

[176] P. Asadzadeh, L. Kulik, and E. Tanin. Gesture recognition using rfid technology. In *Personal and Ubiquitous Computing*, volume 16, pages 225–234. Springer, 2012.

[177] S. Gupta, D. Morris, S. Patel, and D. Tan. Soundwave: Using the doppler effect to sense gestures. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1911–1914. ACM, 2012.

[178] K. Ali, A. X. Liu, W. Wang, and M. Shahzad. Keystroke recognition using WiFi signals. In *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*, pages 90–102. ACM, 2015.

[179] H. Ketabdar, P. Moghadam, B. Naderi, and M. Roshandel. Magnetic signatures in air for mobile devices. In *Proceedings of the 14th International Conference on Human-computer Interaction with Mobile Devices and Services Companion*, pages 185–188. ACM, 2012.

[180] C. Bo, X. Jian, X.-Y. Li, X. Mao, Y. Wang, and F. Li. You're driving and texting: detecting drivers using personal smart phones by leveraging inertial sensors. In

*Proceedings of the 19th annual international conference on Mobile computing & networking*, pages 199–202. ACM, 2013.

[181] S. Agrawal, I. Constandache, S. Gaonkar, R. Roy Choudhury, K. Caves, and F. DeRuyter. Using mobile phones to write in air. In *Proceedings of the 9th International Conference on Mobile Systems, Applications, and Services*, pages 15–28. ACM, 2011.

[182] B. D. Mayton, N. Zhao, M. Aldrich, N. Gillian, and J. A. Paradiso. Wristque: A personal sensor wristband. In *IEEE International Conference on Body Sensor Networks*, pages 1–6, 2013.

[183] L. Porzi, S. Messelodi, C. M. Modena, and E. Ricci. A smart watch-based gesture recognition system for assisting people with visual impairments. In *Proceedings of the 3rd ACM international workshop on Interactive multimedia on mobile & portable devices*, pages 19–24, 2013.

[184] P. Jing and G. Ye-peng. Human-computer interaction using pointing gesture based on an adaptive virtual touch screen. In *International Journal of Signal Processing, Image Processing and Pattern Recognition*, volume 6, pages 81–92, 2013.

[185] C. Xu, P. H. Pathak, and P. Mohapatra. Finger-writing with smartwatch: A case for finger and hand gesture recognition using smartwatch. In *Proceedings of the 16th International Workshop on Mobile Computing Systems and Applications*, pages 9–14. ACM, 2015.

[186] J. Rekimoto. Gesturewrist and gesturepad: Unobtrusive wearable interaction devices. In *Fifth International Symposium on Wearable Computers*, pages 21–27. IEEE, 2001.

[187] S. Shen, H. Wang, and R. Roy Choudhury. I am a smartwatch and i can track my user's arm. In *Proceedings of the 14th Annual International Conference on Mobile Systems, Applications, and Services*, pages 85–96. ACM, 2016.

[188] T. Park, J. Lee, I. Hwang, C. Yoo, L. Nachman, and J. Song. E-gesture: A collaborative architecture for energy-efficient gesture recognition with hand-worn sensor and mobile devices. In *Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems*, pages 260–273, 2011.

[189] L. Ardüser, P. Bissig, P. Brandes, and R. Wattenhofer. Recognizing text using motion data from a smartwatch. In *2016 IEEE International Conference on Pervasive Computing and Communication Workshops*, pages 1–6, 2016.

[190] X. Zhang, X. Chen, W.-h. Wang, J.-h. Yang, V. Lantz, and K.-q. Wang. Hand gesture recognition and virtual game control based on 3D accelerometer and emg sensors. In *Proceedings of the 14th International Conference on Intelligent User Interfaces*, pages 401–406. ACM, 2009.

[191] J. Han, E. Owusu, L. T. Nguyen, A. Perrig, and J. Zhang. Accomplice: Location inference using accelerometers on smartphones. In *Fourth International Conference on Communication Systems and Networks*, pages 1–9. IEEE, 2012.

[192] P. Marquardt, A. Verma, H. Carter, and P. Traynor. (sp) iphone: decoding vibrations from nearby keyboards using mobile phone accelerometers. In *Proceedings of the 18th ACM conference on Computer and communications security*, pages 551–562, 2011.

[193] E. Owusu, J. Han, S. Das, A. Perrig, and J. Zhang. Accessory: password inference using accelerometers on smartphones. In *Proceedings of the Twelfth Workshop on Mobile Computing Systems & Applications*, page 9. ACM, 2012.

[194] G. Wang, Y. Zou, Z. Zhou, K. Wu, and L. M. Ni. We can hear you with Wi-Fi! In *Proceedings of the 20th annual international conference on Mobile computing and networking*, pages 593–604. ACM, 2014.

[195] H. Li, W. Yang, J. Wang, Y. Xu, and L. Huang. WiFinger: Talk to your smart devices with finger-grained gesture. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pages 250–261, 2016.

[196] Y. Kim and H. Ling. Human activity classification based on micro-doppler signatures using a support vector machine. In *IEEE Transactions on Geoscience and Remote Sensing*, volume 47, pages 1328–1337, 2009.

[197] W. Xi, J. Zhao, X.-Y. Li, K. Zhao, S. Tang, X. Liu, and Z. Jiang. Electronic frog eye: Counting crowd using WiFi. In *International Conference on Computer Communications*, pages 361–369. IEEE, 2014.

[198] C. Han, K. Wu, Y. Wang, and L. M. Ni. WiFall: Device-free fall detection by wireless networks. In *International Conference on Computer Communications*, pages 271–279. IEEE, 2014.

[199] Y. Wang, J. Liu, Y. Chen, M. Gruteser, J. Yang, and H. Liu. E-eyes: Device-free location-oriented activity identification using fine-grained WiFi signatures. In *Proceedings of the 20th Annual International Conference on Mobile Computing and Networking*, pages 617–628. ACM, 2014.

[200] R. Nandakumar, B. Kellogg, and S. Gollakota. Wi-Fi gesture recognition on existing devices. In *arXiv preprint arXiv:1411.5394*, 2014.

[201] S. Gezici, Z. Tian, G. B. Giannakis, H. Kobayashi, A. F. Molisch, H. V. Poor, and Z. Sahinoglu. Localization via ultra-wideband radios: a look at positioning aspects for future sensor networks. In *IEEE signal processing magazine*, volume 22, pages 70–84, 2005.

[202] J. M. Rehg and T. Kanade. Visual tracking of high dof articulated structures: An application to human hand tracking. In *Proceedings of the third European Conference-Volume II on Computer Vision - Volume II*, pages 35–46. Springer-Verlag, 1994.

[203] T. Starner and A. Pentland. Real-time american sign language recognition from video using hidden markov models. In *Motion-Based Recognition*, pages 227–243. Springer, 1997.

[204] E. Larson, G. Cohn, S. Gupta, X. Ren, B. Harrison, D. Fox, and S. N. Patel. Heatwave: thermal imaging for surface user interaction. In *In Proceedings of CHI 2011*, pages 2565–2574.

[205] B.-J. Ho, H.-L. C. Kao, N.-C. Chen, C.-W. You, H.-H. Chu, and M.-S. Chen. Heatprobe: a thermal-based power meter for accounting disaggregated electricity usage. In *Proceedings of the 13th international conference on Ubiquitous computing*, pages 55–64. ACM, 2011.