Collaborative Localization Using Wireless Sensor Networks

A Dissertation

Presented to

The Faculty of the School of Engineering and Applied Science
University of Virginia

In Partial Fulfillment

of the Requirements for the Degree

Doctor of Philosophy in Computer Science

by

Zhiheng Xie

May, 2016

Xie, Zhiheng

Collaborative Localization Using Wireless Sensor Networks

University of Virginia, Ph.D., May 2016

Approval Sheet

The dissertation is submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Computer Science

Zhiheng Xie

This dissertation has been read and approved by the Examining Committee:

John Stankovic, Ph.D., Dissertation Advisor

Kamin Whitehouse, Ph.D., Committee Chairman

Alfred Weaver, Ph.D., Committee Member

Gabriel Robins, Ph.D., Committee Member

John Lach, Ph.D., Committee Member

Accepted for the School of Engineering and Applied Science:

Dean, School of Engineering
and Applied Science
May, 2016

Abstract

Wireless sensor networks (WSN) are now widely used in many applications. Knowing each sensor node's position is always a critical issue for these applications. Without the position information, the sensory data become meaningless, and a number of location based routing protocols would not work.

In the WSN research field, the existing localization solutions can be divided into range-based solutions and range-free solutions. Despite which method is used, the existing solutions suffer from one or more of the following drawbacks: 1) only including homogeneous positioning information; 2) only suitable for static WSNs, but not mobile WSNs; 3) requiring pre-set infrastructures; 4) non-deterministic uncertainty. Another category of mobile node localization methods are from the robotics field. The representatives are simultaneous localization and mapping (SLAM)and the collaborative localization (CL). However, SLAM solution is not suitable for resource constrained WSN because of its iterative dynamic model, high computational complexity and not taking advantage the collaboration between nodes. The existing CL solutions suffer from either inefficiency or the over confidence problem.

This dissertation proposes a unified range-based localization mathematical model, which can be applied to a large scale static WSN, to a large scale WSN with dynamic topology changes, or to a mobile WSN. When designing such a model, the following purposes are addressed: 1) unified representation of various measurement types; 2) suitable for both mobile and static networks; 4) both centralized and decentralized architecture support; 5) providing not only position estimation, but also the quantitative uncertainty (the covariance matrix) of the estimation; 6) efficiency and scalability. The effectiveness and the efficiency of this model are demonstrated by a decentralized and fully self-contained indoor pedestrian localization system.

We first propose the incremental node-voltage analysis (INOVA) localization algorithm, which is used to localize a stationary WSN in a centralized way. INOVA analogizes a WSN to a generalized electrical network, and borrows the node-voltage analysis from the electrical engineering field to reduce the computational complexity by 70 times (comparing with the optimization technique based solution–Best Linear Unbiased Estimator). Since INOVA is fast on updating, it is suitable to localize a large scale static WSN with frequently dynamic changes or a mobile network. By using the same idea, an overlapping subgraph estimator of covariance (OSE-COV) algorithm is proposed. Together with the original overlapping subgraph estimator (OSE) algorithm, it is able to estimate both the positions and the covariance matrices of sensor nodes in a decentralized way. In order to localize mobile nodes in even a more efficient way, the elastic decentralized collaborative localization (EDCL) algorithm is proposed. Different form the above two algorithms, which are theoretically optimal and asymptotically optimal respectively, EDCL is a non-optimal solution. By controlling the marginalization factor Q, which indicates the number of the historical measurements allowed to store in memory, EDCL is able to make the trade-off between the optimality and the resource consumption. Besides the localization theory, we also deduce how to use the quantitative uncertainty for confidence region inference and guiding the anchor selection process.

To demonstrate the effectiveness of the EDCL algorithm, a hybrid pedestrian collaborative localization system is built. The system is fully decentralized self-contained. It consists of three modules, a RSS to distance estimator, a foot-mounted inertial navigation module, and the EDCL filter. The experimental results show that EDCL reduces the error by as much as 49.44% over the inertial-only solution, and the resource consumption is low (with maximum memory usage 760 bytes and average communication cost 33 bytes per message).

Acknowledgments

Thirty five years ago, my mom, Shanmei Chen, brought me to this world, and began her journey on loving and supporting me. She is a passionate and kind educator, and gives her all to the education field and raising me. Whenever I read her writings, I experience how deeply she loves me, and how much faith she has in me. As I finish my Ph.D. dissertation, I want to say: Mom, this is in honor of you. I hope you will always be proud of me. I will preserve your love and spirit in my heart forever. I am deeply grateful to my dad, Qinglin Xie, who always loves me in his traditional Chinese way. Without his patience and support, I would have not finished this dissertation.

I am extremely thankful to my advisor, Professor John A. Stankovic. He is always patient, and encourages and inspires me all the time. He is my role model both academically and in life. I also want to take the opportunity to thank my committee members, Professors Alfred Weaver, Gabriel Robins, John Lach and Kamin Whitehouse, who have given me a lot of support and helpful feedback.

I am glad that I have many good friends who truly care about me. Hengchang Liu is like a brother, encouraging and trusting me. Without him, it would have been be a lonely journey. Keye Sun, Chunhu Zhang, Guangfu Wang and Wenlu Sun spent a lot of their time helping out with my experiments. Wei Zheng worked with me on many difficult math problems. Jiguang Li, Baile Chen, Huilin Li, Yiran Xia and Qiang Li kept me company when I felt discouraged.

Last and most importantly, I am very grateful to my girlfriend Rica, who is next to me during my hardest times. She always trusts and supports me. Without her, I cannot be where I am today.

Acronyms

AOA	Angle of Arrival.
AP	Access Point.
BLUE	Best Linear Unbiased Estimator.
CDL	Combined and Differentiated Localization.
CL	Collaborative Localization.
COTS	Commercial Off-The-Shelf.
CSMA	Carrier Sense Multiple Access.
DSSS	Direct-Sequence Spread-Spectrum.
DV	Distance Vector.
EDCL	Elastic Decentralized Collaborative Localization.
EIF	Extended Information Filter.
EKF	Extended Kalman Filter.
EMG	Extended Measurement Graph.
FPGA	Field-Programmable Gate Array.
GNSS	Global Navigation Satellite System.
GPS	Global Positioning system.
IMU	Inertial Measurement Unit.
INOVA	Incremental Node-Voltage Analysis.
INS	Inertial Navigation System.
IRG	Information Relation Graph.
LLE	Locally Linear Embedding.
LP	Linear Programming.
LS	Least Square.
MEMS	Microelectromechanical systems
MDS	Multidimensional Scaling.
OSE	Overlapping Subgraph Estimator.
OSE-COV	Overlapping Subgraph Estimator of Covariance Version.
PN	Pseudorandom Noise.
RF	Radio Frequency.
RFID	Radio-frequency identification.
RI	Radio Interferometry.
RMG	Relative Measurement Graph.
RSS	Received Signal Strength.
RSSI	Received Signal Strength Indicator.
SDP	Semidefinite Programming.
SLAM	Simultaneous Localization and Mapping.
SISR	Snap-Inducing Shaped Residuals.
SLAM	Simultaneous Localization and Mapping.
SNR	Signal to Noise Ratio.
TDOA	Time Difference of Arrival.
TOA	Time of Arrival.
TOF	Time of Flight.

UWB	Ultra-Wideband.
WSN	Wireless Sensor Network.
ZUPT	zero-velocity-update
ZARU	zero-angular-rate-update

Contents

List of Figures **xiii**

List of Tables **xvii**

1 Introduction **1**

1.1 Motivation . . . 1

1.2 Challenges . . . 6

1.3 Contributions . . . 7

1.3.1 INOVA and OSE-COV . . . 8

1.3.2 Confidence Region Inference and Anchor Selection Strategy . . . 9

1.3.3 EDCL . . . 10

1.3.4 Infrastructure Free Indoor Pedestrian Localization System . . . 11

1.4 Organization . . . 12

2 State of the Art **13**

2.1 Localization with Wireless Sensor Networks . . . 14

2.1.1 Range Based Localization . . . 14

2.1.1.1 Received Signal Strength Ranging based Localization . . . 14

2.1.1.2 Time of Flight based Localization . . . 18

2.1.1.3 Angle of Arrival based Localization . . . 22

2.1.1.4 Radio interferometry based Localization . . . 25

2.1.2 Range Free Localization . . . 28

2.1.2.1 Anchor Proximity . . . 29

2.1.2.2 Network Connectivity . . . 31

2.1.2.3 Event Based Localization . . . 37

2.2 Localization in Robotics . . . 42

2.2.1 SLAM . . . 42

2.2.1.1 SLAM Modeling, Convergence Analysis and Classical Solutions . . . 43

2.2.1.2 Algorithms . . . 47

2.2.1.3 Computation efficiency . . . 53

2.2.1.4 Data association . . . 54

2.2.2 Collaborative Localization . . . 55

2.3 Indoor Hybrid Localization Systems . . . 58

2.3.1 Inertial Pedestrian Navigation and Infrastructure based Indoor Localization . . . 60
2.3.1.1 Inertial Pedestrian Navigation . . . 61
2.3.1.2 Infrastructure based Indoor Localization . . . 63
2.3.2 Localization with Inertial Devices and Infrastructure . . . 65
2.3.3 Localization with Inertial Devices and Map . . . 67
2.3.4 Localization with Inertial Devices, Infrastructure and Map . . . 69

3 Stationary Wireless Sensor Network Localization 71
3.1 System Formalization . . . 73
3.1.1 Measurement Graph . . . 75
3.2 Electrical Network Analogy . . . 76
3.2.1 Linear Case–BLUE Algorithm . . . 76
3.2.2 Linear Case–INOVA Algorithm . . . 80
3.2.2.1 The Analogy Between A Measurement Graph and A Generalized Electrical Network . . . 81
3.2.2.2 The INOVA Algorithm . . . 84
3.2.3 Extension to The Nonlinear Case . . . 86
3.2.3.1 Nonlinear case . . . 87
3.2.3.2 Unifying Anchor Nodes and Non-anchor Nodes . . . 89
3.3 Decentralized Algorithm . . . 91
3.3.1 OSE-COV for One Node Identity Current Graph . . . 92
3.3.2 The OSE-COV Algorithm . . . 96
3.3.2.1 Calculating Each Node's Voltage Potential in Each Node's Identity Current Graph . . . 97
3.3.2.2 Obtaining the Knowledge of All Nodes . . . 98
3.3.2.3 The OSE-COV Algorithm Summary and Analysis . . . 100
3.4 Evaluation . . . 102
3.4.1 Inova Localization Accuracy . . . 103
3.4.2 Linearization Analysis . . . 106
3.4.3 The efficiency of Inova . . . 108
3.4.4 The localization accuracy and convergence speed of OSE-COV . . . 110
3.5 Discussion . . . 112

4 Quantitative Uncertainty Analysis and Usage 114
4.1 Normality Proof . . . 114
4.2 Anchor Selection Application . . . 116
4.3 Evaluation . . . 119
4.4 Conclusion . . . 121

5 Mobile Sensor Network Localization 123
5.1 System Formalization and Challenges . . . 125
5.1.1 System Formalization . . . 125
5.1.2 The Benefits and Challenges of CL . . . 127
5.2 EDCL Overview and Extended Measurement Graph . . . 129

5.3 Information Relation Graph and Its Operations . . . 131
5.3.1 Update, Augmentation and Marginalization . . . 132
5.3.2 Information Relation Graph . . . 134
5.4 The Elastic Decentralized Collaborative Localization Algorithm . . . 135
5.4.1 DR and GPS Measurement Update . . . 135
5.4.2 Inter-agent Measurement Update . . . 136
5.4.3 Marginalization . . . 139
5.5 EDCL Analysis . . . 140
5.6 Evaluation . . . 141
5.6.1 Simple Simulation . . . 141
5.6.2 Taxi Localization . . . 144
5.7 Discussion . . . 145

6 Indoor Pedestrian Localization System 147
6.1 System Overview . . . 148
6.2 Inter-agent RSSI Distance Estimator . . . 150
6.2.1 RSSI study . . . 150
6.2.2 The RSSI to Distance Estimation Module . . . 155
6.2.2.1 The Corrected Maximum Likelihood Distance Estimator . 156
6.2.2.2 The Corrected Flattened Maximum Likelihood Distance Estimator . . . 159
6.2.2.3 The Corrected Truncated Maximum Likelihood Distance Estimator . . . 161
6.2.3 Frequency Hopping TDMA . . . 163
6.2.3.1 Protocol Time Setting . . . 165
6.2.3.2 Master Node Selection . . . 168
6.2.3.3 Network Dynamics and Protocol Packet format . . . 168
6.2.4 Evaluation . . . 170
6.2.4.1 Configuration . . . 170
6.2.4.2 One Channel versus Channel Hopping . . . 171
6.2.4.3 Estimator Performances Evaluation . . . 174
6.3 Foot-mounted Inertial Navigation System . . . 179
6.3.1 Zero Velocity Update Methodology . . . 180
6.3.1.1 The Mechanization Model . . . 181
6.3.1.2 The Zero Velocity Detector . . . 183
6.3.2 IMU Calibration and ZUPT Detector Threshold Setting . . . 184
6.3.2.1 IMU Characterization . . . 184
6.3.2.2 ZUPT Detector Threshold Setting . . . 190
6.3.3 Evaluation . . . 191
6.4 The System Integration . . . 194
6.4.1 Packet Format and Time Synchronization . . . 196
6.4.2 Measurement Generation . . . 199
6.5 Evaluation . . . 202
6.5.1 Localization Accuracy and Uncertainty . . . 204
6.5.2 EDCL Resource Usage Evaluation . . . 211

6.5.3 Conclusion 218
6.6 Discussion 219

7 Conclusions 221
7.1 Summary of Dissertation 221
7.2 Discussion and Future Work 225

8 Bibliography 228

List of Figures

2.1 A comparison between the standard squared residual and SISR residual [1] 17
2.2 Triangulation [2] 23
2.3 Geometric locus of a point seeing two known landmarks at a given angle, is a circle. $\widehat{AOB} = 2\widehat{ADB}$ [2] 23
2.4 Radio interferometric ranging technique [3] 26
2.5 APIT overview [4] 30
2.6 Approximate Point-In-Triangulation Test [4] 30
2.7 Dimension Reduction Illustration [5]. For a 3D manifold, after sampling and using dimension reduction technology, it is transferred to a set of 2D data, which preserves the neighborhood geometric structure 32
2.8 Localizing WSN by using dimension reduction technology [6]. The original pairwise range measurements are deduced by DV propagation (the red path). The estimated coordinates best fit the original measurement (the blue line). 33
2.9 Different constraints [7]. (a) radial constraint; (b) angular constraint; (c) quadrant constraint; (d) trapezoid constraint 34
2.10 Non-uniform WSN Localization [8] 36
2.11 REP localization illustration [8] 36
2.12 Lighthouse system illustration [9] 37
2.13 Lighthouse system 3 beams [9] 38
2.14 39
2.15 40
2.16 MSP extension [10] 41
2.17 SLAM illustration. A mobile robot moves through the environment, and measures the relative position to each feature. By using the observation measurements and its odometer, it estimates the location of each feature as well as its own position. [11] 43
2.18 Information matrix with tradeoff the number of the states and the density of the matrix [11] 51
2.19 Covariance Integration Illustration. The two big ellipses are the covariance of two random variables, the center ellipse is the optimal fusion, and the dashed ellipses at the intersection areas of the two big ellipses are the results of covariance integration by different choices [12] 52

3.1 Measurement Graph. The green Node 0 is the global origin, red Nodes 1 and 4 are anchor nodes, and black Nodes 2 and 3 are the nodes to be localized. The blue edges e2 and e4 represent the range measurements. The other black edges represent relative position measurements or GPS measurements (e6 and e7) 75
3.2 The Analogy Between a measurement graph and the generalized electrical network. The red nodes are anchor nodes. 78
3.3 90
3.4 Node 1's identity current graph. Nodes 4 and 5 are anchors. 92
3.5 Inova and OSE-COV simulation configuration 102
3.6 104
3.7 The error standard deviation comparison between Inova and the average-based method for 50 runs 105
3.8 107
3.9 108
3.10 109
3.11 110
3.12 111

4.1 Anchor Node Selection 116
4.2 120
4.3 Anchor Selection Strategies Comparison 121

5.1 The root mean square errors (RMSE) of DR and CL with different number of agents. 127
5.2 128
5.3 Extended Measurement graph. Each vertex represents a space-time state. Vertex 0 is the global origin. All solid lines are measurements, where the arrowed solid lines are strong measurements (GPS or relative position) and the non-arrowed solid lines are weak measurements (range or relative angle measurements). The dotted lines represent the covariance information, where the dotted line connecting a state and the origin represents self covariance $Cov(\mathbf{x}_{i,k}, \mathbf{x}_{i,k})$ (the diagonal elements of the covariance matrix), and the dotted line connecting states represents covariance of these two states $Cov(\mathbf{x}_{i,k}, \mathbf{x}_{j,l})$ (the off-diagonal elements of the covariance matrix). The vertexes connected to the global origin with dotted lines are the base nodes, which represent the base belief, i.e. $bel(\mathbf{x}_B) = (\hat{\mathbf{x}}_B, Cov(\hat{\mathbf{x}}_B, \hat{\mathbf{x}}_B))$. Here $\mathbf{x}_B = [\mathbf{x}_{1.1}^T, \mathbf{x}_{2.1}^T, \mathbf{x}_{3.2}^T]^T$ 130
5.4 136
5.5 RMSE for Agent 1 142
5.6 Localization error for Agent 1 142
5.7 The number of agents vs. Q 142
5.8 Average error over time 142
5.9 Localization error for a typical taxi 142
5.10 Localization RMSE for a typical taxi 142
5.11 EDCL overhead for a typical taxi 142
5.12 Average error vs. network density 142
5.13 Average error of non-GPS taxis 142

6.1 Indoor pedestrian localization system diagram. 149
6.2 151
6.3 Wifi channel frequencies vs. 802.15.4 channel frequencies. 152
6.4 CC2420 transmission power configuration [13]. 153
6.5 154
6.6 154
6.7 RSSI distribution 155
6.8 RSSI of body shadowing vs no body shadowing. 155
6.9 Frequency Hopping TDMA Design. There are two time phases: the TDMA time phase and the control phase. In the control phase, the frequency is fixed. The purpose of the control phase is to coordinate the channels and sending time for each participant in the following TDMA phase. In the TDMA phase, it has multiple TDMA periods. Each period has $N \times C$ sending time slot, where N is the number of participants, and C is the number of channels. The different filling colors of time slots represent the different participants (the corresponding participant should send in its own time slot). The different frame colors for a group of time slots represents the different channels. . . . 164
6.10 TelosB transmission and channel hopping delay 166
6.11 The RSS to distance estimation mobile case setting 170
6.12 171
6.13 Linear regression fitting for multi-channel RSS values versus distances . . . 172
6.14 RSS series of distance 2m with frequency hopping TDMA 173
6.15 MLE estimation accuracy with different distances under different channels 174
6.16 Wifi signals in Rice Hall 174
6.17 175
6.18 176
6.19 177
6.20 177
6.21 RSS to distance estimation error at long distances 178
6.22 Valid estimation number of RSS to distance mapping at long distances . . . 178
6.23 180
6.24 An example of log-log plot of Allan Deviation analysis results [14]. 186
6.25 187
6.26 187
6.27 189
6.28 ZUPT threshold setting 191
6.29 Foot-mounted inertial navigation module set up 191
6.30 192
6.31 193
6.32 194
6.33 The indoor pedestrian localization system setup 195
6.34 The GUI of the data collector 195
6.35 Synchronize Agent 2's IMU local time to Agent 1's IMU clock 197
6.36 The RSS data and the IMU data after pressing the key of number 1. 198

6.37 Two lines of successive RSS records from sender Id 1. 199
6.38 200
6.39 The trace of the covariance matrix of ZUPT estimation may decrease in some time period 202
6.40 205
6.41 206
6.42 207
6.43 209
6.44 210
6.45 210
6.46 Pedestrian 2's trajectory estimation in Route "together". The blue curve is the inertial-only estimation. The green curve is the estimation of EDCL64. The green square marker is the ground truth of the waypoints. 211
6.47 Maximum memory usage in Route "together" 212
6.48 213
6.49 214
6.50 215
6.51 216
6.52 217
6.53 217

List of Tables

3.1 74
3.2 95

6.1 188
6.2 188
6.3 203
6.4 203
6.5 204

Chapter 1

Introduction

1.1 Motivation

Recent improvements in micro-electronics, wireless communications, and low-cost sensor technologies have made wireless sensor networks (WSN) widely applicable to people's daily life. A wireless sensor network consists of a number of tiny sensor nodes which are capable of sensing, data processing and short range wireless communication. Sensor nodes are deployed either in fixed places or on mobile agents. By sensing, processing and communicating, this network of sensors collaborate to fulfill a specific task. Because of the low cost, small size, processing and wireless communication properties, WSNs have already been used in many applications, including environmental monitoring [15–19], health care [20–24], home automation [25–29], emergency support [30–34], vehicle networks [35–37] and military tasks [38–41].

Regardless of the different purposes of WSN applications, node localization is always a critical issue for these applications. Node localization means knowing the position of each sensor node in the network. Its importance is three-fold: first, in many applications, sensor data are only meaningful when they are bound with the position information. For example, in environment monitoring applications [15–17], sensors are used to create a "macroscope" of the monitored environment. If the light, temperature and pressure sensor data are not combined with position information, then we have no idea of what the environment looks like. In emergency support applications [30,32,33], first responder biometrics are monitored in order to ensure safety for the first responders. However, if the first responders' positions are not known, it is impossible to search for and guide them when they are in danger. In military applications [38, 40], the position information of the enemies are even more important, since we want to not only detect the appearance of the enemies, but also know where they are. Second, a family of WSN routing protocols are based on geographical information [42–45]. The idea of these protocols is to use location information to reduce the routes search space so that the routing overhead (computation and communication cost) can be decreased. Third, the localized sensor nodes can work as a location reference to provide position information to other agents which are close to them. This method is widely used in pervasive computing [46, 47], home automation [48, 49] and indoor hybrid tracking [50] systems.

Because of aforementioned reasons, the localization problem had received a significant amount of research for decades. Many localization solutions are developed in the WSN research field. They are divided into two categories: the range-based solutions [2, 51–57] and the range-free solutions [4, 9, 58–66]. Range-based solutions usually use special devices to measure the distances or bearing among nodes, and then apply a triangulation or multilateration algorithm to estimate the location. They usually can achieve a relatively accurate result, but the cost is also high. Range-free localization solutions do not directly measure the distances or angles among nodes, but make use of proximity information or position relevant events to infer node positions. Although its cost is lower than range-based solutions, the accuracy is also relative low especially when there are few redundant nodes in the network. Despite which method is used, the localization solutions from the WSN field suffer from the following drawbacks:

- Excluding heterogeneous information: many solutions only use one type of positioning measurements without taking advantage of heterogeneous positioning sources, which could otherwise dramatically increase the accuracy. For example, many acoustic ranging based Time of Arrival (TOA) localization system usually exclusively use one type of measurement (the relative distance). It is known [67] that as the distance increases, the accuracy of the acoustic ranging decreases. So if we can roughly detect that nodes are far away from each other, and then combine other positioning sources such as routing hops and received signal strength indicator (RSSI) values, the localization accuracy could be further improved.

- Not suitable for mobile networks: many localization solutions in WSN are designed only for static networks due to its batch processing property, i.e. one node position change causes the whole network to be recalculated [5, 68, 69].

- Requiring pre-set infrastructures: many solutions require infrastructures for localization, including huge devices [60,61], pre-deployed references [70,71], or map information [72,73]. However, these assumptions are unrealistic for some scenarios, e.g. fire fighting.

- Non-deterministic uncertainty: most of the solutions claim their accuracy in terms of average localization error. However, for a particular node, they can not guarantee its position estimation uncertainty, and this can be critical in some applications.

Another category of localization solutions is from robotics field. A technology called simultaneous localization and mapping (SLAM) [11, 74] has received a significant amount of research. SLAM solves the problem of localizing a mobile robot and simultaneously building a consistent map of the environment when the robot has no knowledge about the environment. In SLAM, the robot is usually equipped with some ranging sensors (e.g., an acoustic range finder, a laser scanner or a camera), measuring its range to the landmarks while it is moving. By obtaining these ranging measurements, the robot uses iterative Bayesian filters (e.g., the extended Kalman filter (EKF) or a particle filter) to update the location estimates of itself and the landmarks over time. After the notable successes of the research on SLAM, another interesting localization problem, called collaborative localization

(CL) [75–78], gained much attention from scientists. CL is to solve the problem of localizing multiple robots in an unknown environment merely through their collaboration. When a robot localizes itself merely based on the dead reckoning method, the uncertainty of its position estimation grows without any bound. However, when multiple robots correlate their position estimates through the inter-robot ranging measurements, the growth rate of the estimation uncertainty is dramatically reduced. Although great successes have been achieved on SLAM and CL, when applied to various applications, they still suffer from the following problems:

- SLAM: first, its iterative dynamic model is suitable for the mobile agent centric localization, but not the stationary WSN localization. Second, its high computational complexity, which is caused by the high dimension state space involving a large number of landmarks, prevents it from being applied to resource constrained devices. Third, it does not take advantage of the collaboration between agents, which could further improve the location estimation.

- CL: currently, there is no satisfactory decentralized CL solution. The existing decentralized CL solutions suffer from either the inefficiency in terms of memory usage and computational complexity (the optimal filters), or the over confidence problem (the approximate filters), where agents over-claim the accuracy of their location estimation.

Under the Moore's law [79], the size of the ranging hardware are getting smaller, and its cost are getting lower. Therefore, this thesis targets to the range-based localization solution, since it is more accurate than the range-free solutions, and does not have high requirement on node density. To overcome the aforementioned drawbacks of existing localization systems, the goals of our solution are listed as below:

- Abstract representation of range: to maximize the utilization of heterogenous ranging information, the range information in a localization system should be represented in a well formed abstraction. Under such an abstraction, the theoretical solution should not depend on any particular ranging technique, and hybrid localization systems would be possible.

- Suitable for both stationary and mobile sensor networks: the solution should be easily and efficiently applied to both stationary and mobile sensor networks.

- Infrastructure-free: the solution should be totally self-contained, i.e. not rely on any pre-set infrastructure, such as satellites, cellular towers, maps, reference nodes or environment profiling. But if there is infrastructure available, the solution should also be able to take advantage of such information.

- Quantitative uncertainty: the solution should provide not only position estimates for each node, but also the quantitative uncertainty of the estimates. Since the quantitative uncertainty provides great valuable information, e.g., guiding the anchor selection process, searching for a particular node, and evaluating the performance of the localization system.

- Centralized and decentralized support: centralized architecture can achieve global optimal estimation, while the decentralized architecture has the benefits of scaling well and not constrained by a third party center. Therefore, the thesis should provide solutions of both centralized and decentralized versions.

- Efficiency: the solution should be efficient in terms of memory usage, computational complexity and communication overhead, since the resources of the sensor nodes are typically very limited.

- Scalability: the solution should have good scalability, since a wireless sensor networks could contain thousands of nodes, and millions of range measurements could be generated over time.

1.2 Challenges

Due to the constrained resources of sensor nodes and the intrinsic complexity of the localization problem, the challenges mainly include the following aspects.

- *Formalization*: since we are considering heterogeneous ranging, the formal model should cover different types of measurements. However, the position related measurement has linear and non-linear form, and unary (e.g., the Global Positioning System (GPS)) and binary relationships (e.g., relative distance). How to unify these range types and the operations into one form?

- *Quantitative uncertainty representation*: quantitatively representing the uncertainty of ranging measurements and position estimates is important, because it not only leads to more accurate estimation, but also is very useful in practice, such as inferring the confidence region of the position estimates and guiding the anchor selection process. However, it remains a question that how to quantitatively represent the uncertainty and how to utilize it?

- *Efficiency*: due to the limited resources of sensor nodes, we put a severe requirement on efficiency in terms of memory usage, computational complexity and communication overhead. However, it is a challenging work: first, in a large scale network, the state space itself is in high dimension, e.g. if there are n nodes in the network and 3D coordinate is considered, the dimension of the state space is $3n$; second, heterogeneous devices may keep generating a large number of ranging measurements over time. To further refine the results, nodes' position estimates need to be updated frequently by using these measurements; third, nodes may quit and join the network at any time, and the network may be mobile. This means that it is necessary to re-estimate node positions frequently.

- *Decentralized solution*: when nodes collaborate with each other, their position estimates are correlated. This means, when one node's position estimate is updated, all the other correlated nodes' position estimates should also be updated. In a decentralized version of the localization problem, to have a consistent estimation, each node needs to preserve all the correlation information to other nodes, however, the limited resources of the sensor nodes do not allow this. Therefore, the practical challenge of the decentralized collaborative localization is how to balance the trade-off between the consistency and the efficiency.

- *Implementation challenges*: besides the theories, this thesis also includes the real system implementation. However, the real world is not perfect. The challenges are how to deal with the large noises of sensors, how to do the time synchronization, how to compose different modules into one system, and how to evaluate it.

1.3 Contributions

The work in this thesis has contributions at both the theoretical and practical level. For the theory part, three algorithms are created: the incremental node-voltage analysis (INOVA) algorithm is used to localize a stationary WSN in a centralized way. The overlapping subgraph estimator of covariance version (OSE-COV) algorithm, together with the original overlapping subgraph estimator (OSE) algorithm [69], is used to localize a stationary WSN in a decentralized way. The elastic decentralized collaborative localization (EDCL) algorithm is used to localize a mobile WSN in a decentralized way. All of the three algorithms provide not only the position estimates, but also the quantitative uncertainty of the estimates, and furthermore are efficient in terms of computational complexity, memory usage and communication overhead. Additionally, we also deduce how to use the quantitative uncertainty for confidence region inference and guiding the anchor selection process. For the system part, to demonstrate the effectiveness of the EDCL algorithm, we build a hybrid collaborative localization system for indoor pedestrians. This system achieves more accurate position estimation than typical independent inertial navigation systems (INS) without any pre-set infrastructure, such as reference nodes, maps, GPS, cellular towers and radio stations. The brief description and more specific contributions of each part are described in the following subsections.

1.3.1 INOVA and OSE-COV

Previous work [69] solves the stationary wireless sensor networks localization problem by using the Best Linear Unbiased Estimator (BLUE). It assumes that the ranging measurement has a linear representation, i.e. knowing both the relative angle and the relative distance between a node pair. Under this assumption, the localization problem can be formulated into a linear system, and thus can be solved by BLUE. Although the solution is elegant, it has two drawbacks: first it has $O(mn^2)$ computational complexity to form the linear system, and $O(n^3)$ to solve the linear system, where n is the total node number and m is the total measurement number. However, the computational complexity depending on the measurement number m makes this algorithm not scale well, since m would be a very large number as the node number increases (the possible different edge number of a graph is of the order of $O(n^2)$, and the same edge may be measured multiple times in practice). Second, this method follows a one-time calculation scheme, which means all the estimates must be recalculated when the measurement topology changes, e.g. nodes quit, join or move, or new measurements are generated.

Additionally, it has been shown [80] that the above relative measurement topology can be analogous to a generalized electrical network, where the node set is the same, the relative measurements correspond to the edges, and the covariance matrix of the relative measurements correspond to the generalized resistances. Inspired by this analogy, we derive that localizing each node in the network is equivalent to determining all the generalized currents on the corresponding branches and all the generalized voltage potential of the corresponding nodes. By borrowing the node-voltage analysis technique from electrical network theory, we efficiently solve the localization problem. The essential reason for the efficiency of INOVA is that when the network changes, only a constant number of elements change. Therefore, if the topology of the network is recorded by some mean, it only needs $O(1)$ time complexity to update when the change happens. Additionally, since the position estimation may be not needed for each time step, by delaying the estimation, the time complexity of INOVA can be further reduced.

The contributions of INOVA are two-fold: first, it provides a deeper understanding of the analogy between the relative measurement graph (RMG) and the electrical network. Second, it dramatically reduces the computational complexity of localization, and thus makes it possible to localize a large scale WSN with high dynamics.

In the work [69], another algorithm, called OSE, is created to iteratively estimate each node's position in the decentralized way. However, this method only calculates the position estimates, but not the covariance matrices. By exploiting the analogy between the RMG and the electrical network, we develop a method–OSE-COV, which can estimate the covariance matrices in the decentralized way. The nice feature of OSE-COV is that it fully fits into the same communication scheme of OSE. Therefore, by combining OSE and OSE-COV together, nodes' position estimation and the corresponding covariance matrices can be simultaneously obtained in the decentralized way.

1.3.2 Confidence Region Inference and Anchor Selection Strategy

As aforementioned, for the localization problem, not only the position estimate, but also the corresponding confidence region is important. The confidence region inference solves the problem of associating a node's position estimate with an area A and a trust level α, where the real position of the node falls into A with a probability of α. By using the central limit theorem of Lindeberg's condition [81], we prove that the estimates of INOVA follow the multivariate normal distribution regardless of the distribution of the measurement error. Hence, provided with the covariance matrix, we can easily deduce the confidence region based on the multivariate normal distribution property.

Anchor nodes are defined as the nodes which have accurate position information under a global coordinate frame. The anchor selection is to solve the problem of selecting an proper anchor (or a number of ahcnors) so that the localization accuracy of the whole network could be improved. Given the associated covariance matrix of each node, the most intuitive way is to select the node which has the maximum uncertainty. However, this variance-based strategy may be not the best, since it does not consider the correlation between nodes. In this thesis, we derive an optimal anchor selection strategy. The simulation result shows that the optimal strategy is the upper bound of the other strategies, and reduces the total mean square error of the whole network by four times as much as the variation-based algorithm in the best case.

1.3.3 EDCL

The INOVA algorithm works nicely for both stationary and mobile WSNs, but it follows a centralized architecture. OSE plus OSE-COV are distributed algorithms, but they are only suitable for stationary WSNs. A localization algorithm is needed for the mobile decentralized networks. Therefore, we develop the elastic decentralized collaborative localization (EDCL) algorithm for this most challenging case.

The idea of the decentralized mobile network localization is straightforward: each mobile agent only maintains the states of itself. When they meet with each other, they refine their estimates by sharing their position information. However, the biggest challenge is that as agents collaborate with each other, their position estimates are correlated. When one agent's state gets updated, it needs to bring this update to its correlated agents in future when they meet. However, since the embedded devices have a limited resources, we can not preserve all these correlations for all the time. Therefore, the key issue is: when to forget these correlations and how to update the other agent when they meet?

This problem is a decentralized CL problem, and there are two types of existing decentralized CL algorithms: the consistent (optimal) algorithms and the approximate algorithms. However, both have their drawbacks. The consistent algorithm preserves all correlation information, and thus it is unrealistic since the resource consumptions (in terms of memory, computation and communication) grow without any bound. The approximate algorithm forgets all correlations, which causes the over-confident problem, and thus impacts the accuracy of the estimation and the confidence region inference.

To solve this dilemma, we propose the EDCL algorithm. The underlying idea of EDCL is to introduce a parameter Q, which represents the number of the measurements to be preserved in the memory, so that the trade-off between the consistency and the efficiency becomes adjustable. By using EDCL, devices with different capabilities can choose different Q, and work collaboratively.

The contributions of this work are: first, it is the first decentralized CL algorithm which bridges the gap between the optimal filters and the simplest approximate filters. Second, to implement EDCL in an efficient way, an information relation concept and a chain-based data structure are introduced. By using these data structures, EDCL is able to process out-of-order measurements, and reduce the communication overhead to a very low level. Third, EDCL is evaluated through a large scale taxi localization problem, which is based on the real data set involving 200 taxis in Beijing within one day in 2008. The result shows that the performance of EDCL is accurate, and the overhead is low.

1.3.4 Infrastructure Free Indoor Pedestrian Localization System

To demonstrate the feasibility of the EDCL algorithm, an indoor pedestrian localization system is implemented. This system is fully decentralized and self-contained, i.e. not dependant on any infrastructure or map information. The system consists of three modules: an inter-agent ranging module, which is implemented by a RSSI based range estimator, a dead reckoning module, which is implemented by a foot-mounted inertial navigation system, and a decentralized fusion filter which uses the EDCL algorithm.

Since RSS is notorious noisy, we use a moving window median filter, place the mote on top of a person's head, propose a frequency hopping TDMA protocol, and develop the corrected truncated maximum likelihood distance estimator to overcome the body shadowing effect, multipath effect and selective frequency effect. For the foot-mounted inertial navigation module, the ZUPT algorithm is adopted. The Allen Variance technique and the simulated annealing algorithm are used to characterize the inertial sensors. To integrate all these modules of multiple pedestrians, a time synchronization mechanism is implemented by carefully designing the packet format. Furthermore, to reduce the number of the measurements without losing the information, most of the inertial displacement measurements are compounded. The experimental results show that the localization accuracy of EDCL is close to the optimal filter–INOVA, which reduce the error as much as 50.41%, and the resource consumption is low, which has the maximum memory usage of 760 bytes and the average communication cost 33 bytes per message for $Q = 64$.

The contributions of this localization system are as follows: as far as we know, it is the first fully decentralized, self contained, sub-optimal collaborative pedestrian localization system. By using low cost devices, it achieves a relatively high localization accuracy (49.44% better than the inertial-only method).

1.4 Organization

The rest of the thesis is organized as follows. Chapter 2 thoroughly examines the state of the art on the localization problem. Chapter 3 explains the INOVA and OSE-COV algorithms. Their performance is evaluated through extensive simulations. Chapter 4 studies the problems of confidence region inference and anchor nodes selection. Simulations are conducted to evaluate the effectiveness of our solutions. Chapter 5 introduces the EDCL algorithm. A semi-simulation based evaluation is provided to show the accuracy and the efficiency of EDCL. Chapter 6 describes the real infrastructure free indoor pedestrian collaborative localization system. Real experiments are conducted for the evaluation. Finally, Chapter 7 provides the conclusions, limitation discussions, and the future works.

Chapter 2

State of the Art

The localization problem has been studied for many years in various fields. The research directions include estimation algorithms, ranging techniques, and localization systems. This chapter provides a survey on the state of the art which is most related to this thesis. Section 2.1 presents the work in the wireless sensor network field. The approaches in WSN are divided into two categories: range based solutions and range free solutions, which depend on whether the solution requires ranging measurements (e.g., the relative distance or the relative angle between nodes) for localization or not. Section 2.2 focuses on two localization problems in the robotics field: simultaneous localization and mapping (SLAM) and the collaborative localization (CL) problem. The former is to solve the problem of localizing a mobile agent and simultaneously building a map of the environment when the robot is put into an unknown environment. The latter aims to localize multiple mobile agents only through their collaboration. Section 2.3 presents a survey on real systems which use hybrid techniques to localize mobile agents.

2.1 Localization with Wireless Sensor Networks

2.1.1 Range Based Localization

Range based localization solutions require nodes to have the ranging capabilities. A range measurement is a relative position measurement between two nodes, which may refer to the relative distance, the relative angle, or the relative velocity. By collecting theses ranging measurements from nodes, various algorithms, such as triangulation, bilateration, multilateration and convex optimization, can be applied to compute each node's position coordinate. Based on the fundamentally different ranging technologies, we classify the range based localization solutions into the following four categories: (i) received signal strength (RSS) based solutions; (ii) time of flight (TOF) based solutions; (iii) angle of arrival (AOA) based solutions; and (iv) radio interferometry (RI) based solutions. Note that this classification does not prevent systems from using hybrid technologies to enhance the localization accuracy.

2.1.1.1 Received Signal Strength Ranging based Localization

The received signal strength ranging technology makes use of the fact that the RSS is a function of the propagation distance to estimate the range between the signal transmitter and the receiver. This relationship holds for the signals such as radio (electromagnetic wave), sound or seismic waves. While the sound signal is usually used in another more reliable form–TOF [67], and the seismic signal is used for localizing the earthquake center or tracking vehicles [82], the radio frequency (RF) signal [1,82–89] is widely used and studied in the WSN field, since it comes almost for free.

H. T. Friis first derives the following formula to calculate the received signal power in free space at distance d from the transmitter [90].

$$P_r(d) = \frac{P_t G_t G_r \lambda^2}{(4\pi)^2 d^2 L} \tag{2.1}$$

where P_t is the transmitted signal power. G_t and G_r are the antenna gains of the transmitter and the receiver, respectively. $L(L \geq 1)$ is the system loss, and λ is the wavelength.

Equation 2.1 only considers the ideal case. However, in reality the received power at certain distance is a random variable due to the multipath propagation effects, which are also known as fading effects. Therefore, a more sophisticated model is presented in the book [91].

$$\tilde{P}_r(d) = \tilde{P}_r(d_0) - 10\beta \log_{10} \frac{d}{d_0} + n_{dB} \tag{2.2}$$

where $\tilde{P}_r(d)$ and $\tilde{P}_r(d_0)$ are the received signal powers in dB at the distances of d and d_0, respectively. β is the path loss exponent, and is usually empirically determined by field measurement. $\beta = 2$ for free space propagation, and the large β represents the fast attenuation of the signal power when it is propagating. n_{dB} is a Gaussian random variable with zero mean and the standard deviation σ_{dB}. σ_{dB} is called the shadowing deviation, and is independent with the distance.

Although Equation 2.2 models the path loss and the shadowing effects, in practice, the RSS is often unpredictable due to many factors, such as unknown path loss, multipath propagation effects, hardware discrepancy, antenna issues and radio noises [66,83–85]. Therefore, much work has been done to characterize various properties of RF signal propagation [83–87], and use different noise sifting algorithms [1,86,92] or hybrid range estimation technologies [89] to improve the RSS based localization accuracy.

Savvides, et al. [83] tested the RSS based ranging under different environments with different transmission power settings. They found that it was hard to get a consistent model in indoor environments due to the multiple path fading effects. They also observed that the node heights from the ground impact the RSS significantly. Additionally, they also pointed out that the low power devices exhibit significant variation in actual transmit power for the same transmit power level setting, or in the RSSI measured for the same actual received signal strength. Whitehouse, et al. characterized lower power radios in near-ideal conditions, and found that a standard deviation in RSS readings translated to about 2m standard error at the maximum range of about 20m after calibration [84]. They also verified the correctness of the *Noisy Disk* radio model, which had been widely used in simulations, with the real experimental data, and found that when the radio connection degree was high, the noise could be modeled as a Gaussian distribution, but when the degree was low, it could not. They also analyzed how these errors could impact multihop localization solutions. Miluzzo, et al. characterized 802.15.4 radios which were worn by mobile persons [87]. They found that the body shadowing factor had significant effect on the RSS, and different placements on the body also had much impact. Ziguo Zhong, et. al. observed in their outdoor experiments that although identical RSS values may correspond to different distance, for a same node, the RSS values mostly decreased monotonically with the increasing distance [66].

Although RSS is notorious for poor range estimation, Kamin Whitehouse, et. al. demonstrated that RSS can be used to localize multi-hop sensor networks [85]. They achieved 4.1m error in a 49 node network deployed in a half-football field sized area by using the DV-distance algorithm [2]. In their experiments, they found that careful calibration, increasing network density and lowering the transmission power were helpful to improve the localization accuracy.

Patwari, et al. derived statistic models for RSS measurements and connectivity [86]. Based on these models, they gave a corrected maximum likelihood estimator for RSS to distance mapping, and pointed out that the standard deviation of the estimate was proportional to the actual range. Finally, they presented three *manifold learning* algorithms, including Isomap [93], dwMDS [94] and LEAN [95], and compared their computational cost and localization accuracy by using RSS or connectivity measurements.

Since RSS based ranging measurements are often subject to large errors, it is necessary to find a good algorithm which can discern the quality of measurements so that accurate measurements are weighted more and the inaccurate measurements are weighted less in the localization process. Therefore, Kung, et al. proposed an error-tolerant localization method,

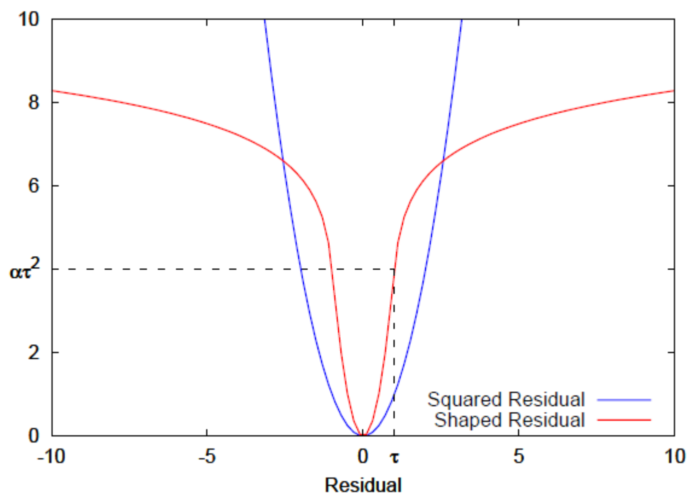


Figure 2.1: A comparison between the standard squared residual and SISR residual [1]

called snap-inducing shaped residuals (SISR), to identify automatically "bad nodes" and "bad links", and so that they obtained different weights in the localization process [1]. The key idea of SISR was to use a *wing-shapped* residual function to emphasize the weights of good measurements and constrain the weights of bad measurements. A comparison between the standard squared residual and SISR residual is shown in Figure 2.1. The blue curve in the figure is the standard squared residual, and the red one is the SISR residual. We can see that in the standard squared residual, the cost increases quadratically as the residual increases; while in the SISR residual, the cost increases rapidly when the residual is small ($[-\tau, \tau]$), but very slowly when the residual is big ($(-\inf, -\tau) \cup (\tau, \inf)$). By using such residual functions, SISR demonstrated its better performance than the algorithms Lorentzian [96] and MDS-MAP [97] developed based on outdoor RSS ranging measurements.

In the real world, only using one technology for localization is often not enough. Zhao, et al. proposed a *Combined and Differentiated Localization (CDL)* approach, which was a hybrid system exploiting the strength of range-free approaches and range-based approaches using RSS measurements [89]. It had three components: virtual-hop localization, which was a range free localization approach, and similar with but more accurate than DV-hop algorithms [2,58]; local filtration, which was to identify good nodes and bad nodes through neighborhood hop-count matching and neighborhood sequence matching; and ranging-quality aware calibration, which was similar as SISR, but took the node quality and ranging measurement quality into account for localization. By using these three components, CDL achieved an average localization error of 2.9m in a 300 node network deployed in a forest.

2.1.1.2 Time of Flight based Localization

One large category of ranging technology is based on time of flight (TOF) measurement of signals. The principle is straightforward: given the propagation speed of a signal, the distance between a sender and a receiver can be calculated based on the TOF measurement of this signal. However, there are two questions about this technology: which signal could be used? And how to precisely measure the TOF? For the former question, acoustic signals [51–53, 67, 98–103] (including sound and ultrasound) and RF signals [104–112] are usually used in WSN. For the latter question, there are two means to precisely measure the TOF: directly measure the time of arrival (TOA) between time synchronized sender and receiver [51, 52, 67, 98, 99, 105, 110]; or measure the time difference of arrival (TDOA) to cancel out the local time discrepancy between nodes [53, 100–103, 107, 109, 111].

The main challenges of TOF based localization systems include: i) the time synchronization issue; ii) the non-light of sight (NLOS) problem; iii) and the multipath effect in the indoor environment. Since acoustic signals and RF signals differ significantly, in the following paragraphs, we separately discuss these two technologies.

Time of Flight of Acoustic Signals :

The Bat system [98] achieves accurate indoor localization based on the TOF of ultrasound signals. The system has three components: the Bat tags, the ultrasound receiver units, and the base stations. A Bat is attached to a person or an object to be localized. It consists of a radio transceiver, controlling logic and an ultrasonic transducer, and can emit ultrasound. The receivers are placed at known positions on the ceiling of the rooms. They are used to receive the ultrasound emitted by Bats, and record the time of arrival. The base station is used for time synchronization and position calculation. The whole system works as follows: a base station periodically notifies a particular Bat to emit a short pulse of ultrasound, and simultaneously resets the receivers via the wired network. Receivers monitor the incoming ultrasound and record the time of arrival from the Bat. Then the TOF measurements are converted to distances between the Bat and the corresponding receivers. If there are three or more non-colinear receivers found, then the 3D position of the Bat is determined by the multilateration algorithm. By attaching multiple Bats on an object or using the directional shadowing property of the object, the orientation of the object can also be determined. The Bat system achieves an average accuracy of 3cm. However, the drawback is that it needs carefully pre-installed infrastructures.

To protect the privacy of users, the Cricket [51] system reverses the design: tags are receivers and beacons fixed at the known places are transmitters. Cricket does not use a centralized party for synchronization, instead, it uses the RF signals for time reference. Each time a beacon transmits an ultrasound pulse, it also transmits a RF signal with its Id. The RF signal is long enough so that when the tag receives the ultrasound pulse, it is still in the precess of receiving the RF signal. By using this methodology, the tag knows when the ultrasound transmission begins and which beacon it is talking to. To overcome the possible incorrect association and increase the range estimation accuracy, carefully installation of beacons and statistics based filters are needed. The experiment results show that Cricket achieves a 4×4 feet location granularity.

While Bat and Cricket use RF for time synchronization, there are systems which use time difference cancelation to measure the TOF of signals, which is called the time difference of arrival (TDOA) method. TPS [53] and UPS [101] provide an interesting scheme where a node can be localized in 3D space as long as this node and four other anchor nodes are within one-hop communication range. In the system, time is divided into beacon intervals. In each beacon interval, a master anchor node A initiates the signal broadcast. After receiving the signal from A, Anchor B replies by including the time information Δt_b^i, which indicates the time duration between the time B receiving the signal from A and relying A. Similarly, when Anchor C receives B's broadcast, it replies with Δt_c^i, and Anchor D also replies with Δt_d^i after receiving C's broadcast. Using Δt_b^i, Δt_c^i, Δt_d^i, and the timestamp records when Sensor S hears from A, B, C and D, the 3D position of Sensor S can be calculated. Note that all the time information above is only based on each node's local clock; no global time synchronization is needed. Other systems like [102] and [103] have a similar idea, where nodes and beacons only broadcast their local timestamps, and the discrepancy of node clocks is canceled out through the process.

Another TDOA based system [100], called BeepBeep, proposes an interesting approach: by emitting two beeps, two nodes can estimate their ranges without any synchronization. The basic idea is that when nodes A and B beep in turns, each records the sound from the other as well as the sound from itself. Therefore, totally four local timestamps are generated. By solving a simple linear equation with these four local timestamps, the local time difference is canceled out, and each device can precisely obtain the range between them. In addition, to avoid the timing uncertainty caused by the embedded system, BeepBeep generates the timestamp by directly counting the sample positions in a sound track. The prototype system is implemented on commercial off-the-shelf (COTS) cell phones. The experiment results report that BeepBeep achieves an average accuracy of 1cm to 2cm with a standard deviation of less than 2cm in both indoor and outdoor environments.

Time of Flight of RF Signals: Time of Flight of RF Signals based ranging is extremely challenging in WSN. Besides the aforementioned three factors (time synchronization, NLOS and multipath effect), the extra challenge is mainly from the conflict between the high speed of RF and the constraint hardware of wireless sensor nodes.

The error bound of a TOF of RF based system is analyzed as follows. Assuming the speed of RF signals is 3×10^8m/s and TOF sampling error is within $\pm T_s/2$, where T_s is the sampling rate, it corresponds to the range error $\pm 150/F_s$, where $F_s = 1/T_s$ is the sampling frequency. To keep the range error within ± 1m, a bandwidth of 75MHz is needed, and thus the minimum sampling frequency is 150MHz [110].

Additionally, the Cramer-Rao Lower Bound on the TOF estimate variance is

$$\sigma_{TOF}^2 = \frac{1}{8\pi^2 \cdot SNR \cdot \sqrt{\alpha} \cdot BW^2 \cdot N} \tag{2.3}$$

where SNR is the average signal to noise ratio (SNR), α is the number of code copies averaged, BW is the spectral bandwidth, and N is the number of chips in the pseudorandom noise (PN) code. This formula tells us that to increase the ranging accuracy, we should increase the code length, code rate and the carrier bandwidth. However, in the indoor environment, the ranging accuracy is mainly constrained by multipath effects. Increasing the code length and code rate have very limited effect. But increasing the carrier bandwidth is very helpful to resolve the multipath effect, since the fine time resolution allows differentiation among delayed replicas [111, 112].

To estimate the TOF of RF signals, two techniques are mostly used: direct-sequence spread-spectrum (DSSS) and ultra-wideband (UWB). The basic idea of DSSS systems is: a PN code is transmitted to a receiver which already knows this code in advance. Then the receiver correlates its local PN code with the arriving signal. If a correlation peak above some threshold is found, the signal is considered to be detected and the position of the first peak is used to determine the TOF of the signal. Since DSSS needs to precisely determine the offset of the correlation peak, a high frequency clock is required. Additionally, DSSS requires the time synchronization between the transmitter and the receiver, hence a time drift correction or cancelation module needs to be implemented at the sender and the receiver. To mitigate the multipath effect, a frequency hopping approach is used, since different frequencies correspond to different path lengths in terms of the wavelengths and thus their multipath effects are different [105, 111].

Ultra-wideband is defined as a radio which has relative bandwidth larger than 20% of its central frequency or absolute bandwidth of more than 500MHz [108]. UWB technology is very accurate for short distance ranging in the indoor environment for two reasons. First, since it has a large frequency span, it dramatically increases the possibility for some of the frequency components to penetrate or go around objects, which is an important property especially for indoor NLOS ranging. Second, its fine time resolution allows multipath delayed signals to be differentiated through a matched filter [107]. The UWB system in the work [107] uses a generalized maximum-likelihood estimator to resolve the multipath effect, and a two-way timestamping method to solve the time synchronization issue. The reported indoor localization error is within 4 feet.

The system [110] implements the TOF of RF based ranging module on a field-programmable gate array (FPGA) with an 8-bit 100Msps ADC, using the RF within 2.4-2.5GHz band. It uses Orthogonal frequency-division multiplexing (OFDM) encoding method for simplicity, takes the strongest channel impulse response as the LOS signal to resolve the multipath effect, and requires pre-calibration for time synchronization. The reported ranging error is within -0.5m to 2m for the measured distance as far as 10m. The PinPoit system [109] does not assume a constant time discrepancy (the absolute and the relative time drifts) between devices. It uses two round trip timestamps to calibrate nodes at run time. The advantage of this system is that for a network of n nodes, it only needs totally $O(n)$ message exchanges to obtain the two-way range estimates of all node pairs. Another byproduct of PinPoint is that each node can determine the clock characteristics of the nodes in its neighborhood with an accuracy on the order of its clock tick. Using a FPGA with 300MHz clock and a 802.11 radio chip, PinPoint achieves an average accuracy of 4.18 feet in the indoor environment.

2.1.1.3 Angle of Arrival based Localization

Besides the relative distance measurements, the pairwise angle of arrival (AOA) information between nodes can also be used to localize the WSNs. Assuming that in a wireless sensor network each node is capable of measuring the bearing to its neighbor nodes, and some of them are anchors whose position is perfectly known, then the WSN can be localized in several ways as follows.

As Figure 2.2 shows, Nodes A, B and C are anchors, and Node D is to be localized. If Node D knows the angles $\widehat{BDA}$, $\widehat{ADC}$ and $\widehat{CDB}$, it can find its position using triangulation.

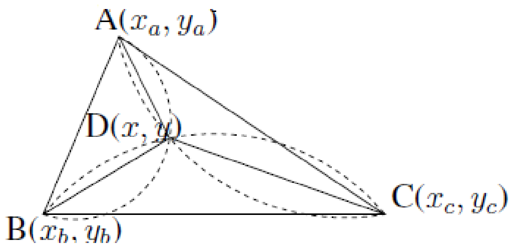


Figure 2.2: Triangulation [2]

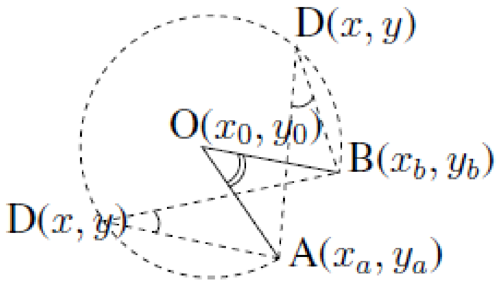


Figure 2.3: Geometric locus of a point seeing two known landmarks at a given angle, is a circle. $\widehat{AOB} = 2\widehat{ADB}$ [2]

This is done by finding the intersection of the three circles determined by the landmarks and the known angles. Actually, the triangulation problem can be transformed to the trilateration problem. Figure 2.3 shows two anchor nodes A and B, and an unknown node D. We can see that if $\widehat{ADB}$ is known, D must be somewhere of the circle determined by the positions of A and B and the angle $\widehat{AOB} = 2\widehat{ADB}$. Therefore, a pair of bearing measurements to two anchors forms one such circle. If there are m anchors in the network, there would be totally $\begin{pmatrix} m \\ 2 \end{pmatrix}$ circles, and thus D can be found at the position which has the minimum mean distance to all these circles. Another posibility is to form all triplets of obtained anchors and find the center of the circumscribed circle for each such triplet and the unknown point D. This leads to the solving of $\begin{pmatrix} m \\ 3 \end{pmatrix}$ trlateration problems [2].

If each node not only obtains the bearing to its neighbor nodes, but also has a compass, then it can obtain the absolute angle measurements to each anchor node with respect to the north, and the localization problem is dramatically simplified. Assuming the coordinate of the node to be localized is $[x, y]^T$, each anchor's known position is $[x_i, y_i]^T, i = 1, 2, \cdots, m$, and the node's angle measurement to each anchor is α_i, then we have an over-determined linear system [113]

$$\begin{bmatrix} 1 & -\tan\alpha_1 \\ 1 & -\tan\alpha_2 \\ \vdots & \vdots \\ 1 & -\tan\alpha_m \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x_1 & -y_1\tan\alpha_1 \\ x_2 & -y_2\tan\alpha_2 \\ \vdots & \vdots \\ x_m & -y_m\tan\alpha_m \end{bmatrix} \tag{2.4}$$

and it can be easily solved by the least square (LS) method.

Above are the possible algorithms for localizing nodes using AOA information, but there remain two challenging questions: i) how to obtain the AOA measurement? ii) How to mitigate the measurement noise effect?

To measure the bearing from a receiver to a sender, there are several possible ways. The Cricket Compass system from MIT [114] uses an array of ultrasound receivers (totally five) to measure the bearing of the receiver to the sender. By carefully placing the five ultrasound receivers on a board and exploring the trigonometric relationship between the placement distances and the angle, it can calculate the bearing angle of the incoming ultrasound signal. Since the distances between the receivers are very small (several centimeters), it is very difficult for the small sensor node to measure the difference of the distances from different receivers to the transmitter. Therefore, it uses phase difference to infer the physical distance. Elvira, et al. [115] use the same idea, but they only use 3 receivers but two transmitters in order to calculate the bearing information in 2D space. In addition, they directly measure the distance difference using TDOA, which needs particular hardware (FPGA). A directional antenna or an adaptive antenna can also be used to measure AOA. The idea is that each anchor node is equipped with a rotating directional antenna. When the anchor begins to rotate its antenna, it simultaneously emits a short omni-directional pulse. Nodes obtain the bearing to the anchor node by measuring the time difference between the reception of the omni-directional pulse and the signal from the rotating directional antenna [113]. Furthermore, angle information can be obtained by the radio interferometry technology [116]. The details of this approach are discussed in the next subsection.

In reality, measurements always come with noise. The ways the localization algorithms fuse the noisy measurements decide the overall accuracy of the system. The work [113] claims that the accuracy of an AOA measurement is proportional to the RSS from the directional antenna, and the nodes which are close to 90° or 270° are prone to have a large uncertainty. Therefore, it weights each measurement with $m_i = P_i/(1 + |\tan\alpha_i|)$, where m_i is the weight to the ith anchor measurement, P_i is the maximum power of the signal received from the ith anchor, and α_i is the angle measurement. Basu, et al. [117] treat the uncertainty of the distance and angle measurements as bounded regions, and transforms the localization problem to a constrained convex optimization problem. They prove that localizing with noisy distance and angle measurements is a NP problem, and the outputs of their method are the upper and lower bounds of the position estimation. Bishop, et al. [118] also model the AOA localization problem as a constrained convex optimization problem. Different from the traditional mean estimation based optimization method, which would break the geometric constraints between measurements, it still keeps all the geometric constraints during the estimation, and thus has a better accuracy than the traditional ones.

2.1.1.4 Radio interferometry based Localization

The radio interferometry based localization approach is different from the aforementioned ranging methods. This technique relies on a pair of nodes emitting radio waves simultaneously at slightly different frequencies. Although the carrier frequency of the transmitter is high (usually above 400MHz), the composite signal at the receiver has a very low frequency envelope (below 1KHz), which is precisely detectable by the resource constrained sensor nodes. The low envelope frequency includes the information related to the involved nodes' positions, and thus can be used for localization. The advantages of the radio interferometry based ranging are: i) the estimation accuracy is high, which can achieve sub-meter level error; ii) the sensing range is long, which can reach as four times far as the communication range; iii) it is resilient to multipath effect if only using the frequency difference information;

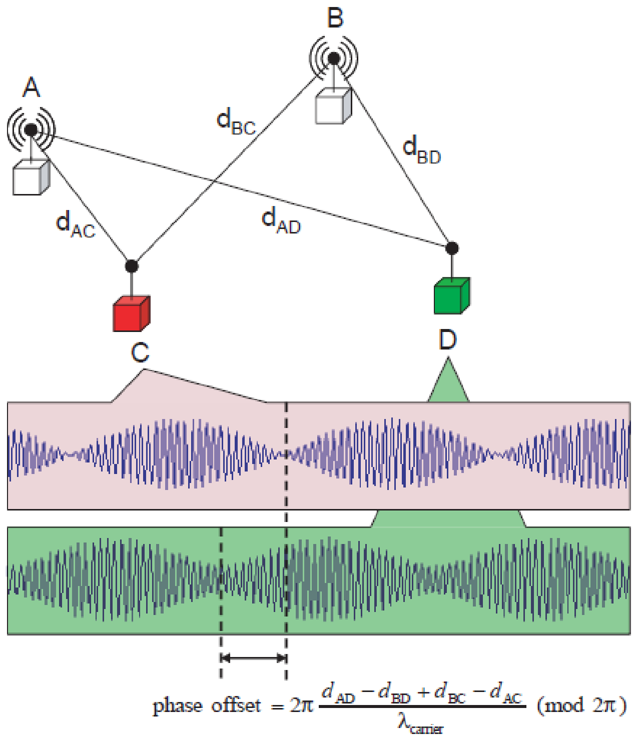

A
B
C
D
d_{BC}
d_{BD}
d_{AC}
d_{AD}
phase offset $= 2\pi \frac{d_{AD} - d_{BD} + d_{BC} - d_{AC}}{\lambda_{carrier}}$ (mod 2π)

Figure 2.4: Radio interferometric ranging technique [3]

iv) and it can obtain different types of range information, such as distance, angle and velocity. However, the shortcomings include i) requiring highly accurate time synchronization between nodes; ii) needing extensive system tuning and calibration; iii) relying on specific type of radios which support transmitting unmodulated sine waves, and tuning the carrier frequency at a fine resolution.

Maróti, et al. were the first to bring the radio interferometry technique into WSN localization [3]. The basic idea is as Figure 2.4 shows. Transmitters A and B simultaneously transmit unmodulated sine waves with frequencies f_A and f_B. Receivers C and D receive the interfered signals, which have the same envelope frequency $\delta = (f_A - f_B)/2$ but different phases. By passing through a low pass filter, the envelope frequency and the relative phase offset between the C and D can be obtained by the resource constrained sensor nodes. And the phase offset is related to the distances among the four involved nodes: $\vartheta_{ABCD}(f) = 2\pi\frac{d_{AD}-d_{BD}+d_{BC}-d_{AC}}{c/f}$ (mode 2π), where d_{XY} is the distance between Nodes X and Y,

$f = (f_A + f_B)/2$ and c is the speed of light. We define $d_{ABCD} = d_{AD} - d_{BD} + d_{BC} - d_{AC}$ as the q-range of Nodes A, B, C and D. Since the q-range is related to the four nodes' positions, with sufficient q-range measurements, the nodes' positions can be reconstructed. By evaluating with 16 nodes within 4×4 grids outdoors, the radio interferometry method results in an average error of 3 cm.

Lately, Kusy, et al. extend the above work by solving two practical problems [119]: i) the ambiguity of the q-range measurement caused by the particular choice of frequencies used in [3]; ii) and the multipath effect which distorts the phase of the interfered signal. By choosing a better frequency separation and developing an iterative localization algorithm, the improved method achieves 4 cm average accuracy for a quasi-random deployment of 16 nodes covering the area of two football fields.

The radio interferometry technique is also applied to tracking mobile nodes. The challenge of tracking a mobile node is that it requires a relative high refresh rate, and thus the in-field calibration and the q-range measurements are not allowed to take a long time. In the inTrack system [120], the mobile node works as a transmitter, and cooperates with the infrastructure nodes to obtain the q-range measurement as it is moving. To reduce the frequency calibration time, the receivers notify the transmitters to calibrate the frequency only when a relative large frequency drift is observed. Therefore, the tuning procedure and the q-range measurement are able to work in parallel. To address the q-range ambiguity and the multipath propagation problem, the ranging module produces a set of q-ranges rather than a single value, and lets the tracking algorithm resolve differences. Since each q-range corresponds to a hyperboloid in 3D and the true location of the mobile node is ideally the intersection point of all these hyperbolaes, the tracking algorithm searches the whole space to find the node's position–a region which has the most intersection points of these q-ranges.

Although inTrack works well for one mobile node, it has scalability problems. As the number of the mobile nodes increases, the computational cost of the searching algorithm grows quadratically. Furthermore, since the mobile nodes work as the transmitters, they have to access the channel exclusively, which increases the latency of the tracking. Therefore, mTrack [121] reverses the roles: the mobile nodes work as receivers. More importantly, mTrack proposes the utilization of the doppler shifts, which not only compensates for the changing q-range, but also provides a byproduct, the velocity of the mobile node.

Inspired by the mTrack system [121], Kusy, et al. proposes a system for tracking mobile nodes by only using RF Doppler shifts. The infrastructure and the interference method are similar as previous systems [120,121], but it only measures the RF Doppler shifts instead of the relative phase offset. It has been found that Extended Kalman Filter is good at tracking nodes with constant speed and steady direction, but fails if the node changes speed and direction abruptly. Therefore, they combine the Extended Kalman Filter and a constrained non-linear least square method, and result in a 50% accuracy improvement.

Since Doppler shifts only occur when the wave source has a relative movement to the observer, it cannot be used to localize the stationary nodes. However, Chang, et al. develop an indoor localization system which is able to localize the stationary nodes [116]. The idea is to let the infrastructure nodes spin and thus generate the Doppler shifts. By using the angulation method, it achieves a sub-meter accuracy.

2.1.2 Range Free Localization

In contrast with range-based localization solutions, the range-free localization methodology does not require nodes to have ranging capabilities. Instead, it uses proximity, connectivity, signature matching or event association to infer nodes' approximate positions. The most important advantage of the range-free localization methodology is cost efficiency, since it does not require special hardware for ranging. But the disadvantages are (i) it requires high density of anchors or target nodes; (ii) and the localization accuracy is usually lower than range-based solutions. In this section, we briefly introduce the range-free localization in three categories: (i) anchor proximity based solutions; (ii) network connectivity based solutions; (iii) signature matching based solutions; (iv) and event based solutions.

2.1.2.1 Anchor Proximity

The underlying idea of anchor proximity solutions is very simple, i.e. if Node A can sense (e.g. by radio, infrared, acoustic or other sensors) Node B's existence, Node A considers Node B is in its vicinity, i.e. $d_{AB} < R_A$, the distance between A and B is considered smaller than the sensing range of Node A. As many sensing information of this type is collected, nodes can be localized accurately.

Centroid [122] is an early representative work of the anchor proximity methodology. In Centroid, if a node hears a set of anchors with certain link quality, it considers itself near these anchors, and estimates its position as the gravity center of these anchors. That is the node's position $(\hat{x}, \hat{y})$ is calculated by the following formula:

$$(\hat{x}, \hat{y}) = \left(\frac{1}{n} \sum_{i=1}^{n} x_i, \frac{1}{n} \sum_{i=1}^{n} y_i \right) \tag{2.5}$$

where n is the number of anchors it hears with certain link quality, and (x_i, y_i) is these anchors' coordinates.

Lately, LANDMARC [123], WCL [124] and improved WCL [125] improves Centroid method by introducing weight on each anchor's coordinate. Instead of using constant weight 1 as Centroid does, these methods use different weights on different anchors' coordinates based on their distances or RSSI values: the closer the anchor is, the larger weight it has. The experiment results demonstrate that these methods outperform the Centroid method. Another more complicated method MSL* (and MSL) is proposed in work [126]. MSL* uses the proximity information of not only anchors but also normal nodes. MSL* uses the particle filter to estimate node's position distribution, which can localize both static and mobile networks.

Another interesting work [4], called APIT, uses area proximity for localization. The idea is that by judging whether a node is inside or outside of a triangle which is formed by three anchor nodes, we know the rough area the node resides. By testing all the combinations of three anchor formed triangles, we can finally reduce the potential residence area of that

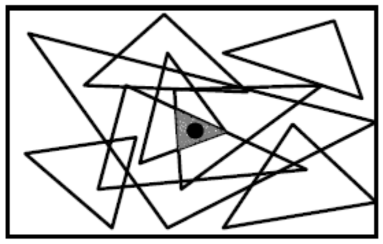


Figure 2.5: APIT overview [4]

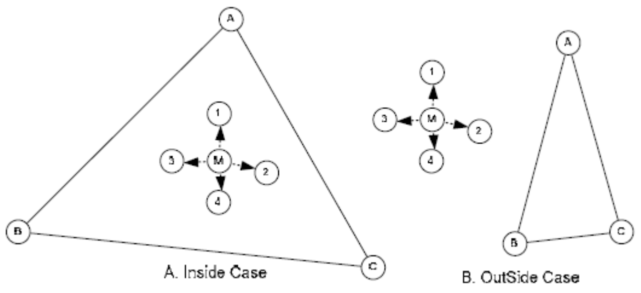


Figure 2.6: Approximate Point-In-Triangulation Test [4]

node into a very small region, and the node's estimated position is calculated as the centroid of that intersection area. Figure 2.5 shows the overview of APIT method.

To test whether a node is inside or outside of the three anchor formed triangle, two assumptions are made: i) in a certain propagation direction of a sending node, the RSSI is monotonically decreasing in an environment without obstacles; ii) the network has certain level of node density so that a movement of a node can be emulated by its neighbor node in that moving direction. By using the first assumption, we can say that if Node A is inside of the three anchor formed triangle, say $\triangle BCD$, and it moves in any direction a little bit, Node A must receives stronger RSSIs from some of these anchors, and weaker RSSI from the remaining; if Node A is outside of $\triangle BCD$, there must exist a direction in which Node A receives either stronger or weaker RSSIs from all three anchors. Using the second assumption, in a static network, the above movement testing can be emulated by checking Node A's neighbors, and thus the above statement is changed to if no neighbor of A receives stronger/weaker RSSIs from all three anchors B, C and D simultaneously, Node A assumes that it is inside triangle $\triangle BCD$. Otherwise, A assumes it resides outside this triangle. Figure 2.6 illustrates this statement. Based on the extensive simulation, the authors demonstrate that APIT achieves high localization accuracy and has a low communication cost simultaneously.

2.1.2.2 Network Connectivity

Besides using the proximity information, researches also found that the connectivity information between nodes are useful to decide node locations. At least two types of constrains can be obtained through connectivity: first, the 1 hop neighbors of a node are very likely within the distance of radius r, which is the maximum radio range, while the multi-hop neighbors are likely further than r; second, in an isotropic dense network, the path with least hop counts between a node pair is a good indicator of the physical distance between them. Based on different ideas, the connectivity based localization solutions can be divided into four categories: (i) the multilateration based distance vector (DV) solutions; (ii) the non-linear dimension reduction based DV solutions; (iii) the radio constrained solutions; (iv) and the solutions dealing with anisotropic networks.

The multilateration based DV solutions :

Dragos Niculescu and Badri Nath first proposed the idea of DV based localization approaches [58, 127]: in an isotropic dense network, the shortest path between two nodes is a good indicator of the distance between them. Therefore, if we know the least hop counts h between the two nodes and the maximum radio range r, then the distance between them is approximately $h \times r$. In 2D space, if we know the distances from a node to at least three anchors, then the location of the nodes can be determined by multilateration. In practice, h can be easily obtained by a constrained flooding. r can be replaced by the average hop distance c_i for a particular anchor i, i.e. $c_i = \frac{\sum \sqrt{(x_i-x_j)^2+(y_i-y_j)^2}}{\sum h_i}, i \neq j, j \in \{anchors\}$

Koen Langendoen and Niels Reijers summarized the DV-based solutions as three phases [128]: 1) determine the distances between unknowns and anchor nodes; 2) derive for each node a position from its anchor distances; 3) and refine the node positions using information about the range (distance) to, and positions of, neighboring nodes. The first phase has different approaches, including Sum-dist [129], DV-hop [58], Euclidean [58], DV coordinate [58] and RSD [66] . The second phase includes methods multilateration [58, 130] and Min-max [128]. The third phase can be 1 hope [131] or N-hop [129] iteratively multilateration.

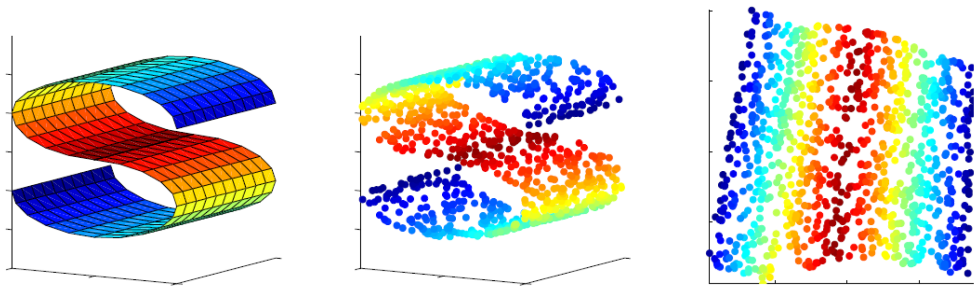


Figure 2.7: Dimension Reduction Illustration [5]. For a 3D manifold, after sampling and using dimension reduction technology, it is transferred to a set of 2D data, which preserves the neighborhood geometric structure

The work [132] analyze the theoretical localization error of connectivity based solution. It claims that the error is bounded by $||x_i - \hat{x}_i|| \leq \frac{r_0}{r} + o(1)$, where $r = o(1)$ is the radio range and $r_0 = C_d(logn/n)^{\frac{1}{d}}$ for some constance C_d which only depends on dimension d.

The non-linear dimension reduction based DV solutions :

The above DV based solutions use multilateration algorithm in their second phase to estimate each node's location. However, the dimension reduction technology can be used in this phase too, which has more tolerance to noises. The definition of the dimension reduction technology is that given N input vectors $\{x_i; i = 1, \ldots, N\}$ where each x_i is of n dimension, looking for N output vectors $\{y_i; i = 1, \ldots, N\}$, where each y_i is of dimension s ($s < n$). Additionally, the distance between x_i and x_j is preserved between output vectors y_i and y_j. Figure 2.7 shows the effect of dimension reduction technology. It reduces the high dimension (3D in the figure) data into low dimension data (2D), and keeps its underlying coordinate characteristics.

Although the purpose of dimension reduction technology is to reduce the dimension and capture the underlying data structure, the true input is a distance or similarity matrix instead of the high dimension data. Therefore, it can be used in WSN localization, where the distance matrix can be obtained by pairwise range measurements or the DV based propagation in above subsection. After obtaining the distance matrix, each sensor node can be localized by finding the coordinate which best matches the original measured distance

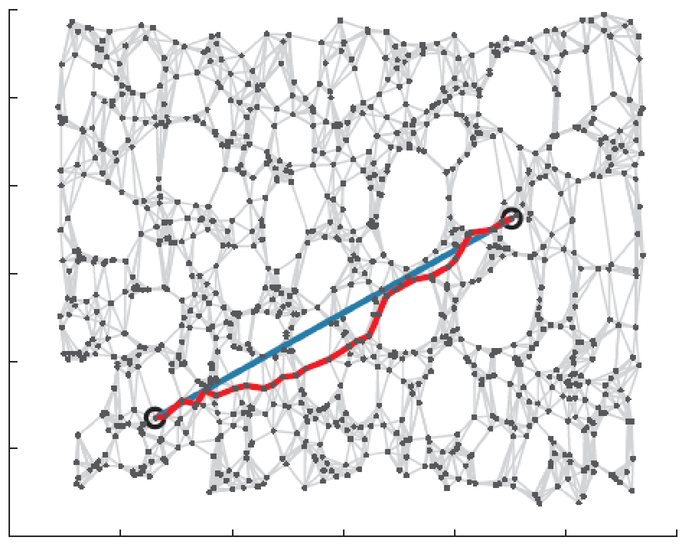


Figure 2.8: Localizing WSN by using dimension reduction technology [6]. The original pairwise range measurements are deduced by DV propagation (the red path). The estimated coordinates best fit the original measurement (the blue line).

using the dimension reduction technology.

There are several types of dimension reduction algorithms. Isomap [6] tries to maintain the shortest distance between node pairs. Its cost function is $\sum_{i,j}(\|x_i - x_j\|^2 - \tilde{\delta}_{i,j}^2)$, where x_i and x_j is the true location of nodes i and j, $\tilde{\delta}_{i,j}$ is the measured distance between nodes i and j. LLE [5, 133] (locally linear embedding) algorithms tries to preserve the local geometry information in the low dimension. It first constructs the weights w_{ij} by minimizing the cost function $\sum_i \|x_i - \sum_j w_{ij}x_j\|^2$ subject to two constraint, i.e. $w_{ij} = 0$ if x_i and x_j are not neighbors, and $\sum_j w_{ij} = 1$. Then it finds the dimension reduced output vectors by minimizing the cost function $\sum_i \|y_i - \sum_j w_{ij}y_j\|^2$, where w_{ij} is fixed and y_i is the estimated variable. Other dimension reduction algorithms are Laplacian Eigenmaps [68], Hessian Eigenmaps [134], dwMDS [135] and CCA [136], which have similar ideas but use different cost functions.

Yi Shang, et al. [93] first applies the classical MDS theory into WSN localization. Their localization approach has three steps: 1) using DV propagation to get all shortest pathes for all node pairs in the WSN; 2) then construct the all-pair distance matrix from the first

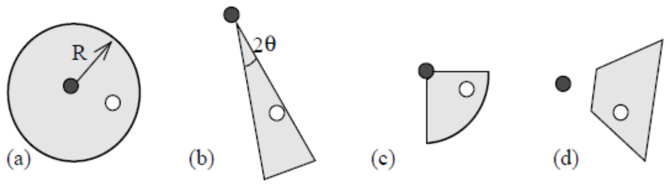


Figure 2.9: Different constraints [7]. (a) radial constraint; (b) angular constraint; (c) quadrant constraint; (d) trapezoid constraint

step and apply MDS to estimate each node's location; 3) finally based on the anchor nodes, transfer the relative map into an absolute map. The experiment results show that MDS has better performance than Hop-Terrain [131], and can achieve accurate location estimation with a few anchors. In the work [86], it compares the performances of three dimension reduction algorithms, and finds that dwMDS [135] performs best.

The above approaches are based on centralized architecture. Other works [137, 138] implemented MDS in a distributed way. The basic idea is to apply MDS in the local area (the neighborhoods of sensors), and then using the common nodes in different nodes' relative maps to make alignment, and finally determine the absolute location of each node.

Radio constraint based solutions :

Different from MDS algorithms, a number of works model WSN localiztion as a linear programming (LP) or semidefinite programming (SDP) problem. If Nodes i and j are within communication range, we can apply the constraint $\|x_i - x_j\| \leq R$, where x_i and x_j are the positions of Nodes i and j, $\|\cdot\|$ is the norm operator in 2D space (assuming localizing in 2D), and R is the maximum radio range. If we use variant radio, so that Node i can detect lower bound of radio range r_{ij} by which Node i can communicate with Node j, then the constraint can be written as $\|x_i - x_j\| = r_{ij}$. Additionally, other constraints, like angle or both angle and distance can be applied too (see Figure 2.9)

The objective function of LP or SDP is obvious, i.e. minimizing the estimation position errors. However, the key challenge is how to convert the above constraints into standard LP and SDP constraints. By using Schur complements [139], the above constraints can be transformed to:

$$\|x_i - x_j\| \leq R \Rightarrow \begin{bmatrix} \mathbf{I}_2 R & x_i - x_j \\ (x_i - x_j)^T & R \end{bmatrix} \succeq 0 \tag{2.6}$$

$$\|x_i - x_j\| = r_{ij} \Rightarrow \begin{bmatrix} \mathbf{I}_2 r_{ij} & x_i - x_j \\ (x_i - x_j)^T & r_{ij} \end{bmatrix} = 0 \tag{2.7}$$

Each distance constraint corresponds to a 3×3 matrix constraint as above, and all these constraint can be stacked as diagonal block elements into a big constraint matrix. The whole problem is a single global convex optimization problem, which can be solved by the standard SDP algorithm.

However, the drawback of the above approach is that it only works well when the anchor nodes are at the boundary of the network, since the estimated positions all lie within the convex hull of the anchors in this case. If there are anchors inside the network, it performs badly. To solve this issue, works [140, 141] introduce more constraints. By using the constraints where nodes do not have communications, it pushes away nodes. This constraint can be written as $\|x_i - x_j\| > R$. However, this is not a convex constraint. By using some relaxation trick, the works [140, 141] are able to convert the non-convex constraints into convex constraint, and thus the standard SDP algorithm can be applied. Based on this, the work [142] further improves the SDP localization approach by dividing the whole big problem into small pieces of subproblems, and thus significantly enhance the scalability.

Dealing with anisotropic networks :

The DV-based solutions in Sections 2.1.2.2 and 2.1.2.2 work well in isotropic networks but not anisotropic networks, since the shortest path between a node pair in an anisotropic network (a network with different dense or holes) is no longer a good indicator of shortest distances between them (See the shortest path between Nodes s and t in Figure 2.10). To overcome this issue, we need to capture the the non-uniform properties in an either implicit

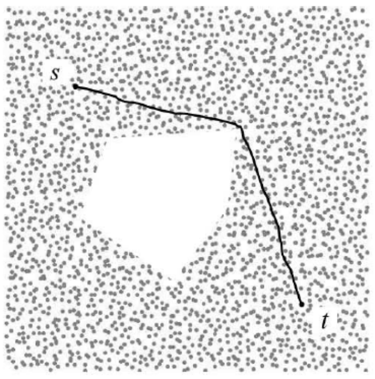


Figure 2.10: Non-uniform WSN Localization [8]

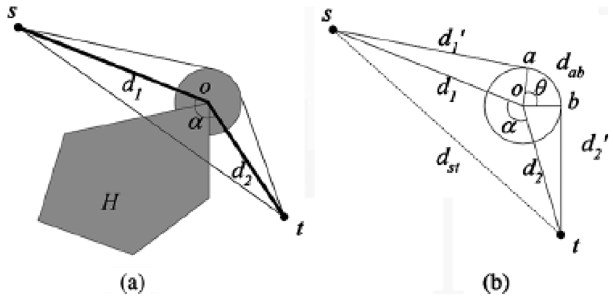


Figure 2.11: REP localization illustration [8]

or explicit ways.

To overcome the non-uniform density issue in WSN, the work [143] takes the density along the path into account and associates confidence with pathes of different hop counts (large hop number path with less confidence). The result outperforms the conventional DV-based solutions in non-uniform networks. To deal with networks with holes, the work [144] uses multidimensional scaling (MDS) to infer a proximity-distance map, which is an optimal linear transformation matrix, to represent the effect of the anisotropic characteristics of the network. When calculating the distance to an anchor node, the inferred distance is corrected by this matrix. The simulation results show that the localization performance in anisotropic networks is significantly improved. REP [8] uses another smart way to solve the hole problems in WSN localization. It utilizes the dense sensor nodes to establish a "virtual holes" around the boundary nodes, and uses geometry math to infer the distance between two nodes which have a hole (or holes) between them (see Figure 2.11). This method and also many other methods need to first detect the holes in the networks. There are also a number of papers which solve the boundary detection problem. For details, please refer to works [145–147].

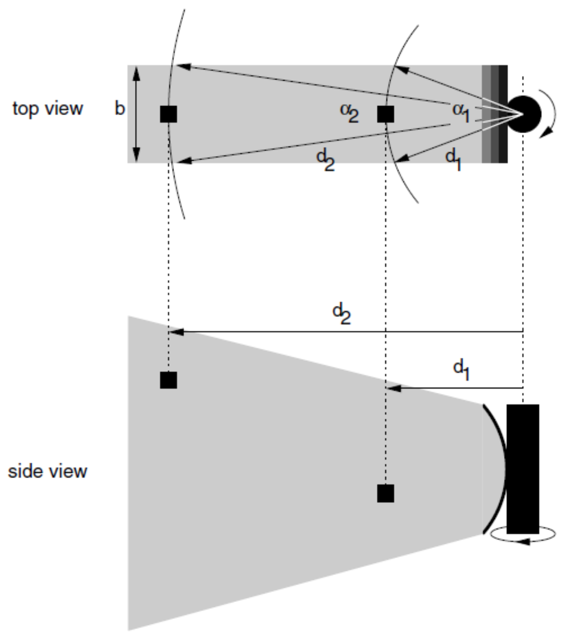


Figure 2.12: Lighthouse system illustration [9]

.

2.1.2.3 Event Based Localization

Another category of range-free localization approaches is the event-based localization solution. The key idea is to make use of the sensors on nodes to detect certain events, which are correlated to nodes' positions. After collecting the event sequences, the central party can infer the position of each node. The most significant benefit of event-based solutions is that its architecture follows the *Asymmetric Function Placement* principle [148], where the sensor node side simply keeps the necessary functionality, while all the complexities are pushed to the central party side. Therefore, the event-based solutions can be applied to sensor nodes with very constrained resources, and can usually localize the whole network in a fast and reliable way. However, the drawback of this approach is that it requires a powerful device, such as a helicopter or a laser array, to generate events, and some events require line of sight for detection.

An early event-based localization system is called Lighthouse [9]. In this system, the sensor nodes are extremely resource constrained, where each node is at cubic millimeter

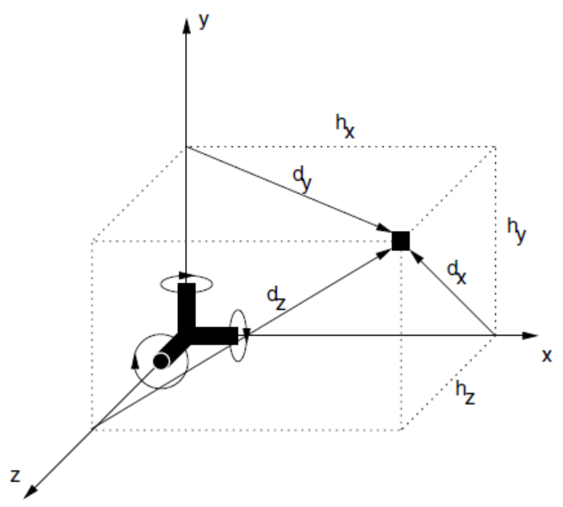


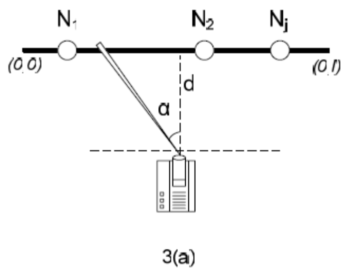
Figure 2.13: Lighthouse system 3 beams [9]

.

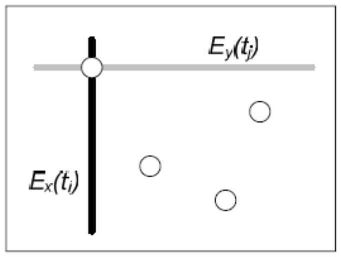
scale, can only communicate with the base station, and has a very limited power and processing capability. Therefore, the traditional WSN localization approaches cannot be used. Taking the limitation into account, Lighthouse uses an event-based solution to push most complexities into base station side, while keeping the sensor side as simple as detecting events and reporting event properties. The basic idea of Light house is as Figure 2.12 shows. Each node has an optical receiver, and the base station is equipped with a laser beam rotator, which keeps rotating at a constant angle velocity. The emitted beam of light is parallel with width b from top view, and from the side view, the angle of the beam spreads largely enough so that it can be seen from most points in space (see Figure 2.12). When the beam passes a node, the node sees the light for time t_{beam}, whose value depends on the distance d between the node and the base station. The node can also detect the beam rotation period t_{turn} when it sees the beam periodically. Then we can calculate the distance d using the following formula:

$$\alpha = 2\pi \frac{t_{beam}}{t_{turn}}, d = \frac{b}{2\sin(\alpha/2)} \tag{2.8}$$

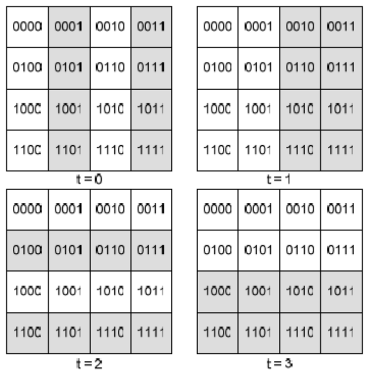
If we combine three mutually perpendicular rotation axes (see Figure 2.13), then each node



(a) Point Scan



(b) Line Scan



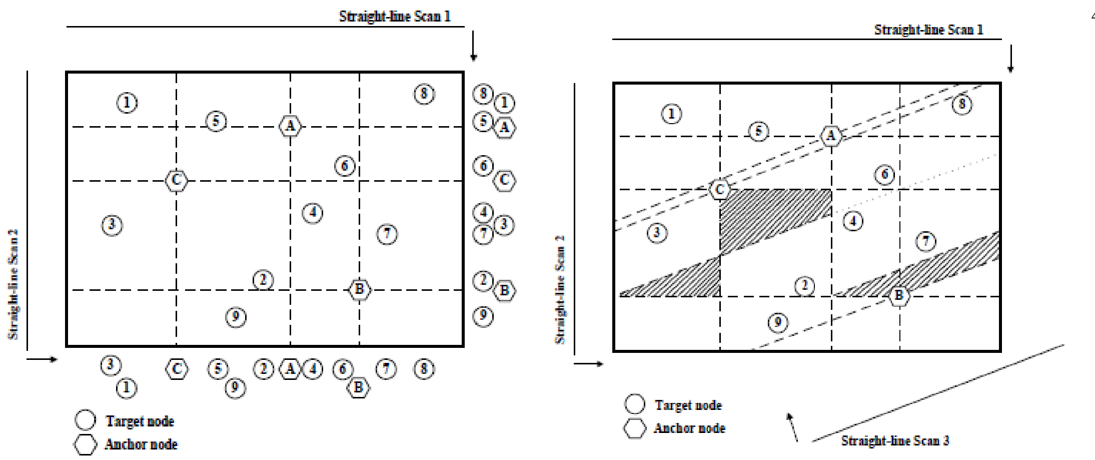
(c) Area Coverage

Figure 2.14: Spotlight system illustration [60].

can detect d_x, d_y and d_z to each axis, and thus can calculate its 3D position. The resulting accuracy of Lighthouse is about 2% of the node's distance to the base station. Another work [149] also use the similar idea, but the difference is that it infers the angle between the node and the base station, and uses trigonometric calculation to localize nodes.

Another interesting work [60], called Spotlight, uses controlled light event distribution to localize nodes. The idea is as Figure 2.14 shows: for a line of nodes, if we generate a point scan with a known constant line speed (see Figure 2.14(a)), then the location of each node on the line can be easily calculated based on the light detection time; similarly, if we scan a light beam vertically and another beam horizontally with a known constant speed across the network, then we can easily get the 2D location for each node based on the two light event detection times (Figure 2.14(b)); additionally, if we divide the whole network area as multiple small areas, and code them, and light the areas with the codes, i.e. code 0 means light, while code 1 means dark, each node can infer its location based on the detected the light and dark event sequence (figure 2.14(c)). The final outdoor localization accuracy is around 20cm.

Inspired by Spotlight, Ziguo Zhong and Tian He proposed a loosely controlled event-based system, called MSP [63]. The enhancement of MSP includes several items: it does not require time synchronization, precise event detection time reporting, constant speed of the light scanning, and mutually perpendicular light beams. Moreover, it is not limited to



(a) Vertical and horizontal scanning

(b) Scanning with any angle and reduction of possible location areas

Figure 2.15: MSP system illustration [63].

light event. Any event can be used, as long as the event occurring sequence of nodes can be detected, e.g. the ultrasound event. Taking light event for example, after light scanning from one side to another, we can get the event occurring sequence of the nodes. By using the anchor nodes, we can infer which nodes are in which sub areas divided by the anchors. After multiple scanning from different angles, we can reduce the possible location of each node to a very small area only based on the event occurring sequence, and finally localize each node by using centroid method. The illustration is shown in Figure 2.15. Figure 2.15(a) shows the obtained node sequence after a vertical and horizontal scanning; Figure 2.15(b) shows a third scan which is not perpendicular with the first two scans and the reduction of possible areas based on node sequence. Nevertheless, the type of the events is not limited to straight beams. The ultrasound event can also be used. If using ultrasound event, the polar coordinates should be used, since the ultrasound waves uniformly propagate in all directions. The shape of node possible location areas are rings in this case.

While MSP is based on loosely controlled events, another system [10] which is an extension on MSP is totally based on uncontrolled events. Its whole idea and architecture are similar with MSP, but it does not require to know the event parameters (the angle of the scanning line). It infers the parameters from the event sequence of anchor nodes. Take

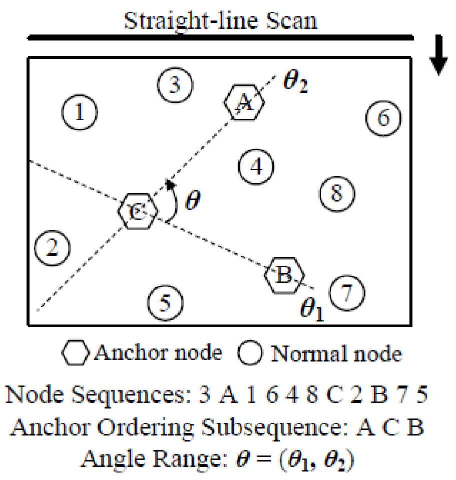

Straight-line Scan
1
3
A
θ_2
6
4
8
C
θ
2
B
5
7
θ_1
Anchor node
Normal node
Node Sequences: 3 A 1 6 4 8 C 2 B 7 5
Anchor Ordering Subsequence: A C B
Angle Range: $\theta = (\theta_1, \theta_2)$

Figure 2.16: MSP extension [10]

.

Figure 2.16 for example. If we know the event detection sequence of anchor nodes is A, C, B, then by using anchors' coordinates, we can infer that the scanning angle is between θ_1 and θ_2. We can also infer the nodes' possible location areas using the angle range and node sequence. After multiple scanning in different directions, we can use the similar technique as MSP to reduce the possible area of node into very small pieces, and then use centroid or distribution based method to deduce each node's position. Actually, both MSP and MSP extension can be modeled as a linear programming problem, and be solved in a standard optimization way [150].

Another interesting event-based solution is proposed by Radu Stoleru et al, called S-tarDust [61]. The idea of this solution is that each node is equipped with a optical retro-reflector, an aerial device projects light towards the deployed sensor network, and records an image of the reflected light. Then by using image processing and node ID matching technique, each node is localized. For this approach, it is not hard to get the location of each reflected point (sensor node) in the image, as long as the aerial device knows its position when the image is taken. The challenge part is that how to associate these locations to actual sensor nodes, i.e. the node ID matching. The work uses four types of constraints (can be combined), i.e. color constraints, connectivity constraints, time constraints and space constraints, and solve the matching problem through the relaxation technique.

2.2 Localization in Robotics

In sensor networks, the purpose of localization is to estimate each sensor node's position in the network, where the challenge is how to cooperatively collect useful information and compute on the resources constrained sensor nodes. However, in robotics field, the localization is to estimate the mobile robot's position using robot's observation (sensors such as sonars and lasers) and odometer (inertial sensors), where the challenge is to update robot's position in real time by dealing with complex computation for filtering sensor data and dealing with the correlations between robots and map features.

In robot localization, there are two different directions which are related to each other: simultaneous localization and mapping (SLAM), and collaborative localization (CL). SLAM solves the problem of localizing a mobile robot in an unknown environment and building a map simultaneously. CL is to localize multiple robots collaboratively. The related works for both of these two directions are discussed separately in the following subsections.

2.2.1 SLAM

SLAM is used to solve the problem when a mobile robot is placed into an unknown environment, how it builds a consistent map of the environment and simultaneously determines its position. The SLAM solution has been considered as a "holy grail" for the mobile robotics community.

After over two decades research, the solution of SLAM is now in a mature status, and SLAM can be considered as a solved problem. But there are still substantial issues, such as rich map (with large number of different types of landmarks) building in practical applications, need to be further studied. In the following sections, we introduce the basic SLAM framework, different algorithms, the research on computation efficiency, and the data association problem.

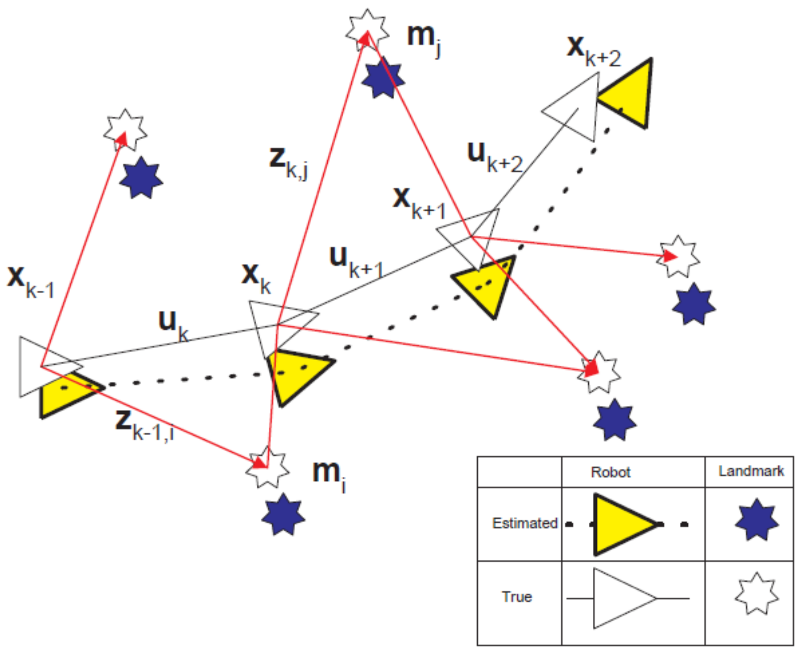


Figure 2.17: SLAM illustration. A mobile robot moves through the environment, and measures the relative position to each feature. By using the observation measurements and its odometer, it estimates the location of each feature as well as its own position. [11]

.

2.2.1.1 SLAM Modeling, Convergence Analysis and Classical Solutions

Modeling :

Figure 2.17 illustrates the process of SLAM: a robot moves through the environment, and make observations (detecting the relative position) to each feature (landmark) in the environment using sensors such as laser scanners, sonar and cameras. Based on these observations along with an odometer, which provides the velocity or acceleration measurements in real time, the robot can build a consistent map of this environment and simultaneously determine its position. Following work [11] and assuming all landmarks are static, we define the quantities at time instant k as below:

- $\mathbf{x}_k$: the state vector describing the location and orientation of the robot.
- $\mathbf{u}_k$: the control vector, which drives the robot from position $\mathbf{x}_{k-1}$ to $\mathbf{x}_k$.
- $\mathbf{m}_i$: the vector describing the location of the ith landmark.

- $\mathbf{z}_{ik}$: the observation taken from the robot's location of the ith landmark at time k. Note that we assume that the robot can identify the landmark, i.e. associate the observation data to the correct landmark.
- $\mathbf{X}_{0,k} = \{\mathbf{x}_0, \mathbf{x}_1, \cdots, \mathbf{x}_k\}$: the history of robot location from time 0 to time k.
- $\mathbf{U}_{0,k} = \{\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_k\}$: the history of control inputs from time 0 to time k.
- $\mathbf{m} = \{\mathbf{m}_1, \mathbf{m}_2, \cdots, \mathbf{m}_n\}$: the set of all landmark positions.
- $\mathbf{Z}_{0,k} = \{\mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_k\}$: the set of all landmark observations.

In probabilistic form, SLAM can be represented in the following way:

$$P(\mathbf{x}_k, \mathbf{m} | \mathbf{Z}_{0:k}, \mathbf{U}_{0:k}, \mathbf{x}_0) \tag{2.9}$$

The formula describes that the joint posterior density of the robot location and the landmark locations depend on all the observations measurements, the control inputs and the initial position of the robot. To simplify the calculation, we prefer to use a recursive form, i.e. $P(\mathbf{x}_k, \mathbf{m} | \mathbf{Z}_{0:k}, \mathbf{U}_{0:k}, \mathbf{x}_0)$ which is computed based on the density at the last step $P(\mathbf{x}_{k-1}, \mathbf{m} | \mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k-1}, \mathbf{x}_0)$ by feeding the inputs $\mathbf{z}_k$ and $\mathbf{u}_k$. Then it requires the observation and motion models.

The observation model describes the probability of sensor readings when the robot and the observed landmark locations are known. So the observations are conditionally independent given the robot's location and landmark's location. The form is as below:

$$P(\mathbf{z}_k | \mathbf{x}_k, \mathbf{m}) \tag{2.10}$$

The motion model describes the probability of a robot's new location when it has a control input $\mathbf{u}_k$ at time $k-1$. The state transition of the robot is a Markov process in which the next state only depends on the last state and the control input, and is independent of observations and the map. The probability form of the motion model is as below:

$$P(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{u}_k) \tag{2.11}$$

The recursive SLAM algorithm is represented with the following two steps. The time-update step,

$$P(\mathbf{x}_k, \mathbf{m}|\mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k}, \mathbf{x}_0) = \int P(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{u}_k) \times P(\mathbf{x}_{k-1}, \mathbf{m}|\mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k-1}, \mathbf{x}_0) d\mathbf{x}_{k-1} \tag{2.12}$$

and the measurement update step,

$$P(\mathbf{x}_k, \mathbf{m}|\mathbf{Z}_{0:k}, \mathbf{U}_{0:k}, \mathbf{x}_0) = \frac{P(\mathbf{z}_k|\mathbf{x}_k, \mathbf{m})P(\mathbf{x}_k, \mathbf{m}|\mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k-1}, \mathbf{x}_0)}{P(\mathbf{z}_k|\mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k})} \tag{2.13}$$

Convergence Analysis :

Referring to Figure 2.17, we can see that the errors of landmark location estimates are common. This is due to the fact the estimation errors are from a single source–the robot's estimation error. However, although the uncertainty for one landmark's location may be large, the uncertainty of the relative position between landmarks is small. This is because the common error between landmarks are compensated for when calculating the relative position between them, and plus since the robot makes many independent observation measurements, the uncertainty of the relative position is further diminished.

Another important property of SLAM is that the estimations of landmark positions are tightly correlated to each other. This is due to two facts. First, as Figure 2.17 shows, landmarks have common estimation errors from a single source, which means their estimations are correlated. Second, since they are correlated, when the robot observes a landmark and does the update, the new error is propagated to all the correlated landmarks even though the robot currently does not see them. This makes the landmark position estimates correlated more tightly. The work [151] proves that in the limit as the number of observations increases, the landmark estimates become fully correlated (it increases monotonically).

Based on the first property, the relative position estimates between landmarks become more and more accurate as more independent observations are taken. Based on the second property, since the correlations between landmarks monotonically increase, one update for one landmark is quickly propagated to another, which makes the convergency even faster. Therefore, the estimation of the relative map is converged under the SLAM solution, and thus the robot's position also converges because of the consistent map.

Classical Solutions :

The classical solution for SLAM uses Extended Kalman Filter (EKF) proposed by Smith et al. [152]. In this solution, the motion model is represented as

$$\mathbf{x}_k = f(\mathbf{x}_{k-1}, \mathbf{u}_k) + \mathbf{w}_k \tag{2.14}$$

where $f(\cdot)$ models robot kinematics and $\mathbf{w}_k$ is the additive, zero mean Gaussian noise with covariance $\mathbf{Q}_k$. The observation model is described as

$$\mathbf{z}_k = h(\mathbf{x}_k, \mathbf{m}) + \mathbf{v}_k \tag{2.15}$$

where $h(\cdot)$ describes the observation function, and $\mathbf{v}_k$ is additive, zero mean Gaussian noise with covariance $\mathbf{R}_k$.

The joint posterior state of robot location and landmark locations is expressed as $\begin{bmatrix} \hat{\mathbf{x}}_{k|k} \\ \hat{\mathbf{m}}_k \end{bmatrix}$, and the covariance is $\mathbf{P}_{k|k} = \begin{bmatrix} \mathbf{P}_{xx} & \mathbf{P}_{xm} \\ \mathbf{P}_{xm} & \mathbf{P}_{mm} \end{bmatrix}$. Then the time update is:

$$\hat{\mathbf{x}}_{k|k-1} = f(\hat{\mathbf{x}}_{k-1|k-1}, \mathbf{u}_k) \tag{2.16}$$

$$\mathbf{P}_{xx,k|k-1} = \Delta\mathbf{f}\mathbf{P}_{xx,k-1|k-1}\Delta\mathbf{f}^T + \mathbf{Q}_k \tag{2.17}$$

where $\Delta\mathbf{f}$ is the Jacobian of $f(\cdot)$ evaluated at $\hat{\mathbf{x}}_{k-1|k-1}$. The observation update is:

$$\begin{bmatrix} \hat{\mathbf{x}}_{k|k} \\ \hat{\mathbf{m}}_k \end{bmatrix} = [\hat{\mathbf{x}}_{k|k-1}\hat{\mathbf{m}}_{k-1}] + \mathbf{W}_k[\mathbf{z}_k - h(\hat{\mathbf{x}}_{k|k-1}, \hat{\mathbf{m}}_{k-1})] \tag{2.18}$$

$$\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1} - \mathbf{W}_k\mathbf{S}_k\mathbf{W}_k^T \tag{2.19}$$

where

$$\mathbf{S}_k = \Delta\mathbf{h}\mathbf{P}_{k|k-1}\Delta\mathbf{h}^T + \mathbf{R}_k \tag{2.20}$$

$$\mathbf{W}_k = \mathbf{P}_{k|k-1}\Delta\mathbf{h}\mathbf{S}_k^{-1} \tag{2.21}$$

and where $\Delta\mathbf{h}$ is the Jacobian of $\mathbf{h}$ evaluated at $\hat{\mathbf{x}}_{k|k-1}$ and $\hat{\mathbf{m}}_{k-1}$.

This EKF solution is optimal (regardless of the linearization part) when the control noise and the observation noise are actually Gaussian distributions, and has minimum mean square error when they are non-Gaussian distributions. The storage cost of this solution is $O(n^2)$, and the computational complexity is $O(n^2)$ for each update step, where n is the number of landmarks the robot currently knows.

2.2.1.2 Algorithms

Besides the classical EKF algorithm [152, 153], SLAM can be solved in different ways, including probabilistic form [154–158], extended information filter (EIF) [159–163], covariance integration [12], and neuro-network [164] based methods.

From the above section, it can be seen that the most significant benefit of EKF-based SLAM is simplicity, i.e. both the model representation and the recursive computation form are simple. But the disadvantages are: the computational complexity is high, which results in the bad scalability on the number of landmarks in the environment; the linearization error makes SLAM estimation diverge [165]; the model is only optimal (ignoring the linearization part) when the noise follows a Gaussian distribution, and thus may fail when the actual probability is multimodal, i.e., some environment has repetitive similar structures.

In contrast, the probabilistic form does not need the Gaussian distribution assumption, and has better capability to recover from the linearization corruption. The computational complexity depends on what algorithm it uses: for iterative optimization algorithm, it may be very fast if a good initial estimate is found; and for particle filter method, it depends on how many particles the method uses. For the extended information filter, it is a dual form of EKF. It has a very sparse information matrix (the dual form of covariance matrix), so it saves storage space. The computational complexity for updating is constant, but has a complexity of $O(n^3)$ when calculating the covariance matrix. By using approximation or tracking the whole trajectory of a robot without discarding the historical estimates, EIF can cut down this complexity to $O(1)$. The covariance integration based SLAM has very simple form, and the cost for computation and storage is very low. But, the disadvantage is that it is conservative, which means it does not use all the information it has and thus it converges slower than the above methods. The neuro-network based SLAM has a totally different form from the above methods. It does not have explicit observation and motion models, instead, it embeds this information into the weight of the neuro-network, which evolves over time. The benefits of this method is that the computation and storage are very efficient, and the data association is very reliable.

Probabilistic Form :

The work [154] models SLAM as a constrained, probabilistic maximum-likelihood estimation problem. It uses the E-step (expectation step) and M-step (maximization step) to estimate the map and robot position. In the E-step, it contains both incremental estimation and backwards revising estimation to estimate robot's historical positions. This method could converge to a local maximum in likelihood space. Experiments were conducted in cyclic environments of size up to 80 by 25 meters. Human tele-operates the robot through the environment, and tells (via a push button) the robot a pre-selected landmark is reached. The robot travels around several loops inside the building. It reduces the mapping of a building from one week (manually) to an hour.

Other probability based solutions use the particle filter. The most renowned works are FastSLAM 1.0 and 2.0 [157, 158]. FastSLAM directly represents non-Gaussian distributions by using the particle filter, a method of Monte Carlo sampling. The key idea of FastSLAM is to factor the high dimension state into a set of conditionally independent distributions, i.e. $P(\mathbf{X}_{0:k}, \mathbf{m}|\mathbf{Z}_{0:k}, \mathbf{U}_{0:k}, \mathbf{x}_0) = P(\mathbf{m}|\mathbf{X}_{0:k}, \mathbf{Z}_{0:k})P(\mathbf{X}_{0:k}|\mathbf{Z}_{0:k}, \mathbf{U}_{0:k}, \mathbf{x}_0)$, where $P(\mathbf{m}|\mathbf{X}_{0:k}, \mathbf{Z}_{0:k}) = \prod_{j}^{M} P(\mathbf{m}_j|\mathbf{X}_{0:k}, \mathbf{Z}_{0:k})$. The reason we can factor the joint state in this way is that each landmark is conditionally independent given the trajectory of the robot. The detail implementation and theoretical deduction can be found in the works [157, 158]. The storage cost for FastSLAM is $O(MK + MN)$, and the computational cost is $O(M \log N)$, where M is the number of particles, K is the time steps, and N is the landmarks the robot currently knows. Although the work [166] shows that FastSLAM actually diverges because the resampling process makes the discarded particles forget the hypothesis of maps and thus the later particles become correlated. In practice, FastSLAM is reliable and shows accurate estimation results.

Extended Information Filter :

The extended information filter has a dual formation with respect to EKF. Assuming the sate estimate and the covariances are $\hat{\mathbf{x}}_k$ and $\mathbf{P}_k$, respectively, the information matrix and the information vector are defined as

$$\mathbf{Y}_k = \mathbf{P}_{\mathbf{k}}^{\mathbf{-1}} \tag{2.22}$$

$$\hat{\mathbf{y}}_k = \mathbf{Y}_k \hat{\mathbf{x}}_k \tag{2.23}$$

Corresponding to the EKF time update of Equation 2.16, assume at time $k-1$ the state estimation is

$$\hat{\mathbf{x}}_{k-1|k-1} = \begin{bmatrix} \hat{\mathbf{x}}_{\{m,\mathbf{x}_{0:k-2}\}} \\ \hat{\mathbf{x}}_{R_{k-1}} \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{x}}_1 \\ \hat{\mathbf{x}}_2 \end{bmatrix}, \mathbf{P}_{k-1|k} = \begin{bmatrix} \mathbf{P}_{11} & \mathbf{P}_{12} \\ \mathbf{P}_{12}^T & \mathbf{P}_{22} \end{bmatrix} \tag{2.24}$$

$$\hat{\mathbf{y}}_{k|k-1} = \begin{bmatrix} \hat{\mathbf{y}}_1 \\ \hat{\mathbf{y}}_2 \end{bmatrix}, \mathbf{Y}_{k-1|k-1} = \begin{bmatrix} \mathbf{Y}_{11} & \mathbf{Y}_{12} \\ \mathbf{Y}_{12}^T & \mathbf{Y}_{22} \end{bmatrix} \tag{2.25}$$

where $\hat{\mathbf{x}}_2 = \hat{\mathbf{x}}_{R_{k-1}}$ is the robot's position estimation at time $k-1$, and $\hat{\mathbf{x}}_1 = \hat{\mathbf{x}}_{\{m,\mathbf{x}_{0:k-2}\}}$ is the remaining states including the historical robot's position estimates and landmarks' position estimates. Equation 2.24 is the EKF form and Equation 2.25 is the EIF formation. The information form of time update is as follows:

$$\hat{\mathbf{y}}_{k|k-1} = \begin{bmatrix} \hat{\mathbf{y}}_1 \\ \hat{\mathbf{y}}_2 - \Delta\mathbf{f}_{\mathbf{x}_2}^T \mathbf{Q}_k^{-1}[\mathbf{f}(\hat{\mathbf{x}_2}, \mathbf{u}) - \Delta\mathbf{f}_{\mathbf{x}_2}\hat{\mathbf{x}_2}] \\ \mathbf{Q}_k^{-1}[\mathbf{f}(\hat{\mathbf{x}_2}, \mathbf{u}) - \Delta\mathbf{f}_{\mathbf{x}_2}\hat{\mathbf{x}_2}] \end{bmatrix} \tag{2.26}$$

$$\mathbf{Y}_{k|k-1} = \begin{bmatrix} \mathbf{Y}_{11} & \mathbf{Y}_{12} & \mathbf{0} \\ \mathbf{Y}_{12}^T & \mathbf{Y}_{22} + \Delta\mathbf{f}_{\mathbf{x}_2}^T \mathbf{Q}_k^{-1} \Delta\mathbf{f}_{\mathbf{x}_2} & -\Delta\mathbf{f}_{\mathbf{x}_2}^T \mathbf{Q}_k^{-1} \\ \mathbf{0} & -\mathbf{Q}_k^{-1}\Delta\mathbf{f}_{\mathbf{x}_2} & \mathbf{Q}_k^{-1} \end{bmatrix} \tag{2.27}$$

Corresponding to the EKF observation update of Equation 2.18, assume the state before the observation is

$$\hat{\mathbf{x}}_{k|k-1} = \begin{bmatrix} \hat{\mathbf{x}}_{\{\mathbf{m},\mathbf{x}_{0:k-1}\}-\{\mathbf{m}_k\}} \\ \hat{\mathbf{x}}_{\mathbf{x}_k,\mathbf{m}_i} \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{x}}_1 \\ \hat{\mathbf{x}}_2 \end{bmatrix}, \mathbf{P}_{k|k-1} = \begin{bmatrix} \mathbf{P}_{11} & \mathbf{P}_{12} \\ \mathbf{P}_{12}^T & \mathbf{P}_{22} \end{bmatrix} \tag{2.28}$$

$$\hat{\mathbf{y}}_{k|k-1} = \begin{bmatrix} \hat{\mathbf{y}}_1 \\ \hat{\mathbf{y}}_2 \end{bmatrix}, \mathbf{Y}_{k|k-1} = \begin{bmatrix} \mathbf{Y}_{11} & \mathbf{Y}_{12} \\ \mathbf{Y}_{12}^T & \mathbf{Y}_{22} \end{bmatrix} \tag{2.29}$$

where $\hat{\mathbf{x}}_{\mathbf{x}_k,\mathbf{m}_i}$ is the block state estimation for the robot and the landmark i at time k, and $\hat{\mathbf{x}}_{\{\mathbf{m},\mathbf{x}_{0:k-1}\}-\{\mathbf{m}_k\}}$ is the remaining state elements. Then, the information form of observation

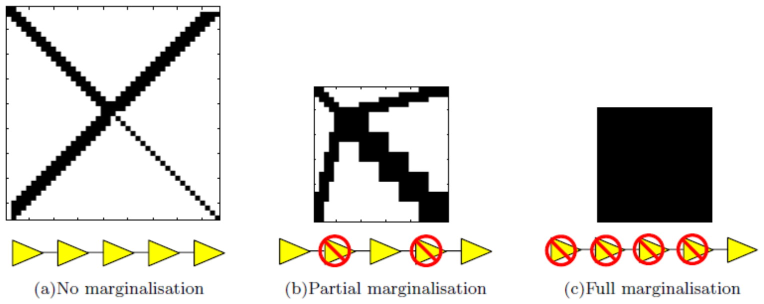


Figure 2.18: Information matrix with tradeoff the number of the states and the density of the matrix [11]

.

update is:

$$\hat{\mathbf{y}}_1^o = \begin{bmatrix} \hat{\mathbf{y}}_1 \\ \hat{\mathbf{y}}_2 + \Delta\mathbf{h}_{\mathbf{x}_2}^T \mathbf{R}^{-1}(\mathbf{z} - \mathbf{h}(\hat{\mathbf{x}}_2) + \Delta\mathbf{h}_{\mathbf{x}_2}\hat{\mathbf{x}}_2) \end{bmatrix} \tag{2.30}$$

$$\mathbf{Y}^o = \begin{bmatrix} \mathbf{Y}_{11} & \mathbf{Y}_{12} & \mathbf{0} \\ \mathbf{Y}_{12}^T & \mathbf{Y}_{22} + \Delta\mathbf{h}_{\mathbf{x}_2}^T \mathbf{R}^{-1} \Delta\mathbf{h}_{\mathbf{x}_2} & \mathbf{Y}_{23} \\ \mathbf{0} & \mathbf{Y}_{23}^T & \mathbf{Y}_{33} \end{bmatrix} \tag{2.31}$$

From the above Equations 2.26 and 2.30, since the information matrix Y is a sparse matrix, the computational complexity for both time update and observation update is constant if the evaluation state estimate $\hat{\mathbf{x}_2}$ is known. However, to recover the state estimate, it needs the inversion operation on information matrix $\mathbf{Y}$ which takes $O(n^3)$ time in a naive implementation. To avoid the matrix inversion, the work [159] uses an iterative optimization method to find $\hat{\mathbf{x}_2}$, which only takes a constant time, and the work [161] proposes a sub-optimal partial state recovery method to achieve constant time.

Another property of EIF is that if the historical state estimates of the robot are marginalized away, the information matrix becomes dense, and the time update and observation update is not constant anymore (see Figure 2.18). To avoid the density of the information matrix, the work [159] uses the mode-consistency method to sparsify the information matrix, which is to essentially drop the weak links. And the works [161,163] keep all historical

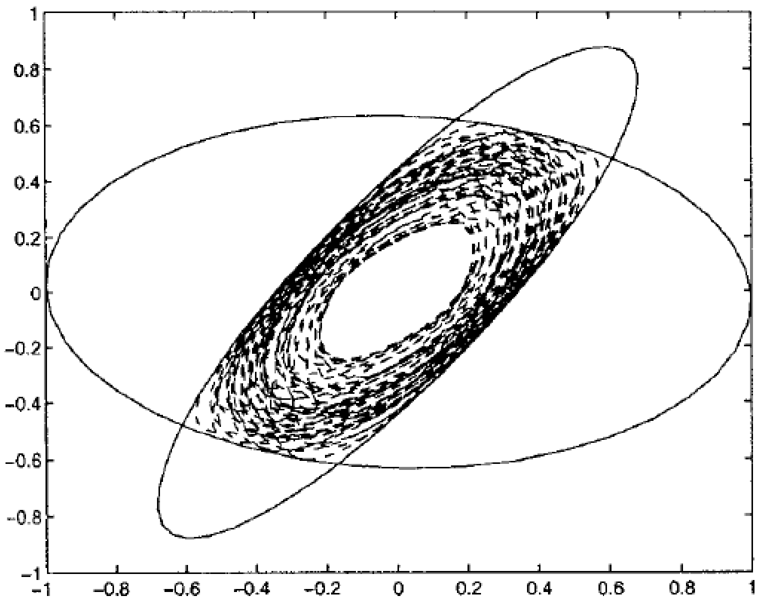


Figure 2.19: Covariance Integration Illustration. The two big ellipses are the covariance of two random variables, the center ellipse is the optimal fusion, and the dashed ellipses at the intersection areas of the two big ellipses are the results of covariance integration by different choices [12]

.

estimates to keep the sparseness of the matrix.

Covariance Integration and Neuro-network SLAM :

The work [12] proposes a fusion algorithm called the covariance intersection algorithm (CI), which fuses two random variables without knowing their correlation. The geometric meaning of CI is to form an estimate from the convex combination of the means and covariances of the two random variables (see Figure 2.19). The estimation result is consistent, but may not be optimal. The simulation results show that in the case without the knowledge of the correlations between input variables, the Kalman filter diverges, but CI stays consistent.

Another work [164] uses neural network and an evolutionary optimization algorithm to solve SLAM. This algorithm does not require a prior knowledge of the robot and the sensor model, and thus has less searching space than other neural network based algorithms. The simulation and real experiments show its better performance than the EKF based SLAM and Fast SLAM.

2.2.1.3 Computation efficiency

Section 2.2.1.1 shows that EKF-based SLAM has a computational complexity of $O(n^2)$ at each observation update step, where n is the number of landmarks the robot knows. This high complexity causes a scalability issue when the number of landmarks is high. The root cause of the high computational complexity is that all landmarks are correlated with each other through the historical robot states, so when a landmark is updated, all the other correlated landmarks have to be updated. EKF uses the dense covariance matrix ($n \times n$) to track all the variances and covariances, and thus one observation update will update the whole covariance matrix with $O(n^2)$ elements.

According to the above causation, there are two approaches to reduce the complexity: first, avoid using EKF so that the covariance matrix is not directly used for the updating; second, do not update or delay the update of all landmarks when one landmark is observed. For the first approach, we can use the EIF, CI or neuro-network algorithms. They all do not use a covariance matrix: EIF only keeps a sparse information matrix; CI only keeps the variance-covariance regardless of covariance; neuro-network also only tracks variance-covariance. For more details of these algorithms, please refer to the works [12, 154–164]. Furthermore, the work [167] uses a conjugate gradient method to reduce the inversion of the matrix from $O(n^3)$ to $O(n \log n)$, and the work [168] uses a square root method to further reduce the computational complexity of EIF.

The second approach refers to submap-based SLAM solutions [169–173]. Instead of keeping all landmarks in the global coordinate frame, these solutions divide the whole map into many small submaps. The coordinates of each landmark are represented in the local coordinate frame. One observation update now only updates the correlated landmarks in the local map. The global map is updated less frequently either in period or at the time when a closed loop is found. The benefits of this solution is that the computational complexity is reduced significantly, and many of the algorithms using this technique are constant time. But, the disadvantages are that not all the landmarks are updated in real time, and the updates are usually conservative (not optimal).

2.2.1.4 Data association

It has been known that in SLAM, to update and add landmarks, any algorithms or frameworks (EKF or EIF) require to know the correct landmarks during the measurement update. The process for identifying the landmarks of the sensor measurements is called data association. There are different cases for data association: false positive–considering there exists a landmark which is actually not existing, false negative–considering there are no landmarks but there actually is, mistakenly considering Landmark A to Landmark B, correctly detecting a new landmark, and correctly detecting an existing landmark. Among the mistaken data associations, the false positive and the mistaken identification are most harmful, in which one point failure causes the whole algorithm to be corrupted. The process of detecting an existing landmark after a relatively long trip is also called "closed loop", which can correct the accumulative estimation error significantly if the data association is correctly made. Therefore, data association is another important component for SLAM.

A naive data association implementation [174] is to treat each measurement-to-landmark independently, i.e. if a new measurement indicates that a landmark is within some old landmark's acceptable uncertainty range, it is considered to be the old one. However, as the robot travels for a long time and accumulates a large error, this method becomes very unreliable.

A more sophisticated method is to consider a batch of data associations together. The work [175] uses a map correlation method to match a batch of historical scans with the current scans. The work [176] proposes a method called a joint compatibility test, which uses the criterion of correlations between the innovations on a set of pairings to make the data association. Combining with a branch and bound search algorithm, it results in a very robust solution. Another work [177] uses graph theory and models the SLAM problem as a minimum energy optimization problem. The data association problem in this model is reduced to checking the energy change of the graph, which uses all the information collected.

Another types of data association is based on the appearance signature, which is usually used in vision based SLAM. The benefit of this method is that it usually can be used independently without referring to a robot's position. The work [178] proposes an image-based place recognition method based on matching image signatures. A signature is defined as an array of values which are calculated by a measurement function on each sub image. By using a set of predefined measurement functions, an image generates a number of signatures which are used for late matching. The work [179] combines both a laser scanner and a camera for SLAM, in which the laser scanner is used to build 3D geometric map, and the camera is used for closed loop detection. The closed loop detection also uses a similar technique as the signature matching method.

A more complicated and reliable solution for data association is called multihypothesis data association. The basic idea is that it simultaneously tracks a set of separate data association hypotheses with an ever-branching tree. This method is reliable, but costly on computation and storage resources. The particle filter based solutions FastSlam 1.0 [157] and FastSlam 2.0 [158] intrinsically belong to this type.

2.2.2 Collaborative Localization

Different from SLAM, collaborative localization (CL) solves the localization problem by only using the relative position information (the relative pose, orientation, bearing and/or range) through collaboration. CL is usually used in scenarios where the map estimation is not important or the computational complexity is too high to track both agents and maps.

Implementing CL using a centralized architecture is trivial: just forward all positioning measurements to the central place, and use EKF-based Bayesian framework to estimate all agents' positions and the joint covariance matrix. However, it is hard to implement CL in a distributed way, because one agent's position estimate is correlated to others. If one agent's estimate is updated, all the other related agents' position estimates need to be updated. The existing distributed CL algorithms can be divided into two categories: the consistent CL, and the non-consistent CL. The consistent CL means the algorithm strictly follows the

Bayesian theorem, using each measurement only once, and the resulting estimate is optimal (Gaussian model) or MMSE (non-Gaussian model); the non-consistent CL does not strictly follow the Bayesian theorem, where one measurement may be used multiple times, and thus its estimated covariance matrix is usually over confident (its estimate is actually not as accurate as it claims). The non-consistent CL is realized through approximation, and used to simplify the computation complexity.

For consistent CL, the early works' idea [75,180,181] is to divide robots into two Group– A and B. When Group A is moving, Group B keeps stationary, working as landmarks for Group A. After Group A moves for a distance, they exchange their roles, i.e. Group B moves and Group A keeps still. Although this idea is demonstrated to work, the biggest disadvantage is that it restricts agents movement patterns, which is impractical in some real applications. Later works eliminate this constraint by using Bayesian estimators. They can be further divided into three categories: the partially distributed algorithm, the decentralized synchronized algorithm, and the decentralized asynchronous algorithm. Note that the difference between "distributed" and "decentralized" is that the former may distribute partial computation to each agent, but still requires a central place for data fusion, while the latter does not involve a central place at all. For "synchronized" and "asynchronous", they indicate whether the method requires a full connection among all agents or not. A synchronized algorithm needs some intermediate results to be broadcast to all agents for synchronization, otherwise the algorithm cannot continue; while an "asynchronous" algorithm does not require the full connection among agents, and agents only exchange information when they meet.

A representative of the partially distributed CL algorithm is presented in work [78]. The essence of this work is to process the egocentric data (e.g. GPS, inertial measurements) locally and fuse the inter-agent measurement at a central place. This approach is based on the information filter, and the "product rule decomposition" is used to packetize the infinite Markov chain on each agent so that the fusion center can fuse these partial chains properly. Although this algorithm is efficient and asynchronous, the drawback is that it requires a central place to perform the data fusion. The work [182] is similar, where it also uses the packetization idea to reconstruct the server pose-graph on the client side, but the proposed method is more robust to packet loss and bandwidth limitation. Also work [183] is another partial distributed implementation for pedestrian collaborative localization.

For the decentralized synchronized approach, work [184] uses an EKF-based approach, and proves that to achieve an equivalent estimation to the centralized approach, agents only need to broadcast the cross-correlation terms of the joint covariance matrix when they are doing an update. Work [185] is based on an information filter, and uses an algorithm that employs distributed Cholesky modification to achieve the decentralized estimation. It requires each agent to forward rows of the Cholesky factor of the information matrix to another agent in some determined order. Although these two methods do not require a central place for data fusion, they both require each agent to share some intermediate information to other agents immediately after it finishes updating (synchronized), otherwise, the whole process cannot move forward. This is also impractical for many scenarios where agents may be often out of the communication range.

For the decentralized asynchronous approaches, work [77] lets each agent maintain separate EKFs for any possible combination of all involved agents so that when doing the fusion, only un-correlated estimates are fused (thus consistent). Therefore, if the total agent number is n, then each agent potentially needs to maintain 2^n EKFs. The drawback of this approach is that it is not scalable–the computation and memory costs are exponential to the number of total agents, and additionally the algorithm is not making use of all available measurements. Another work [186] which is very similar to our proposed work is to let each agent maintain a local measurement graph, and delay the estimation till needed. Since the previous states are not recoverable once they are fused (marginalized), the authors present a method to check when and which states can be safely fused. By using this method, each agent's local graph is reduced by marginalization without breaking the consistence. Although this method is effective and consistent, the disadvantage is that it does not guarantee the boundness of memory usage and computational complexity when agents leave the group and never come back.

The distributed CL algorithms are complicated since they need to undergo significant effort to keep the estimation consistent. To reduce the complexity, approximate decentralized CL algorithms are proposed. An approximate particle filter based CL approach is presented in work [187]. The essence of this work is to ignore the correlation between agents, which breaks an n-dimension problem into n one-dimension problems. In this approach, each agent only needs to maintain its own particles, and when an inter-agent measurement is made, it combines the other one's PDF by assuming they are independent. Another similar work is presented in [188]. The essential idea is the same as work [187], but the difference is that it uses the first two moments for a agent's position estimate, instead of particles. Although these two approaches try to overcome the over-confidence problem by dropping continuous measurements between two agents, the final results show agents' estimates are still over-confident. Another different approach is presented in the works [189, 190], which uses the iterative Jacobi distributed algorithm to estimate each agent's position. In each iteration, each agent assumes its own and neighbors' estimates are correct, updates its own position estimate and broadcasts it to neighbors. The resulting estimates eventually converge to the centralized method. Although the algorithm is very simple and effective, the drawbacks are: first, it only provides the position estimate, but not the covariance matrix; second, the simulation is only based on agents with frequent communications, so the performance may drop when agents are communicating infrequently.

2.3 Indoor Hybrid Localization Systems

Outdoor localization can be considered as a solved problem, in which Global Navigation Satellite System (GNSS), such as U.S.'s GPS, Russia's GLONASS, European Union's Galileo or China's Beidou system, provide reliable position estimates for outdoor receivers with an accuracy about 3-5m and an update rate about 1 second. However, GNSS cannot be used indoors due to the requirement of line of sight between the receiver and satellites.

Currently there does not exist a reliable and generic solution. The main challenges are the complex and volatile indoor environment, which causes the ranging signals, such as electromagnetic waves or ultrasonic, to have unpredictable attenuation and multipath effects. To improve the reliability and accuracy, a hybrid system, which integrates positioning information from different sources, is usually needed for indoor localization.

A hybrid system usually consists of three parts: the positioning subsystems, the auxiliary positioning constraints and localization algorithms. The positioning subsystems obtain the position related measurements, such as relative position measurements (the relative angle between two nodes, or the relative displacement of a node in a time interval) or absolute position measurements (the coordinate of the node). In a hybrid system, multiple positioning subsystems coexist, providing positioning measurements from different sources to compensate for the shortcoming of each other. For example, a foot-mounted inertial navigation subsystem and a WiFi fingerprint based localization subsystem can be integrated to localize a pedestrian so that it has good short term position estimation, high update rate as well as a non-drifting property. The auxiliary positioning constraints are information which constrains the physical possible positions of nodes. For example, indoor maps provide important information which constrains that nodes' trajectory should not cross the walls; maximum velocity constrains the possible area a node could be based on its previous position; the gravity in a particular place and the earth's rotation rate are needed for inertia based localization. The localization algorithms are also significantly important. A sophisticated algorithm can outperform a naive one by nearly 100 times [191]. In indoor hybrid systems, traditional optimization algorithms and the Beyesian inference framework can be applied. The former includes linear/nonlinear analytic optimization techniques and iterative optimization techniques, such as the Gauss-Newton algorithm and the Levenberg-Marquardt algorithm. The latter includes Kalman filter, extend Kalman filter, unscented Kalman filter, information filter, particle filter, etc.

Besides accuracy and reliability, system assumptions matter significantly since it directly decides which techniques can be used and which scenarios the system can be applied to. From an infrastructure viewpoint, the system assumptions include three dimensions: on board device requirements, infrastructure requirements and auxiliary information requirements. The on board device requirement means if the system requires the node to be localized to carry particular devices for either communication or positioning purposes, e.g. a foot mounted navigation device or an inertial sensor equipped smart phone; the infrastructure requirement indicates if the system requires some particular devices to be setup in advance, e.g. WiFi access points (APs) or pre-deployed wall mounted Radio-frequency identification (RFID) tags; the auxiliary information requirement mainly refers to whether the system needs a site map. We can see that one assumption which looks reasonable in one scenario may not hold in another scenario. For example, the indoor firefighter localization system cannot assume there is positioning infrastructure or map information available for the site. In contrast, a positioning system for an office building can assume there are multiple WiFi APs and even the floor plan of the building available, but cannot assume each user has a foot-mounted inertial navigation device.

The following subsections first introduce the basic components for indoor localization systems, i.e. the inertial pedestrian navigation component and the infrastructure based localization component. Then the recent hybrid localization systems are divided into three categories based on the above system assumptions, which include localization with inertial devices and infrastructure, localization with inertial devices and maps, and localization with inertial devices, infrastructure and maps.

2.3.1 Inertial Pedestrian Navigation and Infrastructure based Indoor Localization

For indoor localization in this section, if not specifically pointed out, it refers to indoor pedestrian localization. For a hybrid indoor localization system, the on board devices are usually used as a dead reckoning component which is based on inertial sensors, and infrastructure is used for either absolute positioning or relative range measuring.

2.3.1.1 Inertial Pedestrian Navigation

For pedestrian localization, there are different types of on board devices. They differ in sensor types, implementation hardware platforms and mounted positions. For sensor types, they may or may not have barometers, magnetic sensors, and doppler radars, but all have inertial sensors, i.e. accelerometers and gyroscopes. The more sensors it has, the more complex the fusion algorithm is and the more accurate it could be. In addition, these sensors differ significantly in sensing accuracy, with the trade-offs of cost and size. The implementation hardware can be localization customized sensor boards or off-the-shelf smart phones. What platform the system uses depends on scenarios. For example, you cannot enforce enterprise employees to use foot-mounted inertial sensors for daily tracking inside an office building, whereas smart phones are more appropriate in this case. In contrast, firefighters may use foot-mounted inertial devices for accuracy purposes. By using different hardware platforms, the mounted positions also differ, which can be on foot, on torso, in pocket or held by hands. The placement position decides the algorithm to use. For example, the zero-velocity-update algorithm can be used by foot-mounted inertial devices, but is not proper for torso mounted inertial devices, since the signal patterns are totally different when a pedestrian walks under these two cases.

For cost and portability purposes, the inertial sensors used by pedestrian tracking are based on Microelectromechanical systems (MEMS) technology. These types of sensors are usually cost-efficient and tiny size, but not accurate. The noise for the accelerometer and the gyro is not small enough. To estimate the position and the orientation, we need to integrate the accelerometer readings twice against time, and integrate the gyro readings once against time. Therefore, with simple integration, the position estimation error is accumulated in the rate of cubic of time and the orientation estimation error is accumulated in square of time, which causes the estimation error to be as large as tens of meters within only tens of seconds.

To improve the accuracy, corrections need to be fed back to the inertial system. Foxlin

[192] first introduces the zero-velocity-update (ZUPT) approach for pedestrian tracking. The key idea is that for each step, a pedestrian's one foot must be in a still phase at some time point, and its velocity should be zero. If the velocity estimation is not zero at the still time, it must be caused by the accumulated sensor noise during that step. So by adding the error sates for position, velocity, accelerate, orientation and angular rate, and feeding back the velocity estimation error at each still time, the position and orientation errors are corrected. By using this approach, the estimation error is not dependent on time any more, but on the number of steps, which is usually under 5% of the total travel distance. Following this work, further study and improvements are made. Estefania Munoz Diaz et al. study the minimum frequencies of the accelerometer and gyroscope required by the inertial system. The results show the minimum rate for the accelerometer is between 200 Hz to 300 Hz, while for the gyroscope it is 100Hz. Issac Skog et al. [193, 194] compares the performance of different zero velocity detection algorithms and the mounting places for the inertial sensors. A.R. Jiménez [195] includes the zero-angular-rate-update (ZARU) approach into the system to further improve the accuracy. A.R. Jiménez [196] uses heuristic attitude prediction to overcome the accumulated yaw error, which cannot be corrected by the original ZUPT method. Alonzo Kelly et al. [197] use two shoe-mounted sensors to gain more constraints on the estimation algorithms.

Besides using the customized foot-mounted inertial devices, today's smart phones usually have these inertial sensors embedded already. But we cannot directly apply the foot-mounted inertial navigation algorithm to smart phone based navigation, since their placements are different and the signal patterns are totally different. Based on extensive experiments, research has found that step detection based solutions are more reliable for smart phones. Using the acceleration data, steps can be detected. Various algorithms can be used, such as peak detection [198], zero crossing [199] and auto-correlation [200]. The total distance traveled can be estimated by hip rotation [201] or through step length estimation [202] which is a linear function of the frequency of stepping. The direction of the travel can be estimated by either the gyroscope or a compass. The work HiMLoc [203] uses zero crossing and compass for pedestrian tracking, which results in approximate at 15% error of the total traveled distance. Note that when the smart phone is held at different positions and users do different activities, such as normal walking, stair walking and running, the algorithms or the system parameters are different. There are already a number of works which solve the activity recognition problem [203–205].

2.3.1.2 Infrastructure based Indoor Localization

The infrastructure based indoor localization component is to localize pedestrian with pre-installed devices which are either indoors or outdoors. Most of the infrastructure based systems provide the absolute coordinates, although some of them only provide the relative position measurement. By providing the absolute position, this component can work as an independent localization system, but in a hybrid localization system, it is usually used to correct the accumulated error of the inertial tracking component.

For systems providing absolute coordinates, there are several different types of infrastructure, such as WiFi access points (APs), mobile stations, RFIDs, and UWBs. The most popular one is using WiFi APs, since it is the most common infrastructure in modern office buildings. There are basically three algorithms for WiFi based localization: the RSS based, the fingerprint based and the time of flight (TOF) based methods.

For the RSS based method [206, 207], it estimates the distances between the receiver and the APs based on the received signal strength, and then uses multilateration or the least square error method to estimate the receiver's location. The basic propagation model like Equation 2.2 is used, and the parameters of this formula are pre-estimated.

For the fingerprint based method [206–208], the idea is to divide the interested area into grids, and build up a database which contains a RSS profile for each grid. When a receiver scans the WiFi RSS to each available AP at some place, it compares the current profile with the ones in database, and estimates its position as the one with the most similar RSS profile in the database. Since this method requires pre-collecting the RSS profile for each grid to build up the database, which involves significant amount of work, later works [208–212] rely on a crowd-source method, in which end users contribute to collecting the RSS profiles and updating the database entries. The experimental results show that the fingerprint based method performs better than the RSS based one [206], which usually provides a room level accuracy.

The above two methods are essentially RSS based, which is not reliable in indoor environments due to multipath effects and the volatile property of the environments. In order to avoid these uncertainties, a time of flight method is used together with a WiFi infrastructure. A round trip time (RTT) is measured to calculate the distance between a receiver and an AP. Guenther and Hoene [213] use ICMP-ping requests and the IEEE 802.11 time synchronization function (TSF) to estimate the RTT, which results in a mean ranging error about 8 meters. However, this method requires an additional monitor node to capture the IEEE 802.11 control frames. Ciurana et al. [214] use a Data-Acknowledgement (DATA-ACK) based on ICMP-ping requests, and uses a customized 44MHz WLAN clock to measure the RTT which avoids the system delays. The resultant accuracy has a mean error of 0.81 m. Another solution [215] uses Intel's time stamp counter (TSC) as a high resolution timer to measure the RTT of the outgoing DATA and ACK frames. The resultant accuracy is about 1.7 m. Casacuberta et al. [216] compare three WiFi TOF methods using different time measuring functions: the standard TSF, the TSC, and the invoking TSF directly on a WiFi card. The mean errors of these three methods are 2.8, 1.5 and 4.4 meters, respectively. Lorenz Schauer et al. [217] use NULL-ACK messages combined with TSC to estimate the RTT, which has an error of 1.33 meters in the best case.

The mobile station based localization is similar to the WiFi based solution, it can use RSS based or fingerprint based approaches, however, due to the fewer number of mobile stations, the localization error is much bigger, which reaches hundreds of meters [218, 219]. RFID based localization systems can use RSS [123, 220], angle of arrival [191, 221], synthetic aperture [222,223] and proximity [224,225] to localize objects. However, for passive RFID tags, the reader usually has a very limited range (less than 10m), which brings more constraints on the applications. As described in Section 2.1.1.2, UWB is an accurate indoor ranging technique. It uses TOF, TDOA and AOA to localize the receiver. In a 2D plane, it requires at least 3 UWB anchors, while in 3D space, it needs at least 4. Due to its short signal range, in practice, the number of UWB anchors is much more than that. Although the UWB has accurate performance, the drawback is that the device size is usually large and the cost is expensive.

The above techniques are used for absolute positioning, however, the RFID and UWB based methods can be used for relative position measuring, i.e. instead of providing the absolute coordinate estimate for a target, it can only provide the range or relative angle to a target. By using this methodology, the number of anchors required by the system can be dramatically reduced. Furthermore, by tightly coupling these relative measurements into the system (through Bayesian framework), it usually has better performance than the absolute positioning solutions [226, 227].

2.3.2 Localization with Inertial Devices and Infrastructure

By integrating an inertial device and a infrastructure based localization subsystem, a hybrid system inherits the high update rate and accurate short distance estimation properties from the inertial device as well as the non-drifting over time property from the infrastructure based subsystem. The inertial device and the infrastructure based subsystem can be any type of the above solutions. There are two types of integration methods, loose coupling and tight coupling. In the former approach, two components work independently, each of which outputs a position estimate at some time point, and a Bayesian framework combines them to produce a better estimate. In the latter one, two components work tightly, the relative position measurements from the infrastructure based subsystem are directly fed into the Bayesian framework. While position estimates are being obtained, the parameters (such as parameters in the signal propagation model) of both systems are also updated over time.

Wennan Chai et al. [226] propose an indoor hybrid localization system which integrates a low-cost MEMS unit inertial device and a WiFi based localization subsystem. The work describes three WiFi based localization models: the fingerprints-based approach, the wall attenuation factor (WAF) based approach, and simplified propagation model-based approach. Corresponding to these three models, there are possibly three types of hybrid systems. The first two systems loosely couple the inertial navigation component and the WiFi localization components using the Unscented Kalman Filter (UKF), while the third one tightly couples with the third WiFi approach, and the WiFi propagation parameters are directly estimated by the UKF. The result shows that INS/WiFi integration localization approach bounds the errors, and the third model has the best accuracy.

Nobuo Nakajima et al. [228] propose a hybrid localization system which integrates an inertial pedestrian navigation device and a number of customized antenna array based absolute positioning devices. The absolute positioning device consists of two antennas: one emits a pencil shape beam to detect the distance between the receiver and the emitter; the other emits dual beams which forms a narrow dip signal at front of the emitter to detect the receiver when it is right in front of the emitter. The experiment shows that the localization error is around 10 m after walking 800 m.

Valentin Radu et al. [203] present a localization system using a mobile phone and WiFi infrastructure. The mobile phone is for not only pedestrian inertial navigation, but also activity classification which assists the PDR navigation. The WiFi infrastructure uses a fingerprint based method to estimate the absolute position. A particle filter is used to combine all this information. The experiment shows that the median localization error is less than 3 meters in most cases.

Christian Ascher et al. [227] use UWB and inertial sensors for hybrid localization. A torso mounted sensor system with accelerometer, gyroscope, magnetic and barometer sensors is used for step based dead reckoning. The UWB devices use TDOA to estimate the range to the receiver. The authors compare the performance of the loosely coupled and the tightly coupled Kalman filter, and find that the tightly coupled one outperforms the loosely coupled one.

Markus Langer et al. [229] utilize the weak signals of GPS in indoor environments to improve the performance of inertial navigation. The system uses a deeply acquisition and re-acquisition routine for GPS, and also weights the GPS measurements according to the signal to noise ratio. By coupling the weak GPS signal with the torso mounted inertial device, the localization accuracy is improved.

Francisco Zampella et al. [230] combine inertial sensors, RFID and UWB for hybrid localization. The ZUPT based foot mounted inertial sensors are used for dead reckoning. The RFID tags are pre-deployed on the wall, while the readers are carried by the pedestrian. The RFID subsystem uses RSS for distance mapping to estimate the range between the tag and the reader. The UWB subsystem uses TDOA and AOA to estimate the absolute position of the pedestrian. A novel outlier rejection algorithm is proposed to improve the accuracy, and a simplified particle filter is used to fuse all these measurements in real time. The experiment shows that the localization error is within 2 meters in 90% of the time.

2.3.3 Localization with Inertial Devices and Map

A map is another useful information for indoor localization, which can often improve the indoor localization accuracy by tens of times. With map information, the estimated position can be constrained within the only possible areas. Since indoor environments often have symmetric structures, it causes a multi-modality issue for position distribution. In this case, a particle filter is preferred to a Kalman filter, since it can sample distributions of any form. In a particle filter, each particle represent a possible position of the pedestrian, and is associated with a weight representing the possibility of the position. The particle filter is following the Bayesian framework, in which each particle has a prediction and a update steps. The weight is updated according to the likelihood to the real measurement, and if a particle is across a wall or in an impossible area, its weight is typically set to 0. Using the particles as samples of a distribution, we can get the mean and deviation of the position estimate.

Oliver Woodman and Robert Harle [72] implement an indoor localization system with foot-mounted inertial sensors and the floor plan of the interested building. Based on the accelerometer and gyroscope and using the ZUPT algorithm, foot-mounted inertial device works as a dead reckoning module, which report the step length, the change in height and the change in heading for each step. The building floor plan is represented as a collection of planar floor polygons. Each polygon corresponds to a surface on which a pedestrian's foot may be grounded. A particle filter is used for combining the inertial device outputs and the map polygon constraints. In the prediction phase, the position is estimated by the previous position and the output of the inertial device, and the polygon the pedestrian's foot is on is also predicted. In the update phase, two corrections are made: first, all particles which cross walls are eliminated (set the weight to 0); second the new weight for each particle is based on the difference between the reported height change from the inertial device and height change between the start and end polygons during this step. the Kullback-Leibler divergence (KLD) sampling method [231] is used for the re-sampling strategy. An experiment is taken in a 8725 m^2 building with stairs. The result shows that the localization error is within 0.73m 95% of the time.

Stéphane Beauregard et al. [232] propose a similar pedestrian map-based localization system. The novelty of this work is that it uses the Backtracking Particle Filter (BPF) for data fusion, and is evaluated with maps of different level details. The experiment results show that BPF based localization performs best. Backtracking Particle Filter et al. [73] presents a localization system based on angular probability density function (PDF) which is derived from a diffusion algorithm. The idea is that the inertial devices are usually good at estimating the step length, but not the heading, and thus it would be more effective if we focus on heading estimation instead of distance estimation. The idea of the diffusion algorithm is to have a source continuously effusing gas that disperses in free space and which gets absorbed by walls and other obstacles. Therefore, by treating the current position as the source and weighting more to those particles following the gas path, the heading distribution can be effectively captured. The experiment result shows that the localization error with angular PDFs is much smaller than that without the angular PDFs.

2.3.4 Localization with Inertial Devices, Infrastructure and Map

The more position related information a system obtains, the more constraints it has, and the more accurate estimation it may output. When combining inertial devices, infrastructure and maps, an indoor localization system usually has better performance (w.r.t accuracy and reliability) than those which only use a part of them. But in contrast, it has higher requirements on the inputs of the system, and some of the inputs may not be available in some scenarios.

Henri NURMINEN et al. [70] present an indoor hybrid localization system which relies on inertial sensors embedded in a smart phone, a WiFi indoor localization subsystem, and the building floor plan. Since the smart phone is hand held, ZUPT algorithm is not reliable. It uses a simple step detection algorithm, and leaves the step length and heading estimation to the higher layer particle filter. In addition, a barometer is used to detect the floor change using the particle filter. For the WiFi subsystem, the system uses a RSS based approach, but implements it with a high variance since the RSS measurements are not reliable in indoor environments. The map information is included to constrain the trajectory of the particle filter. But the difference is that the particle filter allows particles to penetrate a wall for reliability and coverage (by assigning these particles with low weights instead of deleting them). Furthermore, the system allows initialization after detecting divergence based on a lightweight fallback filter running in the background, and has the capability of smoothing the historical trajectory. The experimental results show that it achieves a mean error of 2 meters for real time localization, and 1.4 meters for the smoothing method.

Muhammad Irshan Khan et al. [71] implement a hybrid localization system with inertial sensors, WiFi infrastructure, and maps. The novelty of this paper is that it compares the performances of two different representations of maps used by the particle filter. One is to use a Computer Aided Drawing (CAD) format floor plan, in which the particles are eliminated when they are crossing the walls. The other models the map as nodes and links, which constrains the movements of particles on possible links. The experimental results show that both methods have consistent localization error of less than 3 meters. But the node link particle filter based method outperforms the wall crossing particle filter by approximately 1 meter in the majority of the testing case.

Chapter 3

Stationary Wireless Sensor Network Localization

This chapter presents our novel range-based localization approach for stationary wireless sensor networks. In order to make the approach as generic as possible and also have a deep insights, only the algorithm is focused on, and this algorithm is independent of the underlying ranging technique it relies on. According to the resource constrained features of WSN and the scenarios it applies to, we summarize the most important requirement goals as below:

- Independent of measurement types: fulfillment of this requirement makes the localization approach not only generic, but also accurate, since fusing heterogeneous positioning sources is usually more reliable and more accurate than only relying on one source. However, there are many different types of measurements, such as relative range, relative angle or absolute position; and from the functional relationship view, some are in unary form (like the absolute position measurement for one node), some are in binary form (like the relative position measurement between two nodes), and others are even in quad form (like the measurements involving four nodes in this work [121]). Representing and manipulating them in a uniform way is challenging.

- Quantitative uncertainty based estimation: measurements always come with noise, and estimates always have uncertainties. Modeling the measurement uncertainty makes the estimation more accurate, and representing the estimates with uncertainty provides more meaningful information for many applications, such as guiding how to select an anchor node, searching for a particular node, and working as criteria to evaluate the performance of a localization system. However, how to model, manipulate and use the uncertainty is not well formalized in previous works.

- Efficiency: the efficiency is defined in terms of memory usage, computational complexity and communication overhead. Since the tiny sensor nodes are usually resource constrained, the efficiency of a localization approach determines if it can localize n-odes in real time, and how well it can scale in terms of the number of nodes and measurements.

- Centralized and decentralized architecture support: a centralized approach has a glob-al view of the whole network, and thus usually has an optimal solution. But in some scenarios, a centralized architecture is not allowed. In addition, a decentralized approach may have the benefits of simplicity and reliability in some cases. Which architecture we should use highly depends on the application. Therefore, our localization scheme should provide both centralized and decentralized approaches.

The assumptions for this approach are as below:

- sensor nodes in the network can communicate with each other within a limited range, and there may or may not be a centralized computation center (or cluster) that each node can route packets to.

- Each node equips one or more positioning devices, and the types of the measurement it can take could be heterogeneous. For example, nodes may be able to measure the relative range between each other by using an RSSI method or acoustic based TOF method, and some nodes may even have GPS.

- The positioning devices are pre-calibrated, which means that their output measurements are unbiased, and the measurement uncertainty (represented as variance or covariance matrix) is known. In addition, measurements are independent with each other.

- Some nodes' positions are already known precisely or with some uncertainty in advance, either through manual inputs or GPS readings. These nodes are called anchor nodes.

3.1 System Formalization

In this thesis, each node's location is represented in Cartesian coordinate form. If not explicitly specified, the location is considered in a 2D plane, but it can be easily extended to 3D space. Assume there are totally n nodes in a network, Node i's 2D location is $\mathbf{x}_i = [x_i, y_i]^T$, $i = 1, 2, \cdots, n$. There are four basic measurements types in most of the localization systems, and they are modeled as below.

- GPS-like measurement, where $\mathbf{c}_i$ is the noisy GPS measurement and β_i is the measurement noise.

$$\mathbf{c}_i = \mathbf{x}_i + \beta_i \tag{3.1}$$

- Relative position measurement, where $\mathbf{q}_{ij}$ is the noisy relative position measurement between Nodes i and j, and ϵ_{ij} is the measurement noise.

$$\mathbf{q}_{ij} = \mathbf{x}_i - \mathbf{x}_j + \epsilon_{ij} \tag{3.2}$$

- Relative range measurement, where $\mathbf{r}_{ij}$ is the noisy relative range measurement between Nodes i and j, and γ_{ij} is the measurement noise.

$$\mathbf{r}_{ij} = ||\mathbf{x}_i - \mathbf{x}_j|| + \gamma_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} + \gamma_{ij} \tag{3.3}$$

- Relative angle measurement, where $\mathbf{a}_{ij}$ is the noisy relative angle measurement between Nodes i and j in global coordinates, and α_{ij} is the measurement noise.

$$\mathbf{a}_{ij} = atan2(y_i - y_j, x_i - x_j) + \alpha_{ij} \tag{3.4}$$

All the above measurements can be written in a uniform way as Equation 3.5, where $\mathbf{z}_{ij}$ is the noisy measurement, $h(\cdot)$ is the measurement function, $\mathbf{x}$ is the vector which stacks the coordinates of all nodes involved in this measurement, and ω_{ij} is the random noise of the measurement. This form represents the measurements with any number of arguments. For example, in a GPS-like measurement, $\mathbf{x} = \mathbf{x}_i$, and in a binary form measurement like relative position measurement, $\mathbf{x} = [\mathbf{x}_i^T, \mathbf{x}_j^T]^T$. Also note that we assume the uncertainty of the noise ω_{ij} is known and denoted as $\mathbf{P}_{ij}$.

$$\mathbf{z}_{ij} = h(\mathbf{x}) + \omega_{ij} \tag{3.5}$$

The measurement functions of the above types and their first derivatives are summarized in Table 3.1, where the symbol $\hat{x}$ represents the estimation of the variable x, and $h'(\cdot)_{|\hat{\mathbf{x}}}$ represents the derivative of function $h(\cdot)$ at the point $\hat{\mathbf{x}}$.

Table 3.1: The measurement functions and their first derivatives

Type	Function $h(\cdot)$	$\mathbf{H} = h'(\cdot)_{\vert\hat{\mathbf{x}}} = [\mathbf{h}, -\mathbf{h}]$
GPS	$\mathbf{x}_i + \omega_i$	$\mathbf{H} = \mathbf{h} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$
Relative Position	$\mathbf{x}_i - \mathbf{x}_j + \omega_{ij}$	$\mathbf{h} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$
Range	$\vert\vert\mathbf{x}_i - \mathbf{x}_j\vert\vert + \omega_{ij}$	$\mathbf{h} = \frac{(\hat{\mathbf{x}}_i - \hat{\mathbf{x}}_j)^T}{h(\hat{\mathbf{x}}_i, \hat{\mathbf{x}}_j)}$
Angle	$\arctan \frac{y_i - y_j}{x_i - x_j} + \omega_{ij}$	$\mathbf{h} = \left[\frac{\hat{y}_j - \hat{y}_i}{\vert\vert\hat{\mathbf{x}}_i - \hat{\mathbf{x}}_j\vert\vert^2}, \frac{\hat{x}_i - \hat{x}_j}{\vert\vert\hat{\mathbf{x}}_i - \hat{\mathbf{x}}_j\vert\vert^2} \right]$

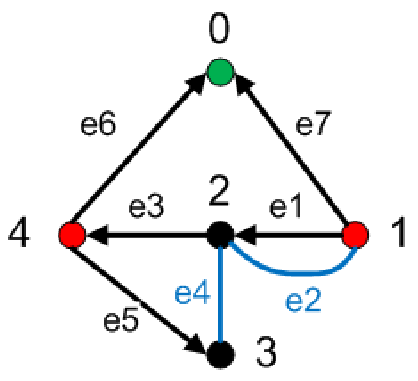


Figure 3.1: Measurement Graph. The green Node 0 is the global origin, red Nodes 1 and 4 are anchor nodes, and black Nodes 2 and 3 are the nodes to be localized. The blue edges e2 and e4 represent the range measurements. The other black edges represent relative position measurements or GPS measurements (e6 and e7)

.

3.1.1 Measurement Graph

A measurement graph is defined as $G = (V, E, F)$, where V is the vertex set including all nodes plus the global origin in the network, E is the edge set representing the measurements between nodes, and F is the mapping $E \to (Z, P)$, where Z is the set of all the measurements and P is the set of the error (noise) covariance matrices of all these measurements. There are two types of measurements (edges). One is a **strong measurement**, which is defined as a measurement which precisely determines a node's location if the measurement is accurate (without noise) and connects this node to another node whose location is precisely known in advance (an anchor) or can be precisely estimated. GPS type measurements and relative position type measurements are strong measurements. Another type is **weak measurement**, which is defined as a measurement which cannot precisely determine a node's location, but can only localize the node in some area even if this measurement is accurate and connects this node to another node whose location can be precisely determined. Relative range and relative angle measurements are weak measurements. In addition, edges are **directional** if $\mathbf{z}_{ij} \neq \mathbf{z}_{ji}$, and **undirectional** if $\mathbf{z}_{ij} = \mathbf{z}_{ji}$. For example, GPS, relative position and angle measurements are directional, but range measurements are undirectional.

An example of a measurement graph is shown in Figure 3.1. The green node (Node

0) is the global origin. The red nodes (Nodes 1 and 4) are anchor nodes, since they have relative position measurements to Node 0 (e6 and e7). The black nodes (Node 2 and 3) are the nodes to be localized. Edges e6 and e7 are GPS measurements, e2 and e4 are relative range measurements, and e1, e3 and e5 are relative position measurements. Edges e2 and e4 are directional, and the other edges are undirectional. Assuming the measurement corresponding to e2 is known as $(\mathbf{z}_{e2}, \mathbf{P}_{e2})$, where $\mathbf{z}_{e2} = [-1, 0]^T$ and $\mathbf{P}_{e2} = \begin{bmatrix} 0.2 & 0 \\ 0 & 0.1 \end{bmatrix}$, then $f(e2) = (\mathbf{z}_{e2}, \mathbf{P}_{e2})$.

Using the measurement graph, the localization problem can be stated as **given a measurement graph $G = (V, E, F)$, estimate the location coordinate $\hat{\mathbf{x}}_i$ and the corresponding estimation error covariance matrix $\mathbf{P}_i$ for each node $v_i \in V$.**

3.2 Electrical Network Analogy

This section explains how to solve the localization problem based on the measurement graph model. We start with a linear case where the measurements are all relative position or GPS measurements, and then extend to a nonlinear case in which the measurements can be mixed with relative position, GPS, range and relative angles. For the linear case, we first show an existing solution which uses the Best Linear Unbiased Estimation (BLUE) algorithm along with its shortcoming. Then, by using an electrical network analogy, we propose a novel incremental algorithm which has less computational complexity in dynamic cases (where measurements keep being generated or nodes join or leave the network at run time). After that, this method is extended to the nonlinear case. Finally, the simulation results are shown.

3.2.1 Linear Case–BLUE Algorithm

Estimating each node's vector-valued variable based on a set of noisy linear measurements, such as problems of time synchronization [233], 2D or 3D WSN localization [69], or motion consensus [69], can be solved by the Best Linear Unbiased (BLUE) estimator based on the measurement graph model.

Assume there are totally n nodes in the network, and the vector-valued variable (location coordinate) for each node is $\mathbf{x}_i \in \mathbb{R}^d, i = 1, 2, \cdots, n$. A set of independent linear measurements about these variables are obtained, denoted as $\zeta_{uv} = \mathbf{x}_u - \mathbf{x}_v + \epsilon_{uv} \in \mathbb{R}^d, u, v \in \{1, 2, \cdots, n\}$, where $\epsilon_{uv} \in \mathbb{R}^d$ is the random error vector with zero mean and a covariance matrix $\mathbf{P}_{uv} = E[\epsilon_{uv}\epsilon_{uv}^T] \in \mathbb{R}^{d\times d}$, representing the measurement uncertainty. By stacking all the node variables into one vector $\mathbf{x} = [\mathbf{x}_1^T, \mathbf{x}_2^T, \cdots, \mathbf{x}_n^T]^T \in \mathbb{R}^{nd}$, all the measurements into one vector $\mathbf{z} = [\zeta_1^T, \zeta_2^T, \cdots, \zeta_m^T]^T \in \mathbb{R}^{md}$ (assuming a total of m measurements) and all the measurement errors into a vector $\epsilon = [\epsilon_1^T, \epsilon_2^T, \cdots, \epsilon_m^T]^T \in \mathbb{R}^{md}$, the measurement equations can be expressed as

$$\mathbf{z} = \mathcal{A}^T\mathbf{x} + \epsilon \tag{3.6}$$

where $\mathcal{A} = \mathbf{A} \otimes \mathbf{I}_d$, $\mathbf{A}$ is the incidence matrix of the measurement graph G, $\mathbf{I}_d$ is the $d \times d$ identity matrix, and $\otimes$ denotes the Kronecker product.

Since we only consider linear measurements, i.e. relative position measurements and GPS measurements, the measurement graph $G = (V, E, F)$ is a directed graph for $\zeta_{uv} \neq \zeta_{vu}$. The incidence matrix $\mathbf{A}$ of measurement graph G is a $n \times m$ matrix, where one row corresponds to one node, one column corresponds to one edge. The element a_{ue} of A is 1 if Edge e leaves from Node u, -1 if e is directed toward u, and 0 if Node u is not involved in Edge e.

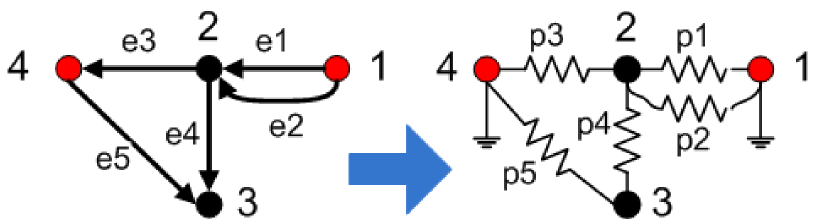


Figure 3.2: The Analogy Between a measurement graph and the generalized electrical network. The red nodes are anchor nodes.

.

For example, the incident matrix $\mathbf{A}$ of the left figure in Figure 3.2 is

$$\mathbf{A} = \begin{bmatrix} 1 & -1 & 0 & 0 \\ 1 & -1 & 0 & 0 \\ 0 & 1 & 0 & -1 \\ 0 & 1 & -1 & 0 \\ 0 & 0 & -1 & 1 \end{bmatrix}$$

The Equation 3.6 is expressed as below in this case:

$$\underbrace{\begin{bmatrix} \zeta_1 \\ \zeta_2 \\ \zeta_3 \\ \zeta_4 \\ \zeta_5 \end{bmatrix}}_{\mathbf{z}} = \underbrace{\begin{bmatrix} I & -I & 0 & 0 \\ I & -I & 0 & 0 \\ 0 & I & 0 & -I \\ 0 & I & -I & 0 \\ 0 & 0 & -I & I \end{bmatrix}}_{\mathcal{A}^T} \underbrace{\begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \\ \mathbf{x}_4 \end{bmatrix}}_{\mathbf{x}} + \underbrace{\begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \\ \epsilon_4 \\ \epsilon_5 \end{bmatrix}}_{\epsilon}$$

Partitioning $\mathbf{x}$ into an anchor node variable vector $\mathbf{x}_r$ and a non-anchor node variable vector $\mathbf{x}_b$, the matrix $\mathcal{A}$ can also be partitioned to an anchor node sub matrix $\mathcal{A}_r$ and a non-anchor node sub matrix $\mathcal{A}_b$. Then Equation 3.6 is rewritten as

$$\mathbf{z} = \mathcal{A}_b^T \mathbf{x}_b + \mathcal{A}_r^T \mathbf{x}_r + \epsilon \quad (3.7)$$

$$\Rightarrow \quad \bar{\mathbf{z}} = \mathcal{A}_b^T \mathbf{x}_b + \epsilon \quad (3.8)$$

where $\overline{\mathbf{z}} = \mathbf{z} - \mathcal{A}_r^T \mathbf{x}_r$. For the example in Figure 3.2, since Nodes 1 and 4 are anchors, the equation is written as

$$\underbrace{\begin{bmatrix} \zeta_1 \\ \zeta_2 \\ \zeta_3 \\ \zeta_4 \\ \zeta_5 \end{bmatrix}}_{\mathbf{z}} = \underbrace{\begin{bmatrix} -I & 0 \\ -I & 0 \\ I & 0 \\ I & -I \\ 0 & -I \end{bmatrix}}_{\mathcal{A}_b^T} \underbrace{\begin{bmatrix} \mathbf{x}_2 \\ \mathbf{x}_3 \end{bmatrix}}_{\mathbf{x}_b} + \underbrace{\begin{bmatrix} I & 0 \\ I & 0 \\ 0 & -I \\ 0 & 0 \\ 0 & I \end{bmatrix}}_{\mathcal{A}_r^T} \underbrace{\begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_4 \end{bmatrix}}_{\mathbf{x}_r} + \underbrace{\begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \\ \epsilon_4 \\ \epsilon_5 \end{bmatrix}}_{\epsilon}$$

$$\Rightarrow \underbrace{\begin{bmatrix} \zeta_1 - x_1 \\ \zeta_2 - x_1 \\ \zeta_3 + x_4 \\ \zeta_4 \\ \zeta_5 - x_4 \end{bmatrix}}_{\overline{\mathbf{z}}} = \underbrace{\begin{bmatrix} -I & 0 \\ -I & 0 \\ I & 0 \\ I & -I \\ 0 & -I \end{bmatrix}}_{\mathcal{A}_b^T} \underbrace{\begin{bmatrix} \mathbf{x}_2 \\ \mathbf{x}_3 \end{bmatrix}}_{\mathbf{x}_b} + \underbrace{\begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \\ \epsilon_4 \\ \epsilon_5 \end{bmatrix}}_{\epsilon}$$

Obtaining an estimate of $\mathbf{x}_b$ becomes a classical estimation problem, which can be solved by BLUE. If G is a weakly connected graph (where there always exist undirected paths between any node pair), the best estimate for $\mathbf{x}_b$ in the linear combination space of all the measurements is uniquely determined by $\hat{\mathbf{x}}_b$, i.e. the solution of the following linear system

$$\mathcal{L}\hat{\mathbf{x}}_b = \mathbf{b} \tag{3.9}$$

$$\Rightarrow \quad \hat{\mathbf{x}}_b = \mathbf{b}\mathcal{L}^{-1} \tag{3.10}$$

where $\mathcal{L} = \mathcal{A}_b \mathcal{P}^{-1} \mathcal{A}_b^T$ (called *G's Kirchhoff Matrix*), $\mathbf{b} = \mathcal{A}_b \mathcal{P}^{-1} \overline{\mathbf{z}}$, and $\mathcal{P}$ is a block diagonal matrix consisting of all the covariance matrices $\mathbf{P}_i$ of $\epsilon_i, i = 1, 2, \cdots, m$. The covariance matrix of the estimation errors is given by

$$\mathbf{\Sigma} = \mathcal{L}^{-1} \tag{3.11}$$

Although the above centralized solution is elegant, it has several drawbacks. First, the time complexity is high. Assuming there are n nodes and m measurements, the anchor number is constant, and the dimension of the variable vector is d, the computational complexity for the covariance matrix $\mathbf{\Sigma}$ is $T(\mathbf{\Sigma}) = O(n^2md^3)$, since $T(\mathcal{L}) = T(\mathcal{P}^{-1}) + T(\mathcal{A}_b\mathcal{P}^{-1}) + T((\mathcal{A}_b\mathcal{P}^{-1})\mathcal{A}_b^T) = md^3 + nmd^2 + n^2md^3 = O(mn^2d^3)$, and $T(\mathcal{L}^{-1}) = n^3d^3$ $(n <= m)$. The computational complexity for estimating nodes' location is $T(\hat{\mathbf{x}}_b) = O(n^2md^3)$, since $T(\hat{\mathbf{x}}_b) = T(\mathcal{L}^{-1}) + T(\mathbf{b}) + T(\mathcal{L}^{-1}\mathbf{b}) = (md^3 + nmd^2 + n^2md^3 + n^3d^3) + nmd^2 + n^2d^2 = O(n^2md^3)$. So the time complexity for calculating both $\hat{\mathbf{x}}$ and $\mathbf{\Sigma}$ simultaneously is $T(\hat{\mathbf{x}}_b, \mathbf{\Sigma}) = T(\hat{\mathbf{x}}_b) = O(n^2md^3)$. The computational complexity depending on the measurement number m makes this solution not scale well, since m would be a very large number as the node number increases (the possible different edge number of a graph is of the order of $O(n^2)$, and the same edge may be measured multiple times in practice).

Second, this method follows a one-time calculation scheme, which means it calculates $\hat{\mathbf{x}}_b$ and $\mathbf{\Sigma}$ for a fixed measurement graph, if the measurement graph changes, all the calculations must be re-executed. However, in practice, it is very likely that new measurements are continuously generated, and sensor nodes may leave or join the network dynamically. Therefore, if the network size is large, the high one-time calculation time complexity would prevent it from providing the latest estimation in real time.

3.2.2 Linear Case–INOVA Algorithm

To overcome the drawbacks of high computational complexity and one-time calculation scheme of the above method, we propose a new computation scheme, called the incremental node-voltage analysis (INOVA) method, which derives the same results as BLUE, but changes the computational complexity from $O(mn^2d^3)$ to $O(n^3d^3)$, which is crucial in practice since usually $n \ll m$. Moreover, INOVA follows an incremental computation scheme, which incrementally calculates $\hat{\mathbf{x}}_b$ and $\mathbf{\Sigma}$ in an inexpensive way as new measurements are generated, or nodes join or leave the network dynamically. Although this method can be derived by analyzing the structure of matrices in Equation 3.9, we choose to deduce it based on the electrical network analogy for the following two reasons: 1) it provides more insights of the decomposition operations; and 2) our subsequent distributed algorithm in Section 3.3 is based on this analogy.

3.2.2.1 The Analogy Between A Measurement Graph and A Generalized Electrical Network

It has been shown in [80] that a measurement graph $G = (V, E, F)$ is analogous to a generalized electrical network $\mathcal{G} = (V, \mathcal{E}, \mathcal{F})$, where V is the node set the same as in G, $\mathcal{E}$ is the edge set the same as E, but without directions, and $\mathcal{F} : \mathcal{E} \rightarrow \mathcal{R}$ is an edge function that assigns each edge e_i a matrix valued resistance $\mathbf{R}_i$ which is numerically equal to the measurement error covariance matrix $\mathbf{P}_i$ in the measurement graph.

A generalized current $\mathbf{J}$ is defined as a $d \times d$ matrix associated with an edge with certain direction, and satisfies the Kirchhoff's Current Law, i.e. for each node in a generalized electrical network, the net generalized current flowing out or into that node is $\mathbf{0}$. A generalized voltage potential difference $\mathbf{U}_{uv}$ between two nodes u and v is defined as a $d \times d$ matrix $\mathbf{U}_{uv} = \mathbf{R}_{uv} \times \mathbf{J}_{uv}$. Similarly, it satisfies Kirchhoff's Voltage Law, i.e. for any loop in a generalized electrical network, the sum of the generalized voltage potential difference in the clockwise or the counterclockwise direction is $\mathbf{0}$. Like the traditional electrical network, each node is associated with a voltage potential if there are currents or voltages imposed to the generalized electrical network. For the anchor nodes in $\mathcal{G}$, they are considered being connected to the ground without a resistance (if the anchor's location is known without uncertainty), and thus their voltage potentials are always $\mathbf{0}$.

One important conclusion from work [80] is that each node's estimation covariance matrix $\mathbf{\Sigma}_{ii}$ (the ith diagonal block element of network's covariance matrix $\mathbf{\Sigma}$) is numerically equal to the effective resistance $\mathbf{R}_i^{eff}$ between that node and the ground in $\mathcal{G}$. Based on this conclusion, we find interesting calculation patterns and an analogy as follows, which is the essence of INOVA and the decentralized algorithm in Section 3.3.

To calculate Σ_{ii}, we can impose an identity generalized current $\mathbf{I}$ to that node, then Node i's voltage potential $\mathbf{U}_{ii}$ is numerically equal to $\mathbf{R}_i^{eff}$, i.e. $\mathbf{U}_{ii} = \mathbf{R}_i^{eff} \times \mathbf{I} = \mathbf{R}_i^{eff} = \mathbf{\Sigma}_{ii}$. We define the generalized electrical network $\mathcal{G}$ imposed by an identity current $\mathbf{I}$ on Node i as **Node i's Identity Current Graph, denoted as $\mathcal{G}^i$.**

Therefore, to calculate $\mathbf{\Sigma}_{ii}, i = 1, 2, \cdots, n_b$ for each node, it is equivalent to calculate its voltage potential $\mathbf{U}_{ii}$ in its Identity Current Graph $\mathcal{G}^i$. The node-voltage analysis technique from electrical theory can be borrowed and extended: set n_b voltage potential variables $\mathbf{U}_i = [U_{i1}^T, U_{i2}^T, \cdots, \mathbf{U}_{in_b}^T]^T$ for every non-anchor node in $\mathcal{G}^i$, and then build the current balance equation for each node based on Kirchhoff's Current Law, i.e. the sum of the currents for any node is $\mathbf{0}$. The formal equation is as below.

$$\underset{dn_b \times dn_b}{\mathbf{L}} \underset{dn_b \times d}{\mathbf{U}_i} = \underset{dn_b \times d}{\mathbf{H}_i} \tag{3.12}$$

where $\mathbf{L} = [\mathbf{L}_{uv}] \in \mathbb{R}^{dn_b \times dn_b}, u, v \in \{1, 2, \cdots, n_b\}$,

$$\underset{d \times d}{\mathbf{L}_{uv}} = \begin{cases} \sum\limits_{\{u,q\} \in \mathcal{E}} \mathbf{P}_{uq}^{-1} \text{ (or } \mathbf{P}_{qu}^{-1}) & \text{if } u = v \\ -\sum \mathbf{P}_{uv}^{-1} (\text{or } \mathbf{P}_{vu}^{-1}) & \text{if } u \neq v \wedge \{u, v\} \in \mathcal{E} \\ \mathbf{0} & \text{if } \{u, v\} \notin \mathcal{E} \end{cases} \tag{3.13}$$

$$\underset{dn_b \times d}{\mathbf{H}_i} = [\underset{d \times d}{\mathbf{0}}, \underset{d \times d}{\mathbf{0}}, \cdots, \underset{d \times d}{\mathbf{I}}, \cdots, \underset{d \times d}{\mathbf{0}}]^T$$

It is not hard to show that $\mathbf{L}$ in Equation 3.12 is equal to $\mathcal{L}$ in Equation 3.9. Equation 3.12 is for one node. If we consider all nodes' $\mathbf{\Sigma}_{ii}, i = 1, 2, \cdots, n_b$ together, we have the following equations:

$$\begin{aligned} \underset{dn_b \times dn_b}{\mathbf{L}} \underset{dn_b \times dn_b}{\mathbf{U}} &= \underset{dn_b \times dn_b}{\mathbf{H}} \\ \Rightarrow \underset{dn_b \times dn_b}{\mathbf{L}} \underset{dn_b \times dn_b}{\mathbf{U}} &= \underset{dn_b \times dn_b}{\mathbf{I}} \end{aligned} \tag{3.14}$$

where $\mathbf{U} = [\mathbf{U}_1, \mathbf{U}_2, \cdots, \mathbf{U}_{n_b}]$ and $\mathbf{H} = [\mathbf{H}_1, \mathbf{H}_2, \cdots, \mathbf{H}_{n_b}] = \mathbf{I}$ (size $dn_b \times dn_b$). The solution is $\mathbf{U} = \mathbf{L}^{-1} = \mathbf{\Sigma}$.

Another observation by relating the structure of matrix $\mathbf{U}$ to the node's Identity Current Graph is that **the covariance matrix $\mathbf{\Sigma}_{ij}$ between Nodes i and j is numerically equal to the voltage potential $\mathbf{U}_{ij}$ of Node i in Node j's Identity Current Graph $\mathcal{G}^j$, which is also equal to the voltage potential $\mathbf{U}_{ji}$ of Node j in Node i's Current Graph $\mathcal{G}^i$ (since $\mathbf{\Sigma}$ is a symmetric matrix)**.

Moreover, $\mathbf{b} = [\mathbf{b}_u]$ in Equation 3.9 where Node u is a non-anchor node can be directly built as follows.

$$\begin{aligned} \mathbf{b}_u &= \sum_{(u,v)\in E} \mathbf{P}_{uv}^{-1}[\zeta_{uv} + \mathbf{c}_v] \\ \mathbf{c}_v &= \begin{cases} \mathbf{x}_v & \text{if Node } v \text{ is an anchor node} \\ \underset{d\times 1}{\mathbf{0}} & \text{otherwise} \end{cases} \end{aligned} \tag{3.15}$$

Comparing this analogy method with BLUE, it reduces the complexity for constructing the matrix $\mathbf{L}$ from $O(mn^2d^3)$ to $O(md^3)$, and the matrix $\mathbf{b}$ from $O(nmd^2)$ to $O(md^2)$, where n is the total node number, m is the total measurement number, and d is the dimension (2D plane or 3D space). After building the matrices $\mathbf{L}$ and $\mathbf{b}$, the computational complexity for $\mathbf{\Sigma}$ and $\hat{\mathbf{x}}_b$ are $O(n^3d^3)$ and $O(n^2d^2)$, respectively, which is $O(mn^2d^3)$ in BLUE. Since in practice, $n \ll m$, the analogy method also outperforms BLUE.

3.2.2.2 The INOVA Algorithm

The above analogy method follows a one time calculation scheme, which means it works well when the network is fixed (the number of nodes and measurements are fixed). However, in practice, many scenarios allow nodes to leave and join, and new measurements to be generated over time. Do we need to redo all the above processes from scratch when these dynamic changes happen?

There are two types of dynamic changes for a WSN: new measurements being generated and nodes quitting or joining the network. For the former problem, when a new measurement ζ_{uv} is generated, only two Nodes u and v are involved. So we only need to modify the balance equations for these two nodes if they are not anchor nodes, i.e. modifying four elements in $\mathbf{L}$ and two elements in $\mathbf{b}$ as follows (if any of them is an anchor, nothing needs to be done for that node).

Operation 1. *When a new measurement ζ_{uv} is generated, if u and v are both non-anchor nodes, update*

$$\begin{bmatrix} \mathbf{L}_{uu} & \mathbf{L}_{uv} \\ \mathbf{L}_{vu} & \mathbf{L}_{vv} \end{bmatrix} = \begin{bmatrix} \mathbf{L}_{uu} & \mathbf{L}_{uv} \\ \mathbf{L}_{vu} & \mathbf{L}_{vv} \end{bmatrix} + \begin{bmatrix} \mathbf{P}_{uv}^{-1} & -\mathbf{P}_{uv}^{-1} \\ -\mathbf{P}_{vu}^{-1} & \mathbf{P}_{vu}^{-1} \end{bmatrix} \tag{3.16}$$

$$\begin{bmatrix} \mathbf{b}_{u} \\ \mathbf{b}_{v} \end{bmatrix} = \begin{bmatrix} \mathbf{b}_{u} \\ \mathbf{b}_{v} \end{bmatrix} + \begin{bmatrix} \mathbf{P}_{uv}^{-1}\zeta_{uv} \\ \mathbf{P}_{vu}^{-1}\zeta_{vu} \end{bmatrix} \tag{3.17}$$

If Node u is an anchor node, update

$$\mathbf{L}_{vv} = \mathbf{L}_{vv} + \mathbf{P}_{vu}^{-1} \tag{3.18}$$

$$\mathbf{b}_{v} = \mathbf{b}_{v} + \mathbf{P}_{vu}^{-1}(\zeta_{vu} + \mathbf{x}_u) \tag{3.19}$$

For the event of a node leaving or joining the network, it only impacts that node and its neighbors. Especially for node joining, the new node should have measurements with the existing nodes in order to keep the graph weakly connected, otherwise, our system would not include this node. The corresponding operations are:

Operation 2. *If a node joins the network with some measurements to some already existing nodes, update*

$$\mathbf{L} = \left[\begin{array}{c|c} \mathbf{L} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{0} \end{array}\right], \mathbf{b} = \left[\begin{array}{c} \mathbf{b} \\ \hline \mathbf{0} \end{array}\right] \tag{3.20}$$

Then perform Operation 1 for each of the newly introduced measurements.

Operation 3. *If Node u leaves the network, for each neighbor q of Node u which connects u with the measurement ζ_{qu}, do $\mathbf{L}_{qq} = \mathbf{L}_{qq} - \mathbf{P}_{qu}^{-1}$, $\mathbf{b}_q = \mathbf{b}_q - \mathbf{P}_{qu}^{-1}\zeta_{qu}$. After updating all its neighbors, delete Row u and Column u in $\mathbf{L}$, and the block element $\mathbf{b}_u$.*

In summary, the initial inputs of INOVA are the measurement set Z, the corresponding error covariance matrix set P and the anchor nodes' location set X_r. The outputs are nodes' location estimates $\hat{\mathbf{x}}_b$ and the covariance matrix $\mathbf{\Sigma}$. After the initial localization, if there is any dynamic change at run time, such as new measurements are generated, new nodes join or existing nodes leave, INOVA incrementally updates the localization results. The whole algorithm is described as follows:

1. Calculate $\mathbf{P}_i^{-1}, i = 1, 2, \cdots, m$ $(O(md^3))$.
2. Scan all initial measurements, build up $\mathbf{L}$ and $\mathbf{b}$ by using Equations 3.13 and 3.15, respectively $(O(md^3)$, $O(md^2))$
3. Calculate $\mathbf{\Sigma} = \mathbf{L}^{-1}$ $(O(n^3d^3))$.
4. Calculate $\hat{\mathbf{x}}_b = \mathbf{L}^{-1}\mathbf{b}$ $(O(n^2d^2))$.
5. If a new measurement is generated, do Operation 1 $(O(d^3))$; if a new node joins the network with constant number of measurements, do Operation 2 $(O(d^3))$; if a node having constant number of neighbors leaves the network, do Operation 3 $(O(d^3))$.

Then repeat Steps 3 and 4 (totally $O(md^3)$) to update the location estimation as needed.

INOVA needs $O(md^3)$ computational complexity and $O(md^2)$ to build up the matrix $\mathbf{L}$ and the vector $\mathbf{b}$ for a fixed network, and $O(d^3)$ for each of the following dynamic changes. After that, it needs $O(n^3d^3)$ computational complexity to obtain the locations and the covariance matrix for all nodes. For some application, it may only need to update the location for some time period T. Then INOVA can run in background for every T period. Comparing with BLUE, the improvement of INOVA is significant for two reasons: first, in a large dense network, m is much larger than n, and thus $O(n^3d^3)$ is much smaller than $O(mn^2d^3)$. Second, if we look into the details of these two processes, we find the key difference is how to build $\mathbf{L}$. In BLUE, it always needs $O(mn^2d^3)$; while in INOVA, it needs $O(md^3)$ initially, but after that, when the network dynamically changes, the time is constant. Therefore, INOVA outperforms BLUE in both the fixed network case and the dynamic changing case. Especially in a dynamic scenario, INOVA dramatically improves the computational complexity (see Section 3.4).

3.2.3 Extension to The Nonlinear Case

Although the preceding section successfully solves the localization problem by using the electrical network analogy, it has two limitations. First, all the measurements are of linear type, specifically, in the form of $\zeta_{uv} = \mathbf{x}_u - \mathbf{x}_v + \epsilon_{uv}$. Second, it treats the anchor nodes differently which makes Formulas 3.15 and Operation 1 unable to unify the measurement updates for those involving anchor nodes and those not. Therefore, the goal of this section is to extend the linear case to any nonlinear case and unify the computation form for both anchor nodes and non-anchor nodes.

3.2.3.1 Nonlinear case

For any function $\mathbf{z} = h(\mathbf{x}) + \omega$, it can be linearized to $\tilde{\mathbf{z}} = h(\hat{\mathbf{x}}) + \mathbf{H}(\mathbf{x} - \hat{\mathbf{x}}) + \omega$, where $\tilde{\mathbf{z}}$ is the linearized approximation for $\mathbf{z}$, $\hat{\mathbf{x}}$ is the point where the linearization expands on, $\mathbf{H}$ represents the derivative of function $h(\cdot)$ at the point $\hat{\mathbf{x}}$. The $\mathbf{H}$'s for the four types of measurements are listed in Table 3.1.

According to the information filter [185], $\mathbf{L}$ and $\mathbf{b}$ should be updated as below:

$$\mathbf{L_z} = \mathbf{L_z} + \mathbf{H}^T\mathbf{P_z}^{-1}\mathbf{H} \tag{3.21}$$

$$\mathbf{b_z} = \mathbf{b_z} + \mathbf{H}^T\mathbf{P_z}^{-1}(\mathbf{z} - h(\hat{\mathbf{x}}) + \mathbf{H}\hat{\mathbf{x}}) \tag{3.22}$$

$\mathbf{L_z}$ represents the sub-matrix of $\mathbf{L}$ which contains the elements of involved nodes. The measurement could involve any number of nodes. Taking the binary form measurement for example, then $\mathbf{L_z} = \begin{bmatrix} \mathbf{L}_{uu} & \mathbf{L}_{uv} \\ \mathbf{L}_{vu} & \mathbf{L}_{vv} \end{bmatrix}$. This is similar for the vector $\mathbf{b_z}$. $\mathbf{P_z}$ is the error covariance matrix for the random variable ω. $\mathbf{H}$ is the first derivative of function $h(\cdot)$. $\hat{\mathbf{x}}$ is the point the derivative function evaluated on the latest position estimate.

For range or angle measurement cases, analogous to the electrical network, $\mathbf{h}^T\mathbf{p_z}^{-1}\mathbf{h}$ (where $\mathbf{H} = [\mathbf{h}, -\mathbf{h}]$ and $\mathbf{P_z} = \mathbf{p_z}$) is the inverse of the resistance $\mathbf{R}$. However, $\mathbf{h}^T\mathbf{p_z}^{-1}\mathbf{h}$ is a singular matrix, which does not have an inverse. Therefore, more generally speaking, we should use **conduction** (which is the inverse of the resistance in the electrical network) for the analogy.

Then, Equations 3.13 and 3.15 are modified as follows to support the nonlinear case.

$$\underset{d\times d}{\mathbf{L}_{uv}} = \begin{cases} \sum\limits_{\{u,q\}\in\mathcal{E}} \mathbf{h}_{uq}^T\mathbf{P}_{uq}^{-1}\mathbf{h}_{uq} & \text{if } u = v \\ -\sum \mathbf{h}_{uv}^T\mathbf{P}_{uv}^{-1}\mathbf{h}_{uv} & \text{if } u \neq v \wedge \{u,v\} \in \mathcal{E} \\ \mathbf{0} & \text{if } \{u,v\} \notin \mathcal{E} \end{cases} \tag{3.23}$$

where

$$\mathbf{h}_{uv} = h'_{uv}(\cdot)\Big|_{[\hat{\mathbf{x}}_u^T, \hat{\mathbf{x}}_v^T]^T}$$

$$\mathbf{b}_u = \sum_{(u,v)\in E} \mathbf{h}_{uv}^T \mathbf{P}_{uv}^{-1}(\mathbf{z}_{uv} - h([\hat{\mathbf{x}}_u^T, \hat{\mathbf{x}}_v^T]^T) + \mathbf{H}_{uv}[\hat{\mathbf{x}}_u^T, \hat{\mathbf{x}}_v^T]^T) \tag{3.24}$$

For a linear case, Formulas 3.21 and 3.22 degrade to Formulas 3.16 and 3.17, or Formulas 3.18 and 3.19. Therefore, Formulas 3.21 and 3.22 are valid for both linear and non linear cases. Moreover, another benefit of Formulas 3.21 and 3.22 is that it supports the measurement function involving any number of nodes.

Comparing the nonlinear case formulas 3.21 and 3.22 with the linear case formulas 3.16 and 3.17, the former needs the initial position estimates for those involved nodes. This means before fusing those nonlinear measurements, an initial localization is needed.

In Section 3.1.1, we defined the strong and weak measurements. The linear measurements such as GPS and relative position measurements belong to strong measurements, while the nonlinear measurements such as relative range and relative angle measurements belong to the weak measurement. Therefore, to initialize nodes' position, the system first localizes each nodes only based on the strong measurements. After that it fuses the weak measurements along with the initial position estimation to further refine the results.

Therefore, under the INOVA framework, **a node is localizable if it is reachable through a strong path (an undirectional path which only consists of strong measurements) starting from an anchor node in the measurement graph**. Note that this is a sufficient condition for localizability. Nodes which have three or more relative range measurements are still localizable through multilateration.

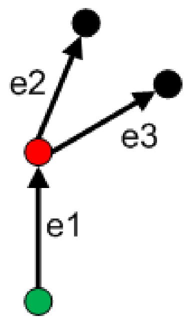
To unify the nonlinear and linear cases, the INOVA algorithm in Section 3.2.2.2 is modified as follows:

1. Calculate $\mathbf{P}_i^{-1}, i = 1, 2, \cdots, m$ ($O(md^3)$).
2. Scan all initial strong measurements, build up $\mathbf{L}$ and $\mathbf{b}$ by using Equations 3.23 and 3.24, respectively ($O(md^3)$, $O(md^2)$).
3. Calculate $\mathbf{\Sigma} = \mathbf{L}^{-1}$ ($O(n^3d^3)$) and $\hat{\mathbf{x}}_b = \mathbf{L}^{-1}\mathbf{b}$ ($O(n^2d^2)$) to get the initial position estimation.
4. Scan all the remaining weak measurements which only involve nodes having the initial estimated positions. Update $\mathbf{L}$ and $\mathbf{b}$ with Formulas 3.21 and 3.22, respectively ($O(md^3)$, $O(md^2)$).
5. If a new measurement is generated, do Operation 1 ($O(d^3)$) with Formulas 3.21 and 3.22 instead; if a new node joins the network with constant number of measurements, do Operation 2 ($O(d^3)$); if a node having constant number of neighbors leaves the network, do Operation 3 ($O(d^3)$).
6. If necessary (e.g. updating on each new measurement or batch-processing them over a time period T), localize nodes with the updated $\mathbf{L}$ and $\mathbf{b}$ using Formulas $\mathbf{\Sigma} = \mathbf{L}^{-1}$ ($O(n^3d^3)$) and $\hat{\mathbf{x}}_b = \mathbf{L}^{-1}\mathbf{b}$ ($O(n^2d^2)$).
7. Repeat steps 5 and 6.

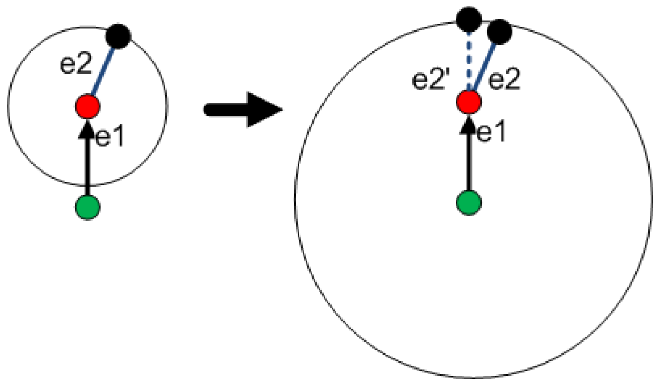
3.2.3.2 Unifying Anchor Nodes and Non-anchor Nodes

Another downside for the INOVA algorithm in Section 3.2.2.2 is that it treats the anchor nodes as special nodes in the electrical network, which directly connect to ground. This results in the need to handle the measurements which involve the anchor nodes separately.

To unify the update operation, one may consider cascading the anchor measurement with another measurement connecting to this anchor as one measurement. For example, if an anchor node's position is $\mathbf{x}_a = [3, 4]^T$ and another measurement is $\zeta_{ab} = [1, -2]^T$, then construct a new measurement $\zeta_{0b} = \zeta_{0a} + \zeta_{ab} = -[3, 4]^T + [1, -2]^T = [-2, -6]^T$ with the same covariance matrix as ζ_{ab}.



(a) Correlation problem



(b) Nonlinear measurement problem

Figure 3.3: The problems to combine an anchor measurement with a regular measurement.

However, this method has two problems. First, the two measurements which contain the same anchor are correlated. As Figure 3.3(a) shows, the new combined measurements $e2' = e1{+}e2$ and $e3' = e1{+}e3$ are correlated with each other, which results in non-optimality by using INOVA.

Second, if the regular measurement is a nonlinear measurement, for example, a range measurement, it will change the semantics of the measurement. Taking Figure 3.3(b) for instance, the original range measurement $e2$ means (assuming it is accurate) the node's position is in the circle which is centered at the anchor node with $e2$ as the radius. However, after combining $e1$ and $e2$, the meaning changes to the node's position is in the circle which is centered at the anchor node with radius $|e1| + e2$, which obviously has more uncertainty than the original one.

To eliminate the speciality of anchor nodes, the approach is very simple: **treating each anchor node as a regular node along with a GPS measurement with a very small uncertainty**. Then the whole system has only one anchor node, which is the origin node $[0, 0]^T$. The operations for anchor nodes and non-anchor nodes are unified.

3.3 Decentralized Algorithm

In some scenarios (e.g., in wild areas), the centralized architecture may not be available, and thus a distributed scheme is required. Prabir Barooah et al. proposed a distributed algorithm called Overlapping Subgraph Estimator (OSE) to estimate each node's location [80]. However, OSE only estimates node positions, but not the covariance matrix. In this section, we propose a distributed scheme called OSE-COV, which estimates not only each node's position, but also the corresponding error covariance matrix for all nodes, i.e. a distributed and complete version of the INOVA algorithm.

The essence of OSE is to use a method, called Asynchronous Filtered Weighted Additive Schwarz (AFWAS) [234], to solve linear Equation 3.9 for the whole network in a distributed way. In OSE, each node assumes its two-hop neighbor's estimations are correct, iteratively estimate its own location based on its two-hop measurement graph, and exchanges the estimates among neighbors. After a number of iterations, each node's estimate converges to the estimate calculated by BLUE.

However, to calculate the covariance matrix of all nodes' estimations is a little different, since the covariance matrix includes not only the variance of each node position estimation ($\mathbf{\Sigma}_{ii}$), but also the covariance between nodes' estimations ($\mathbf{\Sigma}_{ij}$). To solve this problem, besides using the AFWAS method, we also need to borrow the electrical network analogy.

In Section 3.2.2.1, we know that each measurement connects two nodes with some uncertainty. The uncertainty is represented as a covariance matrix (a linear measurement) or $\mathbf{h}^T\mathbf{P}_z^{-1}\mathbf{h}$ (a nonlinear measurement), corresponding to a **general resistance** or a **general conduction** in a general electrical network. We also define Node i's identity current graph $\mathcal{G}^i$ as the electrical network imposed by the identity current on Node i. Therefore, although the topologies of different nodes' identity current graph are the same, the current on each branch and the voltage potential of each node are different. We also observed that the block element $\mathbf{\Sigma}_{ij}$ of the covariance matrix is numerically equal to Node i's voltage potential $\mathbf{U}_{ij}$ in Node j's identity current graph $\mathcal{G}^j$. Since a covariance matrix is a symmetric matrix,

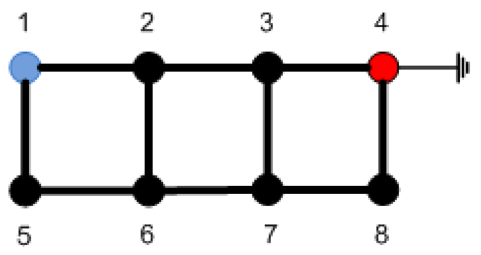


Figure 3.4: Node 1's identity current graph. Nodes 4 and 5 are anchors.
.

Node j's voltage potential $\mathbf{U}_{ji}$ in Node i's identity current graph $\mathcal{G}^i$ is equal to the transpose of Node i's voltage potential $\mathbf{U}_{ij}$ in Node j's identity current graph $\mathcal{G}^j$.

Based on the analogy, the key idea of OSE-COV is that **calculating each block element $\mathbf{\Sigma}_{ij}$ in the covariance matrix $\mathbf{\Sigma}$ is analogous to calculating each node's general voltage potential $\mathbf{U}_{ij}$ in each node's identity current graph $\mathcal{G}^j$.**

3.3.1 OSE-COV for One Node Identity Current Graph

For simplicity, we start with the problem of calculating each node's voltage potential in Node i's identity current graph $\mathcal{G}^i$ in the distributed way. In Figure 3.4, there are 8 nodes, where Node 4 is an anchor node. The edges between nodes represent noisy measurements. For Node 1's identity current graph $\mathcal{G}^1$, an identity general current is imposed to Node 1. Each edge is associated with a general conduction $\mathbf{C}_{ij}$, which is equal to $\mathbf{P}_{ij}^{-1}$ for a linear measurement or $\mathbf{h}^T\mathbf{P}_{ij}^{-1}\mathbf{h}$ for a non-linear measurement (see Section 3.2.2.1 for details). Node 4 connects to ground. Assume the general voltage potentials for each node in $\mathcal{G}^1$ are $\mathbf{U}_{i1}, i = 1, 2, \cdots, 8$, respectively, and $\mathbf{U}_{41} = \mathbf{0}$ since Node 4 connects to ground. Note that the subscripts of a matrix here represent the block indexes of the nodes, e.g. $\mathbf{U}_{41}$ is the sub-matrix of the matrix $\mathbf{U}$ corresponding to Node 4's block row across with Node 1's block column. Without specifying, the following subscripts of a matrix indate block indexes.

According to the node voltage analysis technique, the following equations are obtained:

$$
\begin{aligned}
(\mathbf{U}_{11}-\mathbf{U}_{21})\mathbf{C}_{12}+(\mathbf{U}_{11}-\mathbf{U}_{51})\mathbf{C}_{15}+\mathbf{I}&=\mathbf{0}\\
(\mathbf{U}_{21}-\mathbf{U}_{11})\mathbf{C}_{21}+(\mathbf{U}_{21}-\mathbf{U}_{31})\mathbf{C}_{23}+(\mathbf{U}_{21}-\mathbf{U}_{61})\mathbf{C}_{26}&=\mathbf{0}\\
(\mathbf{U}_{31}-\mathbf{U}_{21})\mathbf{C}_{32}+(\mathbf{U}_{31}-\mathbf{0})\mathbf{C}_{34}+(\mathbf{U}_{31}-\mathbf{U}_{71})\mathbf{C}_{37}&=\mathbf{0}\\
(\mathbf{U}_{51}-\mathbf{U}_{11})\mathbf{C}_{51}+(\mathbf{U}_{51}-\mathbf{U}_{61})\mathbf{C}_{56}&=\mathbf{0}\\
(\mathbf{U}_{61}-\mathbf{U}_{51})\mathbf{C}_{65}+(\mathbf{U}_{61}-\mathbf{U}_{21})\mathbf{C}_{62}+(\mathbf{U}_{61}-\mathbf{U}_{71})\mathbf{C}_{67}&=\mathbf{0}\\
(\mathbf{U}_{71}-\mathbf{U}_{61})\mathbf{C}_{76}+(\mathbf{U}_{71}-\mathbf{U}_{31})\mathbf{C}_{73}+(\mathbf{U}_{71}-\mathbf{U}_{81})\mathbf{C}_{78}&=\mathbf{0}\\
(\mathbf{U}_{81}-\mathbf{U}_{71})\mathbf{C}_{87}+(\mathbf{U}_{81}-\mathbf{0})\mathbf{C}_{84}&=\mathbf{0}
\end{aligned}
\tag{3.25}
$$

Rewrite in the matrix form:

$$
\begin{bmatrix}
\mathbf{C}_{12}+\mathbf{C}_{15} & -\mathbf{C}_{12} & -\mathbf{C}_{15} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0}\\
-\mathbf{C}_{21} & \mathbf{C}_{21}+\mathbf{C}_{23}+\mathbf{C}_{26} & -\mathbf{C}_{23} & \mathbf{0} & -\mathbf{C}_{26} & \mathbf{0} & \mathbf{0}\\
\mathbf{0} & -\mathbf{C}_{32} & \mathbf{C}_{32}+\mathbf{C}_{34}+\mathbf{C}_{37} & \mathbf{0} & \mathbf{0} & -\mathbf{C}_{37} & \mathbf{0}\\
-\mathbf{C}_{51} & \mathbf{0} & \mathbf{0} & \mathbf{C}_{51}+\mathbf{C}_{56} & -\mathbf{C}_{56} & \mathbf{0} & \mathbf{0}\\
\mathbf{0} & -\mathbf{C}_{62} & \mathbf{0} & -\mathbf{C}_{65} & \mathbf{C}_{65}+\mathbf{C}_{62}+\mathbf{C}_{67} & -\mathbf{C}_{67} & \mathbf{0}\\
\mathbf{0} & \mathbf{0} & -\mathbf{C}_{73} & \mathbf{0} & -\mathbf{C}_{76} & \mathbf{C}_{76}+\mathbf{C}_{73}+\mathbf{C}_{78} & -\mathbf{C}_{78}\\
\mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & -\mathbf{C}_{87} & \mathbf{C}_{87}+\mathbf{C}_{84}
\end{bmatrix}
$$

$$
\times \begin{bmatrix}\mathbf{U}_{11}\\ \mathbf{U}_{21}\\ \mathbf{U}_{31}\\ \mathbf{U}_{51}\\ \mathbf{U}_{61}\\ \mathbf{U}_{71}\\ \mathbf{U}_{81}\end{bmatrix} = \begin{bmatrix}\mathbf{I}\\ \mathbf{0}\\ \mathbf{0}\\ \mathbf{0}\\ \mathbf{0}\\ \mathbf{0}\\ \mathbf{0}\end{bmatrix} \tag{3.26}
$$

Equation 3.26 is a linear system, which is similar to Equation 3.9. Therefore, it can be solved in the distributed way by using AFWAS method. To ease the description, we have the following symbols defined in Table 3.2. The distributed algorithm for calculating each node's voltage potential in Node 1's identity current graph is as follows:

- In the beginning, each non-anchor node i initializes its estimate of $\mathbf{U}_{i1}^{0}$ to an arbitrary value, and each anchor node initializes $\mathbf{U}_{i1}^{0}=\mathbf{0}$. By broadcasting twice, each node obtains its two hop measurement graph, its one hop neighbor's $\mathbf{U}_{j1}^{1}, j \in N^{i}(1)$, and two hop neighbor's $\mathbf{U}_{m1}^{0}, m \in N^{i}(2)$. The first two iterations are only for initialization, and thus $\mathbf{U}_{i1}^{2}=\mathbf{U}_{i1}^{1}=\mathbf{U}_{i1}^{0}, i \in V$.

- In the kth iteration, Node i assumes the voltage potential estimates from its two hop measurement graph's assuming anchor nodes $A^i(2)$ are correct, and uses node voltage analysis technique to update $\mathbf{U}_{i1}^k$. The linear system to be solved for Node i is:

$$\mathbf{L}^i(2)\mathbf{U}_{B^i(2)}^{i,1,k} = \mathbf{D}^{i,1}(2) + \mathbf{J}^i(2)\mathbf{U}_{A^i(2)}^{i,1,k} \tag{3.27}$$

where

  - $\mathbf{U}_{\mathbf{B}^i(2)}^{i,1,k} = [(\mathbf{U}_{v_1}^{i,1,k})^T, (\mathbf{U}_{v_2}^{i,1,k})^T, \cdots, (\mathbf{U}_{v_{|B^i(2)|}}^{i,1,k})^T]^T, v_j \in B^i(2), j = 1, 2, \cdots, |B^i(2)|$, each block element of which represents an assumed non-anchor node's (in Node i's two hop measurement graph) voltage potential needing to be solved by Node i in this iteration.
  - $\mathbf{L}^i(2)$ is the Kirchoff matrix of Node i's two hop measurement graph.

$$\mathbf{L}^i(2) = [\mathbf{L}^i(2)_{(v_j)(v_m)}] = \begin{cases} \sum \mathbf{C}_{v_j n} & \text{if } v_j = v_m \text{ and } (v_j, n) \in E^i(2) \\ -\mathbf{C}_{v_j v_m} & \text{if } v_j \neq v_m \text{ and } (v_j, v_m) \in E^{B^i(2)} \\ \mathbf{0} & \text{otherwise} \end{cases} \tag{3.28}$$
$$j = 1, 2, \cdots, |B^i(2)|$$

  - $\mathbf{D}^{i,1}(2) = [\mathbf{D}_{v_1}^{i,1}(2), \mathbf{D}_{v_2}^{i,1}(2), \cdots, \mathbf{D}_{v_{|B_i(2)|}}^{i,1}(2)]^T$, where

$$\mathbf{D}_{v_j}^{i,1}(2) = \begin{cases} \mathbf{I} & \text{if } v_j = 1 \\ \mathbf{0} & \text{otherwise} \end{cases} \tag{3.29}$$
$$j = 1, 2, \cdots, |B^i(2)|$$

  - $\mathbf{U}_{A^i(2)}^{i,1,k} = [(\mathbf{U}_{w_1}^{i,1,k})^T, (\mathbf{U}_{w_2}^{i,1,k})^T, \cdots, (\mathbf{U}_{w_{|A^i(2)|}}^{i,1,k})^T]^T, w_g \in A^i(2), g = 1, 2, \cdots, |A^i(2)|$, each block element of which is the current voltage potential estimate from the assumed anchor node, i.e. $\mathbf{U}_{w_g}^{i,1,k} = \mathbf{U}_{w_g 1}^{k-2}$.
  - $\mathbf{J}_i(2)$ is defined as:

$$\mathbf{J}_i(2) = [\mathbf{J}_{(v_j)(w_g)}] = \begin{cases} \mathbf{C}_{v_j w_g} & \text{if } (v_j, w_g) \in E_i(2), \text{ where } v_j \in B^i(2) \text{ and } w_j \in A^i(2) \\ \mathbf{0} & \text{otherwise} \end{cases} \tag{3.30}$$

After Node i solved Equation 3.27, it updated the voltage potential estimate $\mathbf{U}_{i1}^{k} = \lambda\mathbf{U}_{i}^{i,1,k} + (1-\lambda)\mathbf{U}_{i1}^{k-1}$, and then broadcasts it along with the estimates of its one hop neighbors to its one hop neighbors.

- The algorithm stops after N iterations or any other defined stopping conditions.

Table 3.2: OSE-COV symbol definition

Symbol	Definition
$G^i(n)$	Node i's n hop measurement graph. $G^i(n) = (V^i(n), E^i(n), F^i(n))$
$V^i(n)$	All the nodes in Node i's n hop measurement graph
$E^i(n)$	All the edges in Node i's n hop measurement graph
$F^i(n)$	The mapping which maps each edge to the corresponding measurement in Node i's n hop measurement graph
$N^i(h)$	Node i's n hop neighbors
$A^i(n)$	The assumed anchor nodes in Node i's n hop measurement graph, which includes all n hop nodes and the real anchor nodes
$B^i(n)$	The assumed non-anchor nodes in Node i's n hop measurement graph. $B^i(n) = V^i(n) - A^i(n)$
$G^{A^i(n)}$	The measurement graph consists of the assumed anchor nodes of Node i's n hop measurement graph. $G^{A^i(n)} = (V^{A^i(n)}, E^{A^i(n)}, F^{A^i(n)})$
$G^{B^i(n)}$	The measurement graph consists of the assumed non-anchor nodes of Node i's n hop measurement graph. $G^{B^i(n)} = (V^{B^i(n)}, E^{B^i(n)}, F^{B^i(n)})$
$\mathbf{U}_{ij}^{k}$	The estimate of Node i's general voltage potential in Node j's identity current graph in the kth iteration

Take Figure 3.4 for example. Assume it is in the 5th iteration, and Node 2 needs to calculate its voltage potential in Node 1's identity current graph. Node 2 has its 1 hop neighbors $N^2(1) = \{1, 3, 6\}$, and two hop neighbors $N^2(2) = \{4, 5, 7\}$. The assumed anchor nodes happen to be the same as its two hop neighbors, i.e. $A^2(2) = N^2(2)$, and its assumed non-anchor node set is $B^2(2) = \{1, 2, 3, 6\}$. At this time, Node 2 has the voltage potential estimates of its assumed non-anchor nodes of Iteration 4, which denote as $\mathbf{U}_{v_j 1}^{4}, v_j \in B^2(2)$, and the voltage potential estimates from its assumed anchor nodes in Iteration 3, which are

$\mathbf{U}^3_{w_g 1}, w_g \in A^2(2)$. Node 2 builds Equation 3.27 as follows:

$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 6 \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 2 & 3 & 6 \end{array} \\ \left[\begin{array}{cccc} \mathbf{C}_{12}+\mathbf{C}_{15} & -\mathbf{C}_{12} & \mathbf{0} & \mathbf{0} \\ -\mathbf{C}_{21} & \mathbf{C}_{21}+\mathbf{C}_{23}+\mathbf{C}_{26} & -\mathbf{C}_{23} & -\mathbf{C}_{26} \\ \mathbf{0} & -\mathbf{C}_{32} & \mathbf{C}_{32}+\mathbf{C}_{34}+\mathbf{C}_{37} & \mathbf{0} \\ \mathbf{0} & -\mathbf{C}_{62} & \mathbf{0} & \mathbf{C}_{62}+\mathbf{C}_{65}+\mathbf{C}_{67} \end{array}\right] \end{array} \times \begin{bmatrix} \mathbf{U}_1^{2,1,5} \\ \mathbf{U}_2^{2,1,5} \\ \mathbf{U}_3^{2,1,5} \\ \mathbf{U}_6^{2,1,5} \end{bmatrix}$$

$$= \begin{bmatrix} \mathbf{I} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix} + \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 6 \end{array} \begin{array}{c} \begin{array}{ccc} 4 & 5 & 7 \end{array} \\ \left[\begin{array}{ccc} \mathbf{0} & \mathbf{C}_{15} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{C}_{34} & \mathbf{0} & \mathbf{C}_{37} \\ \mathbf{0} & \mathbf{C}_{65} & \mathbf{C}_{67} \end{array}\right] \end{array} \times \begin{bmatrix} \mathbf{0} \\ \mathbf{U}^3_{51} \\ \mathbf{U}^3_{71} \end{bmatrix}$$

After solving this linear equation, we get the voltage potential $\mathbf{U}_2^{2,1,5}$ for Node 2 in its two hop measurement graph in Node 1's identity current graph. If $\lambda = 0.8$, we update $\mathbf{U}^5_{21} = 0.8\mathbf{U}_2^{2,1,5} + 0.2\mathbf{U}^4_{21}$. Then Node 2 broadcasts the latest estimates of itself and its one hop neighbors, i.e. $\{\mathbf{U}^5_{21}, \mathbf{U}^4_{11}, \mathbf{U}^4_{31}, \mathbf{U}^4_{61}\}$.

Note that each node can maintain more hops for its sub measurement graph. However, there is a tradeoff between the hop number each node maintains and the communication cost. The more hops each node maintains for its sub measurement graph, the faster it converges, but the more packages it needs to send in each iteration. Based on our simulation, two hop is a good balance between the convergency speed and the communication cost.

3.3.2 The OSE-COV Algorithm

Section 3.3 presents the distributed algorithm for calculating each node's voltage potential in one node's identity current graph. The final result set of all the nodes only corresponds to one column of the covariance matrix $\mathbf{\Sigma}$. To obtain the complete $\mathbf{\Sigma}$ and have a generic algorithm, OSE-COV needs to solve the following problems.

- How to calculate each node's voltage potential in each node's identity current graph?
- How to address the issue that each node does not know how many nodes are in the network?
- How to combine the position estimation with the covariance matrix?

3.3.2.1 Calculating Each Node's Voltage Potential in Each Node's Identity Current Graph

Take the same example as in above section. Now we calculate each node's voltage potential in not only Node 1's identity current graph, but also all the other nodes' identity current graphs. Rewrite Equation 3.26 as

$$\mathbf{L}\mathbf{U}_{*1} = \mathbf{I}_1$$

where symbol $*$ represents all the indexes, which are from 1 to 8. Then all the other nodes' identity current graph equations are

$$\begin{aligned} \mathbf{L}\mathbf{U}_{*2} &= \mathbf{I}_2 \\ \mathbf{L}\mathbf{U}_{*3} &= \mathbf{I}_3 \\ &\cdots \\ \mathbf{L}\mathbf{U}_{*8} &= \mathbf{I}_8 \end{aligned}$$

Stacking them together, we have:

$$\begin{aligned} & \mathbf{L}[\mathbf{U}_{*1}, \mathbf{U}_{*2}, \cdots, \mathbf{U}_{*8}] = [\mathbf{I}_1, \mathbf{I}_2, \cdots, \mathbf{I}_8] \\ \Rightarrow \quad & \mathbf{L}\mathbf{U} = \mathbf{I} \end{aligned} \tag{3.31}$$

It means that we need to solve 8 linear systems for Figure 3.4. Similarly, to calculate it in a distributed way, we need to solve 8 sub linear systems simultaneously for each node's two hop measurement graph. Assume at the kth iteration, the assumed anchor nodes of Node i's two hop measurement graph know the existence of all the nodes in the network, and have some voltage potential estimates about itself in each node's identity current graph, denoting as $\mathbf{U}^k_{w_g*}$, where $w_g \in A^i(2)$. Equation 3.27 is rewritten as follows

$$\begin{aligned} & \mathbf{L}^i(2)[\mathbf{U}^{i,1,k}_{B^i(2)}, \mathbf{U}^{i,2,k}_{B^i(2)}, \cdots, \mathbf{U}^{i,8,k}_{B^i(2)}] \\ = \quad & [\mathbf{D}^{i,1}(2), \mathbf{D}^{i,2}(2), \cdots, \mathbf{D}^{i,8}(2)] + \mathbf{J}^i(2)[\mathbf{U}^{i,1,k}_{A^i(2)}, \mathbf{U}^{i,2,k}_{A^i(2)}, \cdots, \mathbf{U}^{i,8,k}_{A^i(2)}] \\ \Rightarrow \quad & \mathbf{L}^i(2)\mathbf{U}^{i,*,k}_{B^i(2)} = \mathbf{D}^{i,*}(2) + \mathbf{J}^i(2)\mathbf{U}^{i,*,k}_{A^i(2)} \end{aligned} \tag{3.32}$$

We observed that the matrices $\mathbf{L}^i(2)$ and $\mathbf{J}^i(2)$ do not change in each node's identity current graph, since they represent the connection structures among assumed non-anchor nodes and among the assumed non-anchor nodes and assumed anchor nodes in Node i's two hop measurement graph, respectively. Note that since Node 4 is a true anchor node, all nodes' voltage potentials in Node 4's identity current graph are $\mathbf{0}$. Therefore, we can take out the elements related to Node 4 in the above equation, which is rewritten as follows:

$$\mathbf{L}^i(2)[\mathbf{U}^{i,1:3,k}_{B^i(2)}, \mathbf{U}^{i,5:8,k}_{B^i(2)}] = \quad [\mathbf{D}^{i,1:3}(2), \mathbf{D}^{i,5:8}(2)] + \mathbf{J}^i(2)[\mathbf{U}^{i,1:3,k}_{A^i(2)}, \mathbf{U}^{i,5:8,k}_{A^i(2)}]$$

where the symbol $m : n$ in the superscript represents all the elements with the superscripts from m to n.

After solving the above equation, Node i updates its voltage potentials in each node's identity current graph

$$\mathbf{U}^k_{i*} = \lambda \mathbf{U}^{i,*,k}_i + (1 - \lambda)\mathbf{U}^{k-1}_{i*} \tag{3.33}$$

3.3.2.2 Obtaining the Knowledge of All Nodes

Using the above algorithm, each node updates its corresponding row in the covariance matrix $\mathbf{\Sigma}$. The assumption for the above algorithm is that each node knows the existence of all the other nodes in the network and can obtain the rows from its two hop measurement graph assumed anchor nodes. It is possible that each node obtains the knowledge via a discovery flooding. However, for a dynamic case, where nodes may join or leave the network frequently, this costs a lot. Instead, we prefer an ad-hoc method where nodes can discovery and obtain this knowledge only through the one hop broadcasting in each iteration.

To achieve this, each node needs to store its one-hop and two-hop neighbors's known node lists M_j^k's and their voltage potential estimates $\mathbf{U}_{jM_j}^k$'s in their known nodes' identity current graphs. Take Node 2 in Figure 3.4 for example. In Iteration k, Node 2 may get its' one-hop and two-hop nodes' the known node lists and voltage potential estimates via its one-hop neighbor's broadcasting: $M_1^{k-1} = (1, 2, 5, 6)$, $M_3^{k-1} = (2, 3, 4, 6, 7, 8)$, $M_6^{k-1} = (1, 2, 3, 5, 6, 7, 8)$, $M_4^{k-2} = (3, 4, 8)$, $M_5^{k-2} = (1, 5, 6)$, $M_7^{k-2} = (3, 6, 7, 8)$, and $\mathbf{U}_{1M_1}^{k-1}$, $\mathbf{U}_{3M_3}^{k-1}$, $\mathbf{U}_{6M_6}^{k-1}$, $\mathbf{U}_{4M_4}^{k-2}$, $\mathbf{U}_{5M_5}^{k-2}$, $\mathbf{U}_{7M_7}^{k-2}$. This means that Node 2 now knows the union of all these known node lists in this iteration, i.e. $M_2^k = (1, 2, 3, 4, 5, 6, 7, 8)$. Hence, it needs to solve its voltage potential in all these nodes' identity current graph.

In Equation 3.32, each block row of $\mathbf{U}_{A^i(2)}^{i,*,k}$ represents an assumed anchor node's voltage potential in all nodes' identity current graphs. However, in Iteration k, Node i and the assumed anchor nodes of Node i's two hop measurement graph may not know the existence of all the nodes. Therefore, Node i can only calculate the voltage potentials in its known nodes' identity current graphs. Additionally, the assumed anchor nodes of Node i's two hop measurement graph know different sets of nodes, and thus $\mathbf{U}_{A^i(2)}^{i,*,k}$ needs to be aligned. We use 0 to fill the holes of the missed voltage potentials. Rewrite the $\mathbf{U}_{A^i(2)}^{i,*,k}$ as:

$$\mathbf{U}_{A^i(2)}^{i,*,k} = [\mathbf{U}_{w_g j}^{i,j,k}] \begin{cases} \mathbf{U}_{w_g j}^{i,j,t} & \text{if } j \in M_{w_g}^t \\ \mathbf{0} & \text{otherwise} \end{cases} \tag{3.34}$$

where $w_g \in A^i(2)$, and $t = k - 1$ if $w_g \in N^i(1)$ or $t = k - 2$ if $w_g \in N^i(2)$. Continue the above example, in Iteration k, Node 2 has

$$\mathbf{U}_{A^2(2)}^{2,*,k} = \begin{array}{c} \\ 4 \\ 5 \\ 7 \end{array} \left[\begin{array}{cccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ \mathbf{0} & \mathbf{0} & \mathbf{U}_{43}^{2,3,k-2} & \mathbf{U}_{44}^{2,4,k-2} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{U}_{48}^{2,8,k-2} \\ \mathbf{U}_{51}^{2,1,k-2} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{U}_{55}^{2,5,k-2} & \mathbf{U}_{56}^{2,6,k-2} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{U}_{73}^{2,3,k-2} & \mathbf{0} & \mathbf{0} & \mathbf{U}_{76}^{2,6,k-2} & \mathbf{U}_{77}^{2,7,k-2} & \mathbf{U}_{78}^{2,8,k-2} \end{array} \right]$$

In summary, in each iteration, each node obtains its one-hop and two-hop neighbors' known node lists and their voltage potential estimates in the corresponding known nodes' identity current graphs, builds $\mathbf{U}_{A^i(2)}^{i,*,k}$ using Equation 3.34, updates its own voltage potential estimates in the latest known nodes' identity current graphs using Equations 3.32 and 3.33, and then broadcasts its and its one neighbors' latest known node lists and voltage potential estimates.

3.3.2.3 The OSE-COV Algorithm Summary and Analysis

The above algorithm only describes the covariance estimation, but it is not difficult to combine with the position estimation. Since the calculation scheme of OSE-COV is almost the same as OSE, we only need to append one position estimate in each node's broadcasting iteration. The whole OSE-COV algorithm is summarized as follows:

- In the beginning, each node (using Node i for example) initializes its known node list $M_i^0 = (i)$, its estimated voltage potentials in each known node list as $\mathbf{U}_{iM_i}^0 = \mathbf{0}$ (for non anchor nodes, they can use arbitrary values, but for anchor nodes, they must be $\mathbf{0}$), and its estimated position $\mathbf{x}_i^0$ as a random valued vector. By broadcasting twice, each node obtains its two-hop measurement graph $G^i(2)$, its one-hop neighbors' position estimates $\mathbf{x}_j$, known node lists M_j^1 and their voltage potential estimates in their known nodes identity current graphs $\mathbf{U}_{jM_j}^1$, where $j \in N^i(1)$, and its two-hop neighbors' position estimates $\mathbf{x}_g$, known node lists M_g^0 and the corresponding $\mathbf{U}_{gM_g}^0$, where $g \in N^i(2)$.

- In the kth iteration, Node i receives the latest position estimates, known node lists and the voltage potential estimates in the known nodes's identity current graphs of both its one-hop and two-hop neighbors from its one-hop neighbors. For the position estimates, it uses OSE [80] to update its latest position estimate $\mathbf{x}_i^k$. For the covariance estimate, it first updates its known node list as

$$M_i^k = M_i^{k-1} \cup \bigcup_{j \in N^i(1)} M_j^{k-1} \cup \bigcup_{g \in N^i(2)} M_g^{k-2} \tag{3.35}$$

  Then it builds Equation 3.32, where $\mathbf{L}^i(2)$, $\mathbf{D}^{i,1}(2)$, $\mathbf{J}^i(2)$ and $\mathbf{U}_{A^i(2)}^{i,1,k}$ are defined in 3.28, 3.29, 3.30 and 3.34, respectively. By solving the equation, Node i obtains $\mathbf{U}_{B^i(2)}^{i,*,k}$. Then it updates its voltage potential estimates in its known nodes' identity current graphs using Equation 3.33.

- After updating, Node i broadcasts itself's and its one-hop neighbors' latest position estimates, known node lists and the voltage potential estimates in known nodes' identity current graphs to its one-hop neighbors.

- The algorithm stops after N iterations or any other defined stopping conditions.

The convergency property of OSE-COV can be proved by using the AWAS framework [234]. Moreover, another important feature of OSE-COV is that the whole process does not need to be synchronized, and it can endure unreliable communication links. Assume the whole network has N nodes, and for Node i, it has $|N^i(1)|$ number of one-hop neighbors. A real number is represented in 32 bits, and a node Id is represented by a 32 bit integer. In each iteration, the computational complexity is $O((|N^i(1)|+|N^i(2)|)^3)$. The communication complexity is $O_{M_i} + O_{M_{N^i(1)}} + O_{\mathbf{\Sigma}_{i*}} + O_{\mathbf{\Sigma}_{N^i(1)*}} + O_{\mathbf{x}_i} + O_{\mathbf{x}_{N^i(1)}} = 4N + 4N|N^i(1)| + 4 \times 4N + 4 \times 4N|N^i(1)| + 2 \times 4 + 2 \times 4|N^i(1)| = O(N|N^i(1)|)$, where O_{M_i}, $O_{M_{N^i(1)}}$, $O_{\mathbf{\Sigma}_{i*}}$, $O_{\mathbf{\Sigma}_{N^i(1)*}}$, $O_{\mathbf{x}_i}$, and $O_{\mathbf{x}_{N^i(1)}}$ denote the costs for broadcasting the known node list of Node i, the known node lists of Node i's one-hop neighbors, the voltage potential estimates in the known nodes' identity current graphs of Node i, the voltage potential estimates in the

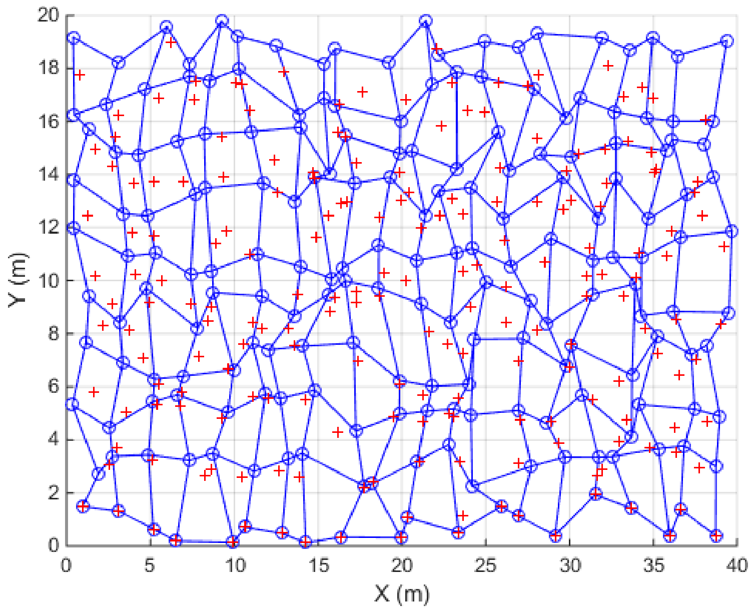


Figure 3.5: Inova and OSE-COV simulation configuration

known nodes' identity current graphs of Node i's one-hope neighbors, the position estimate of Node i, and the position estimates of Node i's one-hope neighbors, respectively.

3.4 Evaluation

In this section, we evaluate the localization accuracy, the time complexity of INOVA, and the convergence speed of the distributed algorithm–OSE-COV. All the evaluations are based on Matlab simulations on a PC with dual Intel(R) Core(TM)2 CPU 6600 @2.40GHz, 2G memory and the Windows XP SP3 operating system.

The baseline configuration is shown in Figure 3.5. 200 nodes are distributed in an area of 20m by 40m (each grid is a 2m $\times$2m square). Each grid has one node randomly deployed inside the area (the blue circle in the figure). The measurements are only generated between the nodes who are in the grids next to each other (the blue lines). A measurement can be a relative range, a relative angle or a relative position measurement. A relative range measurement is generated by $\tilde{r}_{ij} = r_{ij} + \epsilon^{r}_{ij}$, where $\tilde{r}_{ij}$ is the generated noisy range measurement between Nodes i and j, r_{ij} is the real distance, and ϵ_{ij}^r is the noise random variable, which follows $\epsilon_{ij}^r \backsim \mathcal{N}(0, \sigma_r^2)$. To mimic the heterogeneous measurement conditions, we set σ_r^2 a random number, which follows the uniform distribution in the range $[0.01, 0.4 \times r_{ij}]$. A relative angle measurement is generated by $\tilde{\theta}_{ij} = \theta_{ij} + \epsilon_{ij}^{\theta}$, where $\tilde{\theta}_{ij}$ is the generated noisy angle measurement between Nodes i and j, θ_{ij} is the real angle, and ϵ_{ij}^{θ} is the noise random variable, which follows $\epsilon_{ij}^{\theta} \backsim \mathcal{N}(0, \sigma_{\theta}^2)$. Similarly, σ_{θ}^2 is a uniform random variable in the range $[(1^\circ \times \pi/180^\circ)^2, (10^\circ \times \pi/180^\circ)^2]$. A relative position measurement is generated by combining both the range measurement and the angle measurement, which is $\tilde{\mathbf{z}}_{ij} = [\tilde{r}_{ij} \cos \tilde{\theta}_{ij}, \tilde{r}_{ij} \sin \tilde{\theta}_{ij}]^T$, and thus its covariance matrix is

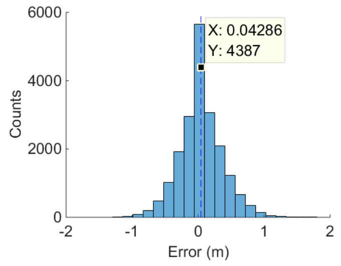
$$P_{ij} = \begin{bmatrix} y^2 \sigma_{\theta_{ij}}^2 + \sigma_r^2 \cos^2 \theta_{ij} & -xy\sigma_{\theta_{ij}}^2 + \sigma_r^2 \sin(2\theta_{ij})/2 \\ -xy\sigma_{\theta_{ij}}^2 + \sigma_r^2 \sin(2\theta_{ij})/2 & x^2 \sigma_{\theta_{ij}}^2 + \sigma_r^2 \sin^2 \theta_{ij} \end{bmatrix} \tag{3.36}$$

where $x = r_{ij} \cos \theta_{ij}$ and $y = r_{ij} \sin \theta_{ij}$.

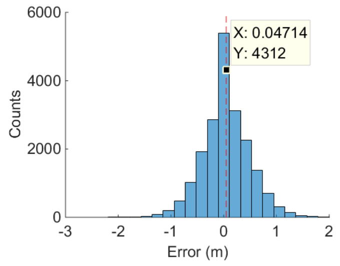
In order to be localizable, the measurements vertically connecting grids are relative position measurements. The horizontal measurements are relative position measurements, relative range measurements or relative angle measurements depending on the interests of the experiments. The bottom row of nodes are anchor nodes (the nodes in the rectangle $0 \leq x \leq 40, 0 \leq y \leq 20$).

3.4.1 Inova Localization Accuracy

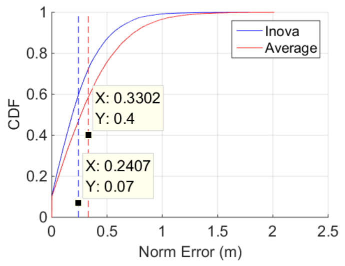
Figure 3.6 compares the localization accuracy between Inova and the averaged method. The averaged method is defined as the localization method which does not take advantage of the measurement uncertainty. In this simulation, we use the same measurement set for both methods, but constantly use the maximum measurement covariances of the range measurements and the angle measurements for the average-based method, i.e. $\tilde{\sigma}_r^2 = 0.4 \times 2$ (the grid side length) and $\tilde{\sigma}_{\theta}^2 = (10^\circ \times \pi/180^\circ)^2$. We use the default configuration described in the above section, and run 50 times for each category of measurements (relative position,



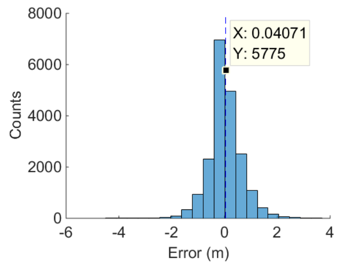
(a) Inova errors (relative position measurements)



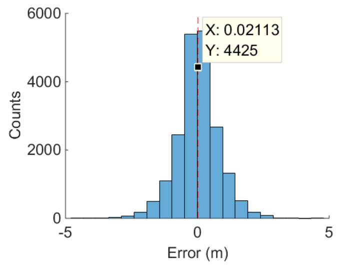
(b) The average-based method errors (relative position measurements)



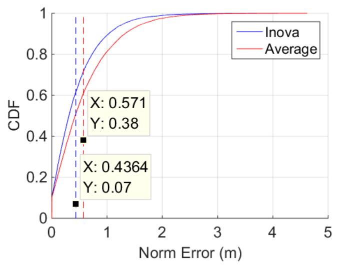
(c) The error cumulative density function comparisons (relative position measurements)



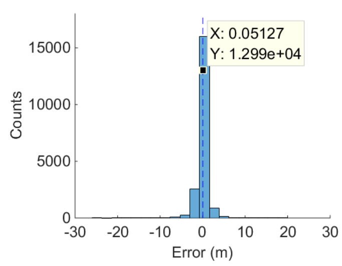
(d) Inova errors (relative range measurements)



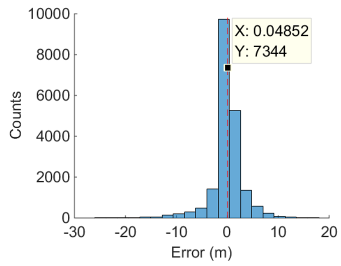
(e) The average-based method errors (relative range measurements)



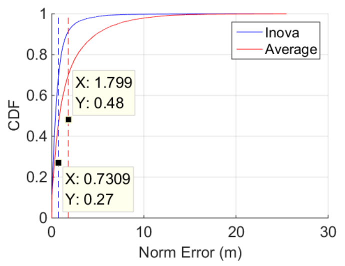
(f) The error cumulative density function comparisons (relative range measurements)



(g) Inova errors (relative angle measurements)



(h) The average-based method errors (relative angle measurements)



(i) The error cumulative density function comparisons (relative angle measurements)

Figure 3.6: The localization accuracy comparison between Inova and the average-based method.

relative range, and relative angle measurements).

Figures 3.6(a), 3.6(b) and 3.6(c) are based on the relative position measurements; Figures 3.6(d), 3.6(e) and 3.6(f) are based on the (horizontal) relative range measurements; and Figures 3.6(g), 3.6(h) and 3.6(i) are based on the (horizontal) relative angle measurements. The first two plots of each category are the histograms of the localization errors of Inova

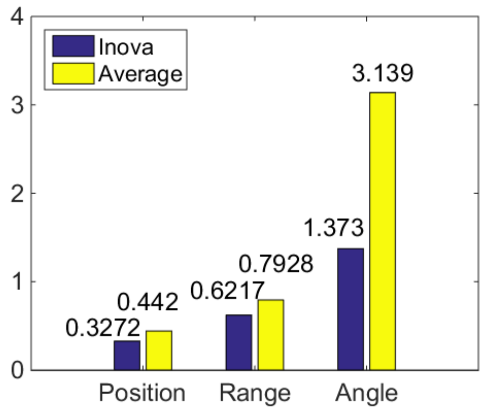


Figure 3.7: The error standard deviation comparison between Inova and the average-based method for 50 runs

and the average-based method, respectively, and the third plot is the cumulative density function (CDF) comparison of the norm errors of these two methods. The errors in the first two plots are defined as $\mathbf{x}_i - \hat{\mathbf{x}}_i$, where $\mathbf{x}_i$ is the real position of Node i, and $\hat{\mathbf{x}}_i$ is the estimated position. The norm errors in the third CDF plot is defined as $|\mathbf{x}_i - \hat{\mathbf{x}}_i|$.

From the figures, we see that the mean errors of the first two figures of each category are very close to 0, and the average-based method is even better than Inova. This is because both of them are unbiased estimators. However, the width of the histogram of the average-based method is bigger than that of Inova, which means Inova is more reliable than the average-based method. To eliminate the compensation of the positive and negative errors, we use the norm error in the third column of figures (Figures 3.6(c), 3.6(f) and 3.6(i)). We plot their CDFs and norm error means. We see that Inova outperforms the average-based method. For the relative position measurements based simulation, Inova improves the accuracy by 27.01% over the average-based method; for the relative range measurements based simulation, Inova improves the accuracy by 23.57%; and for the relative angle measurements, Inova improves by 59.37%.

Another important metric for localization method is the standard deviation of the errors. It represents the reliability of the method. Figure 3.7 shows the error standard deviation comparison between Inova and the average-based method for 3 different types of measurements. We see that Inova improves the standard deviations by 25.99%, 21.58% and 56.25% for the position, range and angle based simulations, respectively.

3.4.2 Linearization Analysis

When using the nonlinear measurements such as relative range and relative angle measurements, Inova needs to linearize them. In this process, the linearization errors are introduced. As described in Section 3.2.3.1, the measurement model is $\mathbf{z} = h(\mathbf{x}) + \omega$, and its linearized form is $\tilde{\mathbf{z}} = h(\hat{\mathbf{x}}) + \mathbf{H}(\mathbf{x} - \hat{\mathbf{x}}) + \omega$. Therefore, the linearization error is

$$\mathbf{e}_l = \mathbf{z} - \tilde{\mathbf{z}} = h(\mathbf{x}) - h(\hat{\mathbf{x}}) - \mathbf{H}(\mathbf{x} - \hat{\mathbf{x}}) \tag{3.37}$$

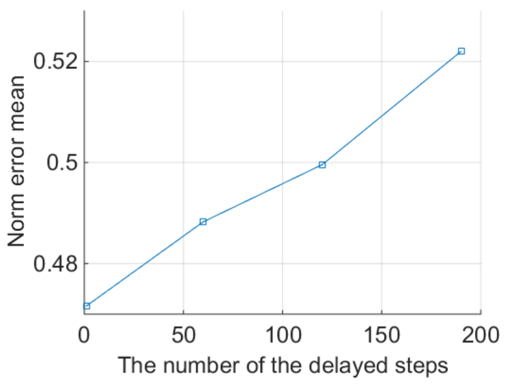
We have

$$\lim_{\hat{\mathbf{x}} \to \mathbf{x}} \mathbf{e}_l = \mathbf{0} \tag{3.38}$$

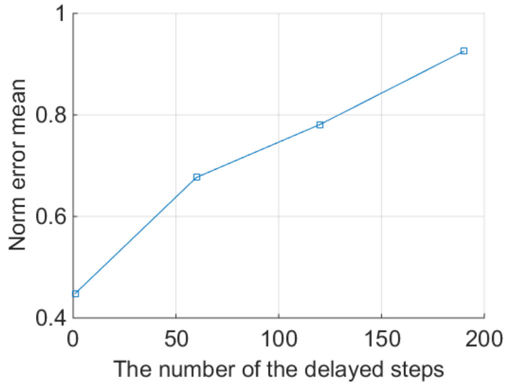
This means the more $\hat{\mathbf{x}}$ is close to $\mathbf{x}$, the less the linearization error is.

Another important property of Inova is that it can delay the location estimation. As analyzed in Section 3.2.3.1, each update for matrices $\mathbf{L}$ and $\mathbf{b}$ only needs $O(1)$ computational complexity, while the estimation needs $O(n^3)$ (n is the number of the nodes in the network). Therefore, delaying the estimation for every k measurements reduces the total computational complexity by k times comparing with estimating for every measurement. However, the side effects of delayed estimation are not only to cause the estimation lag, but also to have more errors if the measurements are in nonlinear form.

Usually, the more measurements we have, the better the localization accuracy we can achieve. According to Equation 3.38, we should update the nodes' location estimates with the latest measurements as soon as possible (without delaying estimation) in order to bring the estimates closer to the real locations, and thus reduce the linearization error for the following nonlinear measurements. Therefore, it is a trade-off between the number of the



(a) Relative range measurements



(b) Relative angle measurements

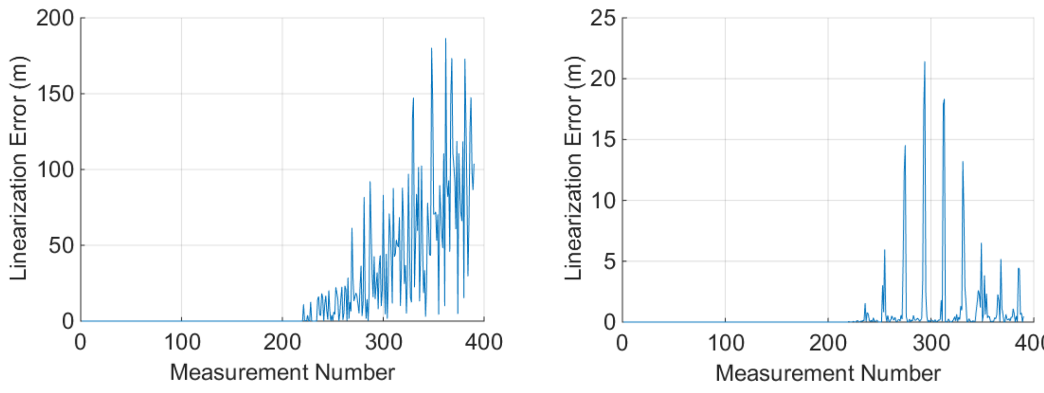
Figure 3.8: The localization accuracy for different delaying steps

delaying steps and the linearization errors.

Figure 3.8 shows how the number of delaying steps impacts the localization errors. Each point in each of the two figures is an average of 10 runs. The number of delaying steps are from 1 (no delay) to 190 (only estimating at the last measurement). The left figure is based on relative range measurements, and the right one is based on relative angle measurements. Both figures show that as the number of delaying steps increases, the localization error also increases. This tells us that depending on different applications, we should choose different number of delaying steps. For resource constrained devices, we may choose a large number, while for accuracy critical application, we may need to reduce the delaying steps.

Another important issue needs to point out is that we should be very careful of using the relative angle measurements in practice. To get the relative angle between two nodes, we usually uses $atan2$ function, whose range is $(-\pi, \pi]$. However, if the relative angle is close to π, a little error may cause the actual result to be $-\pi + \epsilon$, which will introduce a very large linearization error.

Initially, for the relative angle measurements configuration, we set all the horizontal measurements as relative angle measurements and the direction is from left to right, e.g. Node $[1, 1.1]^T$ points to Node $[3, 1]^T$, and we have the angle $atan2(1.1 - 1, 1 - 3)$ which is close to π. This makes all the relative angle measurements close to π (or $-\pi$). The



(a) Relative angle measurements close to π (b) Relative angle measurements not close to π

Figure 3.9: Linearization errors for each measurement

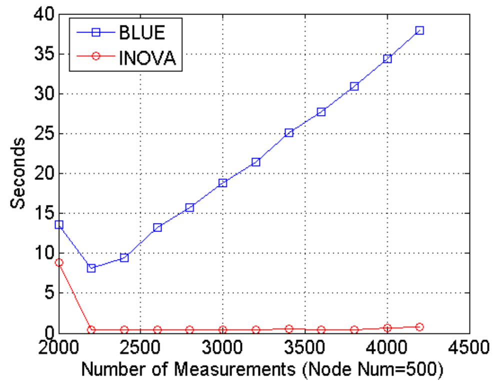
mean error for the 50 runs of Inova is 15.2516 m comparing with 0.7309 m if we set the measurement direction opposite.

Figure 3.9 shows the linearization errors for each measurement in the relative angle measurements based simulation. The left figure is for relative angle measurements close to π, while the right one is for the angles not close to π. We see that when the relative angle is close to π, the linearization error is huge, which results in the divergence of the estimation. Therefore, in practice, if we obtain some relative angle measurements which are close to π, we should change its direction, i.e. change the measurement $\mathbf{z}_{ij} = atan2(y_i - y_j, x_i - x_j)$ to $\mathbf{z}_{ji} = atan2(y_j - y_i, x_j - x_i)$.

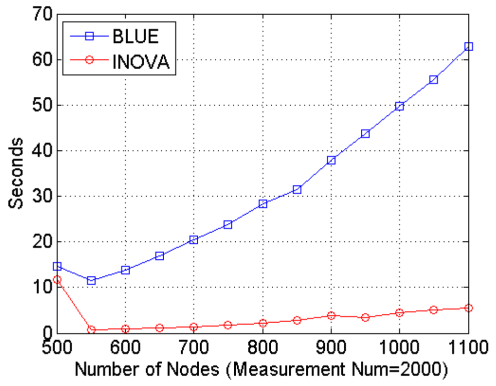
3.4.3 The efficiency of Inova

Although BLUE [69, 80] does not deal with nonlinear measurements, when localizing with only relative position measurements, it has the same accuracy with Inova. However, as stated at the beginning of this chapter, Inova is highly efficient for dynamic networks where nodes are likely to join or leave, or measurements keep being generated at run time.

To evaluate the efficiency of Inova, we use the following configuration. 500 nodes are deployed in a 100 m by 100 m area with one anchor node at the center. 2000 relative position measurements are generated to fully connect them. To evaluate the case where the



(a) The localization time when measurements keep being generated

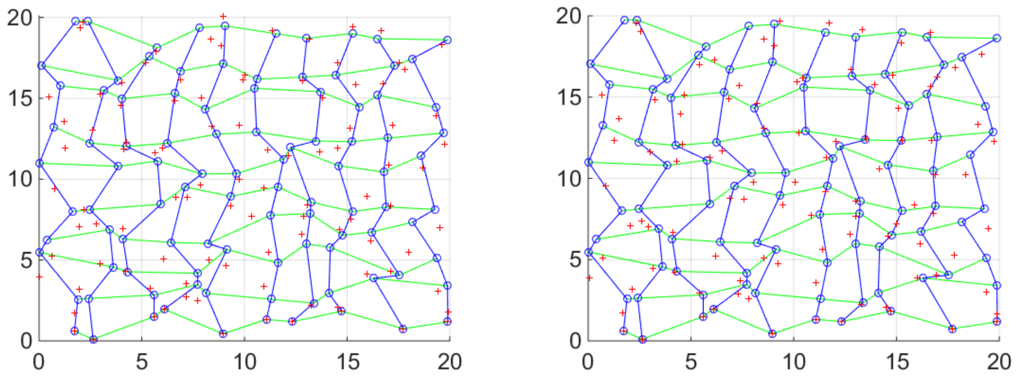


(b) The localization time when nodes keep joining the network

Figure 3.10: Localization efficiency of Inova and BLUE

number of the nodes does not change, but the measurements keep being generated, each time 200 relative position measurements are added till there are 4200 measurements totally (including the first 2000 measurements). The localization times of Inova and BLUE are compared at each step. To evaluate the case where nodes dynamically join the network, starting from the same configuration, 50 nodes are randomly added at each step, and the localization times are compared between Inova and BLUE.

Figure 3.10(a) shows the execution time for INOVA and BLUE as new measurements are incrementally generated. The result shows that initially, to localize a network with 500 nodes and 2000 measurements, BLUE takes 13.5 seconds, while INOVA only needs 8.8 seconds. After that, 200 measurements are incrementally generated in each iteration. The execution time for BLUE increases very quickly, while INOVA almost keeps a constant time. When there are 4000 measurements in the network, the computational time of INOVA is only 0.5 second, which is 70 times as fast as BLUE. Similarly, in Figure 3.10(b), which fixes the measurement number at 2000, but incrementally adds 50 nodes each time, the execution time of BLUE shows a large rate of rise, while INOVA only increases slightly. When there are 1100 nodes in the network, the computational time of INOVA is 5 seconds, which is 12 times as fast as BLUE.



(a) Inova estimation

(b) OSE-COV estimation (200 iterations)

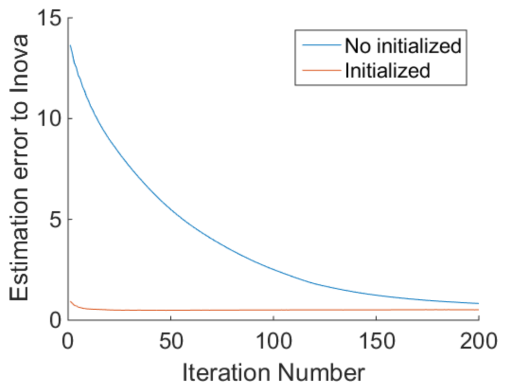
Figure 3.11: Localization comparison between Inova and OSE-COV

This experiment shows that in a large network where measurements are continuously generated or nodes dynamically join or leave the network, INOVA can provide real-time estimates, while BLUE fails. Based on this result, we can even apply INOVA to a mobile network. When a node moves, this node is considered to leave the network, and a new node joins the new position with new measurements. Since INOVA has very low execution time, it can respond to these mobile behaviors very quickly.

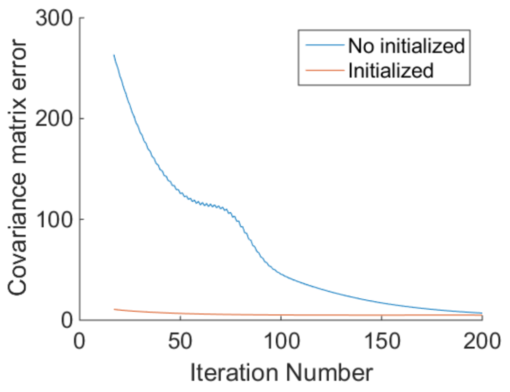
3.4.4 The localization accuracy and convergence speed of OSE-COV

For the distributed localization algorithm OSE-COV, we are interested in the localization accuracy and convergence speed. Theoretically, OSE-COV tends asymptotically towards the estimates of Inova if $k \to \infty$, where k is the number of the iterations. Note that we are interested in not only the position estimates, but also the covariance matrix.

To evaluate this, we use the same configuration as described in Section 3.4, but with 100 nodes in the area of 10 m $\times$ 10 m. Figure 3.11 shows the localization estimation results of Inova and OSE-COV. The blue circles are the real node positions, and the red crosses are the estimated positions. The blue lines are the relative position measurements, and the green ones are the relative range measurements. The total number of iterations is 200. We see that the estimates are very close between Inova and OSE-COV.



(a) The position estimation error of OSE-COV (against Inova)



(b) The covariance matrix estimation error of OSE-COV (against Inova)

Figure 3.12: OSE-COV convergency performance

Figure 3.12 shows the convergence performance of OSE-COV. Figure 3.12(a) shows the node position estimation convergence performance, and Figure 3.12(b) shows the covariance matrix estimation convergence performance. The position estimation error is defined as the mean square root error of the Euclidean distance between the estimates of Inova and OSE-COV, which is $e_p = \sqrt{\frac{\sum_n \|\hat{\mathbf{x}}_i^{Inova} - \hat{\mathbf{x}}_i^{OSE-COV}\|^2}{n}}$, where n is the node number, $\hat{\mathbf{x}}_i^{Inova}$ is Inova's position estimates for Node i, and $\hat{\mathbf{x}}_i^{OSE-COV}$ is OSE-COV's position estimates for Node i. The error for the covariance matrix is defined as $e_c = \frac{\sqrt{\sum_n^i \sum_n^j c_{ij}^{Inova} - c_{ij}^{OSE-COV}}}{n^2}$, where c_{ij}^{Inova} is the element of Row i and Column j of the covariance matrix of Inova, and $c_{ij}^{OSE-COV}$ is the corresponding element of the covariance matrix estimated by OSE-COV.

The blue curves in both figures are obtained by OSE-COV without any heuristic initialization, where the initial position for each node is $[0, 0]^T$ with some jitter, and the initial covariance matrix for each node's position estimation is $\begin{bmatrix} 100 & 0 \\ 0 & 100 \end{bmatrix}$. We see that as iteration number increases, the errors for both the position and covariance matrix converge.

To make the convergence speed faster, a better initial estimates can be fed into each node. By flooding a message from each anchor node and recording the route information, each node can obtain at least one strong path (only consisting of relative position measurements) to one anchor node. By using the measurements along this strong path, each node obtains a much better initial position estimate and the corresponding covariance matrix. The red curves in Figures 3.12(a) and 3.12(b) are the estimation results of OSE-COV with the better initialization. We see that after 3 iterations, the position estimation with the heuristic initialization outperforms the one without it after 200 iterations. Similarly, it only takes 45 iterations to beat the one without the heuristic initialization for the covariance matrix estimation.

3.5 Discussion

In this chapter, we solve the localization problem for stationary sensor networks with different types of relative measurements, which include GPS, relative position, relative range and relative angle measurements. We first model the localization problem with a measurement graph, then analogize the measurement graph to a generic electrical network, and borrow the electrical network theory (node voltage analysis) to solve it.

The nodes and measurements in a measurement graph are mapped to electrical nodes and connections in an electrical network. A covariance matrix (or a variance) of a measurement is analogous to a resistance, or its relation information matrix (for nonlinear case) is analogous to a conduction. Node i's position estimation covariance matrix is corresponding to its voltage potential in this electrical network when an identity current is imposed onto Node i, which is called Node i's identity current graph. The block off diagonal element $\mathbf{U}_{ij}$ of the covariance matrix corresponds to the voltage potential of Node i in Node j's identity current graph.

In Section 3.2, we propose a centralized localization algorithm called Inova, which uses the node voltage analysis technique to solve the localization problem in an incremental way. Inova also solves the nonlinear measurement case issue.

Section 3.3 uses the same node voltage analysis technique and AFWAS framework [234] to estimate the covariance matrix in a decentralized way. Along with the OSE algorithm [80], the OSE-COV algorithm is proposed, which is able to simultaneously estimate node positions and the covariance matrix in a same framework.

Section 3.4 evaluates the performance of Inova and OSE-COV. The metrics include the localization accuracy, linearization errors, localization efficiency and convergency speed. For the localization accuracy, we compare the Inova algorithm and the average-based method. Inova improves the accuracy from 23.57% to 59.37% in a 200 node network. For the linearization analysis, we found that the relative angle measurement may cause large errors when the angle is close to π. The solution of this is to reverse the direction of the measurement so that the angle turns to be close to 0. For the efficiency evaluation, we compare Inova with BLUE. When a network keeps generating measurements (when there are 4000 meausrements), Inova performs 70 times as fast as BLUE for making use of all these measurements. When a large number of nodes keep joining the network (when there are 1100 nodes in the network), Inova is 12 times as fast as BLUE. For the convergence of OSE-COV, we found that as the number of iterations increases, the position and covariance matrix estimation tend asymptotically towards the results of Inova. Additionally, a heuristic initialization dramatically improves the convergence speed of OSE-COV.

Chapter 4

Quantitative Uncertainty Analysis and Usage

As stated in Chapter 3, one important feature of a localization system is that it should model system uncertainty properly, which includes the measurement uncertainty and estimation uncertainty. We already show that modeling the measurement uncertainty produces more accurate estimation. In this chapter, the estimation uncertainty is discussed. We first prove that using INOVA or OSE-COV the position estimate follows the multi-variate normal distribution. Then, we show how to use estimation uncertainty to select anchor nodes for a static sensor network. Actually, besides the anchor selection, the localization uncertainty has many other usages, such as working as a criteria for evaluating a localization system, rejecting estimation outliers, and providing the probability of finding a node within a certain region.

4.1 Normality Proof

In Section 3.2.2, we already deduced the first two moments of each node's position (position estimation and the covariance matrix) for the INOVA algorithm. In this section, we further prove that by using INOVA, each node's position estimate follows a multivariate normal distribution.

In INOVA, a node's position estimate is essentially a linear combination of all measurements in the network. Intuitively, based on the large number theorem from probability theory, the estimate is very likely to follow a multivariate normal distribution. One version of the central limit theorem, named Lindeberg's condition [235], says that for a sequence of independent random variables $X_k, k = 1, 2, \cdots, n$, with $E(X_k) = \mu_k$ and $Var(X_k) = \sigma_k^2$, denoting $s_n^2 = \sum_{k=1}^{n} \sigma_k^2$, if $\max_{k=1,2,\cdots,n} \frac{\sigma_k^2}{s_n^2} \rightarrow 0$, as $n \rightarrow \infty$, i.e. none of σ_k^2 dominates s_n^2, then $Z_n = \sum_{k=1}^{n}(X_k - \mu_k)/s_n$ converges to a standard normal distribution $\mathcal{N}(0,1)$ when $n \rightarrow \infty$.

In INOVA, each node's position estimate is $\hat{x}_u = \sum_{k=1}^{m} W_k^u \zeta_k$, where W_k^u is the weight matrix of the measurement ζ_k Node u (or the generalized current of edge e_k in $\mathcal{G}^u$). Its corresponding covariance matrix is $\Sigma_{uu} = \sum_{k=1}^{m} W_k^u P_k (W_k^u)^T$. Let $y_k^u = W_k^u \zeta_k$. We have $\Sigma_{uu} = \sum_{k=1}^{m} COV(y_k^u)$. Define

$$Z^u = \Sigma_{uu}^{-\frac{1}{2}} \sum_{k=1}^{m} (y_k^u - E(y_k^u)) \tag{4.1}$$

If P_k is a diagonal matrix and $\forall k = 1, 2, \cdots, m, COV(y_k^u)$ does not dominate Σ_{uu}, then Z^u tends to be a standard multivariate normal random vector, i.e. $Z^u \backsim \mathcal{N}_d(\mathbf{0}, \mathbf{I})$, since each element $z_i^u, i = 1, 2, \cdots, d$ of Z^u tends to be a standard normal random variable under Lindeberg's condition. In INOVA, the latter condition ($COV(y_k^u)$ does not dominate) is satisfied, because in practice a large number of measurements are generated by a limited number of devices, and none of a single measurement's uncertainty should dominate. If there is some very inaccurate measurement, we can use some simple outlier detection algorithm to filter it out. For the first condition, based on our simulation, even if P_k is not a diagonal matrix, Z^u still tends to be $\mathcal{N}_d(\mathbf{0}, \mathbf{I})$. Consequently, we derive that

$$\begin{aligned} Z^u &= \Sigma_{uu}^{-\frac{1}{2}} (\sum_{k=1}^{m} (y_k^u) - \sum_{k=1}^{m} E(y_k^u)) = \Sigma_{uu}^{-\frac{1}{2}} (\hat{x}_u - x_u) \\ \Longrightarrow & \quad \hat{x}_u \backsim \mathcal{N}_d(x_u, \Sigma_{uu}) \end{aligned} \tag{4.2}$$

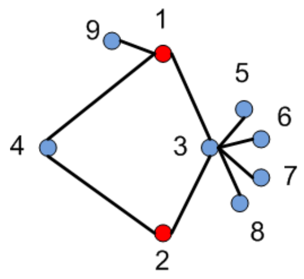


Figure 4.1: Anchor Node Selection

This means each node's estimate from INOVA follows the multivariate normal distribution which is centered at the real position with the covariance matrix Σ_{uu}. This conclusion is also valid for INOVA with nonlinear measurements and the OSE-COV algorithm.

Furthermore, we infer (see [235]) that

$$(\hat{x}_u - x_u)^T \Sigma_{uu}^{-1} (\hat{x}_u - x_u) \backsim \chi_d^2(\text{with } d \text{ degree of freedom}) \tag{4.3}$$

Therefore, the solid ellipsoid $\{x_u : (\hat{x}_u - x_u)^T \Sigma_{uu}^{-1} (\hat{x}_u - x_u) \leq \chi_d^2(\alpha)\}$ is the $100(1-\alpha)\%$ confidence region of x_u, where $\chi_d^2(\alpha)$ denotes the upper (100α)th percentile of the χ_d^2 distribution. For instance, after estimating node positions based on INOVA, Node u's estimate is $\hat{x}_u = [1, 2]^T$ and the covariance matrix is $\Sigma_{uu} = \begin{bmatrix} 1.6 & 0.25 \\ 0.25 & 1.2 \end{bmatrix}$. Then its 90% confidence region for x_u is the solid ellipsoid $\left\{x_u : \left(\begin{bmatrix} 1 \\ 2 \end{bmatrix} - x_u\right)^T \begin{bmatrix} 1.6 & 0.25 \\ 0.25 & 1.2 \end{bmatrix}^{-1} \left(\begin{bmatrix} 1 \\ 2 \end{bmatrix} - x_u\right) \leq 9.21\right\}$, since $\chi_2^2(0.01) = 9.21$.

4.2 Anchor Selection Application

Anchor nodes are defined as the nodes which have accurate position information under a global coordinate frame. Any relative measurement based localization scheme must rely on one or more anchor nodes. Not only the number of anchors, but also which nodes are selected to be anchors impact the localization accuracy of the whole network. Considering a scenario, where hundreds of nodes are set out and we need to localize them with a small number of anchors, the question is which nodes of them should be selected as anchors (we can use GPS or manual measurements to set these anchors with known positions)?

Taking Figure 4.1 for example, if Nodes 1 and 2 are already anchor nodes, obviously setting Node 4 as an anchor node is better than setting Node 9, since the node further away from the anchor nodes usually has large uncertainty, and thus more uncertainty is eliminated by setting Node 4 as an anchor. Additionally, selecting Node 3 is better than selecting Node 4, since it highly correlates to a large number of other nodes. Although the uncertainty of Node 3 is not as much as Node 4, the total uncertainty eliminated by setting Node 3 as an anchor is more (since the uncertainty of Nodes 5, 6, 7 and 8 is also reduced). Although this example provides some intuitive sense on how to select a good anchor, in large networks, anchor selection becomes much more complicated.

Without giving any quantitative uncertainty, the most straightforward strategy is to choose the node which has the max-min distance from all the anchor nodes, i.e. the node which has the maximum distance of the set of nodes' minimum distances to all the anchors is selected to be an anchor. This strategy assumes that the measurement uncertainty monotonically increases as the measured distance increases (e.g. the ToA method in [236]). For example, in Figure 4.1, assume the measured distances of node pairs are $Z = \{\zeta_{13} = 4.1, \zeta_{23} = 4, \zeta_{14} = 5.1, \zeta_{24} = 5, \zeta_{19} = 1, \zeta_{35} = 1, \zeta_{36} = 1, \zeta_{37} = 1, \zeta_{38} = 1\}$. Then the set of nodes' minimum distances to all anchors is $D_{min} = \{D_1 = 0, D_2 = 0, D_3 = 4, D_4 = 5, D_5 = D_6 = D_7 = D_8 = D_9 = 1\}$. Hence Node 4 should be selected.

When the quantitative uncertainty—the covariance matrix of all estimates—is given, the metric for selecting the best anchor node becomes selecting the node as an anchor so that the resultant trace of the covariance matrix is the smallest, that is

$$K = \mathop{\arg\min}_{i=1,2,\cdots,n} trace(\mathbf{\Sigma}_i^{New}) \tag{4.4}$$

Where $\mathbf{\Sigma}_i^{New}$ denotes the new covariance matrix after selecting Node i as an anchor node.

The trace of the covariance matrix represents the sum of mean square errors of all the node estimates. Hence the above metric satisfies the minimum mean square error(MMSE) requirement. To reduce $trace(\mathbf{\Sigma})$ as much as possible, one naive method is to select the node which has the maximum variation, i.e. selecting Node $k = \underset{i=1,2,\cdots,n}{\arg\max} trace(\Sigma_{ii})$. By using this method, the $trace(\mathbf{\Sigma})$ is reduced by at least as much as $trace(\Sigma_{kk})$. Although this method is very simple, it does not take the correlation between nodes into account. Actually, some node which is highly correlated to many other nodes should have the priority to be the anchor node, as Node 3 in Figure 4.1. In following, we give the optimal anchor selection strategy which satisfies Equation 4.4 and its proof. To select multiple anchors, we can just repeat this strategy.

Theorem 1. *If only one node can be set as an anchor, $trace(\mathbf{\Sigma}^{New})$ is minimized only when Node $K = \underset{i=1,2,\cdots,n}{\arg\max} \left[trace\left(\sum_{i=1}^{n_b} (\Sigma_{ik}\Sigma_{kk}^{-1}\Sigma_{ki}) \right) \right]$ is selected.*

Proof. To set Node k as an anchor is equivalent to get a new measurement between Node k and the global origin, and the measurement error covariance matrix $P_k = \mathbf{0}$. Assume $\mathbf{\Sigma}(t)$ and $\mathbf{\Sigma}(t+1)$ are the covariance matrices before and after selecting Node k as an anchor node, respectively. Then, $\mathbf{\Sigma}(t+1) = \mathcal{L}^{-1}(t+1) = (\mathcal{A}_b(t+1)\mathcal{P}^{-1}(t+1)\mathcal{A}_b^T(t+1))^{-1}$, where $\mathcal{A}_b(t+1) = \begin{bmatrix} \mathcal{A}_b(t) & H_k \end{bmatrix}$, $H_k = [\mathbf{0}_1, \mathbf{0}_2, \cdots, \mathbf{I}_k, \cdots, \mathbf{0}_{n_b}]^T$ and $\mathcal{P}(t+1) = \begin{bmatrix} \mathcal{P}(t) & \mathbf{0} \\ \mathbf{0} & P_k \end{bmatrix}$. Note P_k should be equal to $\mathbf{0}$, however, we first assume $P_k \to \mathbf{0}$, but $P_k \neq \mathbf{0}$, and P_k^{-1} exist. Then

$$\begin{aligned}\mathcal{L}(t+1) &= \begin{bmatrix} \mathcal{A}_b(t) & H_k \end{bmatrix} \begin{bmatrix} \mathcal{P}(t) & \mathbf{0} \\ \mathbf{0} & P_k \end{bmatrix}^{-1} \begin{bmatrix} \mathcal{A}_b^T(t) \\ H_k^T \end{bmatrix} \\ &= \mathcal{A}_b(t)\mathcal{P}^{-1}(t)\mathcal{A}_b^T(t) + H_k P_k^{-1} H_k^T = \mathbf{\Sigma}^{-1}(t) + H_k P_k^{-1} H_k^T\end{aligned}$$

By using the formula

$$(\mathbf{A} + \mathbf{B}\mathbf{D}^{-1}\mathbf{C})^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{B}(\mathbf{D} + \mathbf{C}\mathbf{A}^{-1}\mathbf{B})^{-1}\mathbf{C}\mathbf{A}^{-1}(\text{Woodbury matrix identity})$$

we obtain

$$\begin{aligned}
\mathbf{\Sigma}(t+1) &= \mathcal{L}^{-1}(t+1) \\
&= \mathbf{\Sigma}(t) - \mathbf{\Sigma}(t)H_k(P_k + H_k^T\mathbf{\Sigma}(t)H_k)^{-1}H_k^T\mathbf{\Sigma}(t) \\
&\overset{P_k=\mathbf{0}}{=} \mathbf{\Sigma}(t) - \mathbf{\Sigma}(t)H_k(H_k^T\mathbf{\Sigma}(t)H_k)^{-1}H_k^T\mathbf{\Sigma}(t) \\
&= \mathbf{\Sigma}(t) - \mathbf{\Delta}_k
\end{aligned}$$

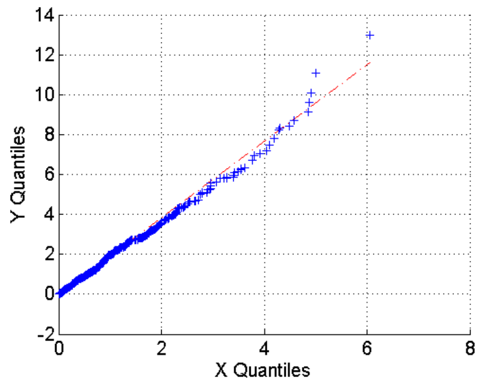
Therefore, $trace(\mathbf{\Sigma}(t+1)) = trace(\mathbf{\Sigma}(t)) - trace(\mathbf{\Delta}_k)$. To minimize $trace(\mathbf{\Sigma}(t+1))$ is equivalent to maximize
$trace(\mathbf{\Delta}_k) = trace\left(\sum_{i=1}^{n_b}(\Sigma_{ik}\Sigma_{kk}^{-1}\Sigma_{ki})\right)$. **Q.E.D.**

To calculate $trace(\mathbf{\Sigma}(t+1))$ for all k's, the time complexity is $O(d^3n^2)$, and to select the maximum $trace(\mathbf{\Sigma}(t+1))$, it takes $O(n)$ time. Therefore, the total time complexity is $O(d^3n^2)$. One benefit of this method is that it is compatible with the OSE-COV algorithm. Since each node maintains one row of the covariance matrix, i.e. $\Sigma_{ki}, i = 1, 2, \cdots, n$, and $\Sigma_{ki} = \Sigma_{ik}$. Hence, nodes can calculate $trace(\mathbf{\Delta}_k)$ locally, and then the whole network selects the node with maximum $trace(\mathbf{\Delta}_k)$ as the anchor node.

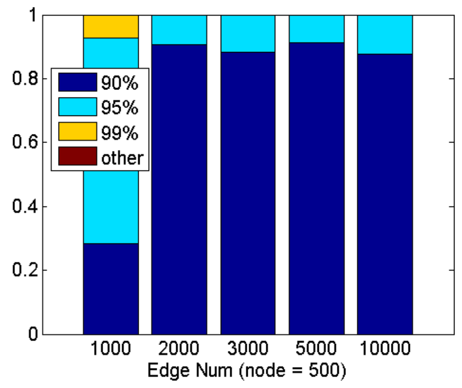
4.3 Evaluation

Based on the analysis in Section 4.1, each node's estimate $\hat{x}_u$ is a multivariate normal random variable, and $(\hat{x}_u - x_u)^T\Sigma_{uu}^{-1}(\hat{x}_u - x_u) \backsim \chi_d^2$(with d d.f.). To test the correctness, we first examine the normality of the estimates. Figure 4.2(a) shows the chi-square plot for all node's estimates in the network as shown in Figure 3.5. The chi-square plot shows the relationship between the statistic distance $(\hat{x}_u - x_u)^T\Sigma_{uu}^{-1}(\hat{x}_u - x_u)$ and the quantiles of chi-square distribution. The closer the result is to the line $y = x$, the more likely the sample x_i's are from a multivariate normal distribution. The plot in Figure 4.2(a) is very closed to $y = x$, and thus the inference of the normality is considered valid.

To show the correctness of our second inference (about the confidence region), we use



(a) Chi-square Plot for All Node's Estimates



(b) Confidence Region Accuracy

Figure 4.2: Normality Evaluation

three confidence levels–90%, 95% and 99%, and the corresponding chi-square quantiles are $\chi_2^2(0.1) = 4.61, \chi_2^2(0.05) = 5.99, \chi_2^2(0.01) = 9.21$. Based on the theorem in Section 4.1, 90% of node's real positions should fall in the ellipse $(\hat{x}_u - x_u)^T \Sigma_{uu}^{-1} (\hat{x}_u - x_u) \leq 4.61$, 95% of nodes' real positions should be within the ellipse $(\hat{x}_u - x_u)^T \Sigma_{uu}^{-1} (\hat{x}_u - x_u) \leq 5.99$, and 99% of them should be in $(\hat{x}_u - x_u)^T \Sigma_{uu}^{-1} (\hat{x}_u - x_u) \leq 9.21$. Figure 4.2(b) shows the statistical results for 5 different networks, which all have 500 nodes, but with 1000, 2000, 3000, 5000 and 10000 measurements, respectively. Except the first network (the ratio of measurement number to node number is low), all others have the consistent results with our inference. We also studied other networks of different sizes. It seems when the ratio of measurement number to node number is bigger than 2, the results are quite consistent and stable. Therefore, in practice, if the network is dense, we can use the covariance matrix to infer the confidence region accurately. Note that all the measurements are not from a multivariate normal distribution (as described with Formula 3.36). Therefore, there is no requirement on the population where the measurements are from.

For anchor selection, we evaluate the performance of the three strategies described in Section 4.2—the distance-based strategy (Dist), the variation-based strategy (Var) and the optimal strategy (Opt). The experiment shows what percentage of total error variation is reduced as the anchor nodes are set one after another by using one specific strategy. In

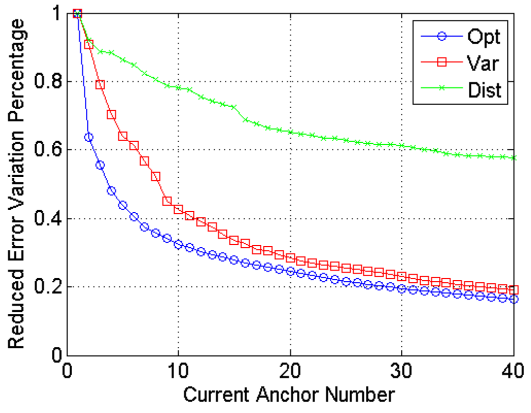


Figure 4.3: Anchor Selection Strategies Comparison

Figure 4.3, the x-axis is the current number of anchors, and the y-axis is the percentage of the current total error variation to that at the beginning. The results show that Dist performs worst, while Opt is the best. We also observe that the first few selections reduce the total error variation a lot, where the reduced error percentage of Opt is four times as much as Var. But as more and more anchors are selected, the reduction rate decreases. This is because the nodes with largest uncertainty are selected at the beginning, leaving the nodes with little uncertainty. Another observation is that as more and more nodes are set to be anchors, method Var trends to have the similar performance as Opt. This is because when multiple anchors are selected, the anchors Var selected are very likely to be similar as Opt, although the selection order may differ.

4.4 Conclusion

In this chapter, we discuss the quantitative position estimation uncertainty and its usage. In Section 4.1, by using one version of the central limit theorem, named Lindeberg's condition [235], we prove that the estimates from INOVA or OSE-COV follow multivariate normal distribution regardless of the distribution of the underlying measurements. In Section 4.2, we show how to use the quantitative estimation uncertainty to select anchors optimally. The basic idea is that to always select the node as an anchor so that the resultant trace of the network's covariance matrix is the smallest.

In the evaluation section, we use a Chi-square Plot to verify the normality of the position estimation, and also validate the accuracy of using the normality to infer the confidential region. For anchor node selection, we compare three different strategies–the distance-based strategy (Dist), the variation-based strategy (Var) and the optimal strategy (Opt). The Opt strategy outperforms the other two.

Chapter 5

Mobile Sensor Network Localization

In Chapter 3, an efficient localization algorithm for a stationary sensor network is proposed. This algorithm models the positioning measurements of a stationary sensor network as a measurement graph, and then uses an electrical network analogy (INOVA) to efficiently and optimally solve the localization problem. It supports both linear and non-linear positioning measurements, and has a decentralized version (OSE-COV). Although this algorithm supports nodes joining and leaving a network at run time, the form and the model is essentially for stationary sensor networks only.

In practice, many applications require location information for mobile agents, such as firefighter/pedestrian indoor localization, vehicle network localization, and underground mining robot localization. For most outdoor location based applications, GPS is reliable and accurate enough (except some application requires high frequency position updates). But for areas that GPS satellites do not cover, there is no generic solution existing currently. In these non-GPS applicable scenarios, relative measurements between mobile or stationary agents, are usually used for localization. Such a method in the robotics field is called collaborative localization (CL).

Since CL may involve a large number of mobile agents, and is implemented in portable devices, an ideal CL approach should satisfy the following three requirements: i) fully decentralized; ii) efficient and scalable; iii) consistent. For CL, a decentralized solution is favored because not only can a decentralized solution always be implemented in a centralized manner, but also in many scenarios a centralized infrastructure is not proper or even possible. Therefore, the first requirement indicates that there should be no centralized party involved, and the full communication connection between agents should not be assumed. The second requirement means that the algorithm should be efficient in terms of computational complexity, memory usage and communication cost, and it should be scalable as the number of participants increase. The third requirement suggests that it should avoid a well known over-confidence problem, where the calculated uncertainty does not reflect the real uncertainty (usually is over optimistic). However, due to the lack of global and future knowledge, the three requirements cannot be simultaneously satisfied. Many existing CL methods satisfy Requirements i and ii, but not iii [187,188], or Requirements ii and iii, but not i [78,184].

In this chapter, we propose a novel CL algorithm, named Elastic Decentralized Collaborative Localization (EDCL). EDCL satisfies Requirement i, and enables a trade-off adjustment between Requirements ii and iii, which is useful in practice. The main idea of CL is to introduce a parameter Q, named marginalization factor, which indicates the number of historical measurements allowed to be preserved in memory. By using the marginalization factor Q, we are able to control how much resource (in terms of computation complexity, memory usage and network bandwidth consumption) EDCL uses, as well as the consistency we can achieve. An extreme case is that when $Q = 1$, EDCL only maintains self-state and has no difference with the weighted average method; and when Q is large enough, EDCL can achieve the optimal results.

The contributions of this work are the following:

- EDCL is the first CL algorithm which is fully decentralized, and the efficiency and consistency are controllable and adjustable.

- An efficient implementation of EDCL is achieved through the information relation graph model which is similar with the measurement graph in 3.1.1 and a chain-based data structure. By using this model and the data structure, EDCL is able to process out-of-order measurements, and reduce the communication overhead to a low level.

- EDCL is evaluated through a complicated emulation on 1,256 taxis' real GPS trajectories in Beijing. The localization accuracy, the system overhead and the parameter influences are studied. The results show that the accuracy of EDCL is close to optimal and the overhead is low.

5.1 System Formalization and Challenges

In this section, the mobile localization problem is formalized, and the benefits and the challenges of the CL algorithm are then discussed. For simplicity, we only consider the case where all agents are mobile. In reality, it is possible that the whole system is mixed of both stationary and mobile agents. Our model is still valid for this case by treating the displacements of these stationary nodes as $\mathbf{0}$ over time.

5.1.1 System Formalization

Suppose there are n (maybe unknown in practice) mobile agents in the system, and their Ids are $N = \{1, 2, \cdots, n\}$. Their position states at time step k are $\mathbf{X}_k = \{\mathbf{x}_{i,k} | i \in N\}$, where $\mathbf{x}_{i,k} = [x_{i,k}, y_{i,k}]^T$ in 2D space. The communication range of an agent is r_c, and the inter-agent measurement range is r_m. We make the following assumptions:

- The belief of each agent's initial position is known. Written as $bel(\mathbf{x}_{i,0}) = (\hat{\mathbf{x}}_{i,0}, \mathbf{P}_{i,i,0})$, $i = 1, 2, \cdots, n$, where $\hat{\mathbf{x}}_{i,0}$ and $\mathbf{P}_{i,i,0}$ are the position estimation and error covariance matrix at time 0, respectively.

- Each mobile agent has a DR module, which measures the relative position (both angle and distance) between two successive time points, represented as

$$\mathbf{u}_{i,k} = f(\mathbf{x}_{i,k-1}, \mathbf{x}_{i,k}, \mathbf{w}_{i,k}) = \mathbf{x}_{i,k} - \mathbf{x}_{i,k-1} + \mathbf{w}_{i,k} \tag{5.1}$$

  where $\mathbf{u}_{i,k}$ is the noisy DR measurement, $\mathbf{x}_{i,k-1}$ and $\mathbf{x}_{i,k}$ are Agent i's positions at time k and $k-1$ respectively, and $\mathbf{w}_{i,k}$ is the measurement noise. Additionally, we use $\mathbf{U}_k$ to represent all DR measurements obtained at time k.

- When two agents are in the range of r_m, both of them measure the range between them. The form of a range could be relative distance, relative angle or both. Formally, the inter-agent measurement is represented as:

$$\mathbf{v}_{i,j,k} = g(\mathbf{x}_{i,k}, \mathbf{x}_{j,k}, \mathbf{e}_{i,j,k}) \tag{5.2}$$

  where $\mathbf{v}_{i,j,k}$ is the noisy inter-agent measurement, g is the measurement function, $\mathbf{x}_{i,k}$ and $\mathbf{x}_{j,k}$ are the positions of Agent i and j at time k respectively, and $\mathbf{e}_{i,j,k}$ is the measurement noise. Additionally, we use $\mathbf{V}_k$ to denote all the inter-agent measurements obtained at time k.

- Some of the agents may have GPS measurements:

$$\mathbf{c}_{i,k} = gps(\mathbf{x}_{i,k}, \beta_{i,k}) = \mathbf{x}_{i,k} + \beta_{i,k} \tag{5.3}$$

  where $\mathbf{c}_{i,k}$ is the noisy GPS measurement and $\beta_{i,k}$ is the measurement noise. Similarly, we use $\mathbf{C}_k$ to denote all the GPS measurements at time k.

- When two agents are in the range of r_c, they communicate with each other. We assume that $r_c \geq r_m$, which is often the truth in reality.

As agents move and obtain these measurements, from a centralized perspective, the goal of a localization algorithm is to estimate each agent's position at each time step: $bel(\mathbf{X_k}) =$

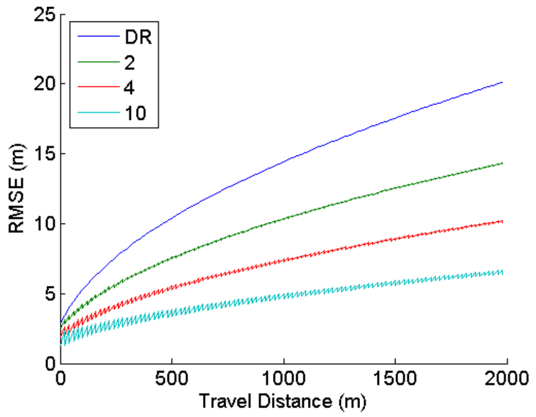


Figure 5.1: The root mean square errors (RMSE) of DR and CL with different number of agents.

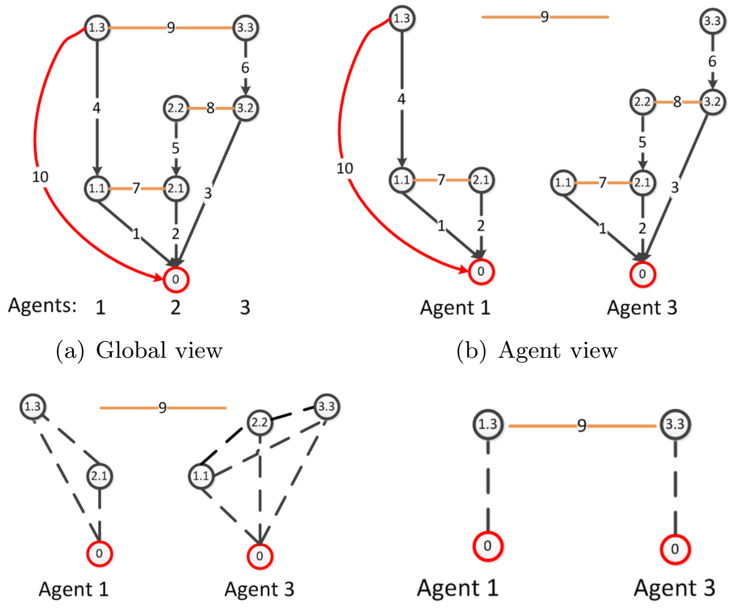
$p(\mathbf{X_k}|bel(\mathbf{X_0}), \mathbf{U}_{1:k}, \mathbf{V}_{1:k}, \mathbf{C}_{1:k})$, which is often represented by its first two moments $bel(\mathbf{X_k}) = (\hat{\mathbf{x}}_k, \mathbf{P}_k)$, where $\hat{\mathbf{x}}_k = [\hat{\mathbf{x}}_1^T, \hat{\mathbf{x}}_2^T, \cdots, \hat{\mathbf{x}}_n^T]^T$ and $\mathbf{P}_k = Cov(\hat{\mathbf{x}}_k \hat{\mathbf{x}}_k^T)$. However, we focus on a more challenging problem, i.e. estimating each agent's position in a fully decentralized way.

5.1.2 The Benefits and Challenges of CL

The benefits of CL is obvious. Since agents share information with each other, as more agents participate in the system, the error growth rate is suppressed. By contrast, if an agent only uses its own DR module, the error accumulates quickly. Figure 5.1 is a simple simulation result of the comparison of DR and CL. We see that the mean error of DR is independent of the agent number. After traveling 2km, the agent has a mean error of 20m. For CL, as the number of agents increases, the error is more bounded. When the number of agents reaches 10, the mean error is only 6.4m.

However, to implement CL in a decentralized way is challenging. This is because as agents collaborate, they correlate with each other. However, since the memory of a device is limited, an agent has to marginalize away (fuse) some of the historical joint states, which causes some of the correlations cannot be updated correctly. Therefore, the key challenges for a decentralized algorithm are: *when to fuse the measurements and how to update when two agents meet?*

Figure 5.2 illustrates the marginalization problem. The horizontal axis is agent identity,



(a) Global view (b) Agent view

(c) Maintain latest states (d) Maintain self-state

Figure 5.2: Measurement graphs from different views at time step 3. Three agents participate in the system. Each node represents an agent state at some time step. Node Id is represented as {Agent Id}.{Time stamp}. Node 0 is the global origin. Black arrowed lines represent DR measurements, orange lines represent inter-agent measurements (usually range measurements), red lines represent GPS measurements, and black dotted lines represent the fused covariance between nodes. (a) and (b) are the measurement graphs from global view and local view, respectively. (c) is the graph after marginalizing all previous states but the latest ones, while (d) only keeps the latest self-state.

and the vertical axis is the time step. Figure 5.2(a) is the global view of the measurement graph at time step 3 when Agent 1 meets Agent 3. If Agents 1 and 3 preserve all the measurements without marginalizing away any historical states as Figure 5.2(b) shows, they can rebuild the graph like Figure 5.2(a) by exchanging measurements, and thus have a consistent estimation. However, if agents fuse measurements too quickly, they may encounter the *correlation structure conflict* problem. There are two types of conflicts: explicit and implicit conflicts.

For two beliefs, say $bel(\mathbf{x}_1) = (\hat{\mathbf{x}}_1, \mathbf{P}_{1,1})$ and $bel(\mathbf{x}_2) = (\hat{\mathbf{x}}_2, \mathbf{P}_{2,2})$, if they are not correlated with each other, then we can combine them consistently as

$$bel(\mathbf{x}_1, \mathbf{x}_1) = \left([\mathbf{x}_1^T, \mathbf{x}_2^T]^T, \begin{bmatrix} \mathbf{P}_{1,1} & \mathbf{0} \\ \mathbf{0} & \mathbf{P}_{2,2} \end{bmatrix} \right) \tag{5.4}$$

However, if agents marginalize their historical states in an improper way, it may cause explicit conflicts, which results that an agent cannot update its measurement graph when it met another agent. In Figure 5.2(c), each agent only keeps the latest states of all agents it met and marginalizes away all historical states. At time step 3, Agent 1 has the belief of $bel(\mathbf{x}_{1.3}, \mathbf{x}_{2.1})$ and Agent 3 has the belief of $bel(\mathbf{x}_{1.1}, \mathbf{x}_{2.2}, \mathbf{x}_{3.3})$. They cannot merge consistently, since the state $\mathbf{x}_{1.1}$ is correlated to $\mathbf{x}_{1.3}$, $\mathbf{x}_{2.1}$ is correlated to $\mathbf{x}_{2.2}$, and the correlation information is missing. Explicit conflicts are detectable, since both Agent 1 and Agent 3 know the states 2.1 and 2.2 are correlated, but the correlation information is missing.

Another case is the implicit conflict where agents seem to be able to merge their measurement graphs, but there are unaware correlations exists, and the result will be inconsistent if they apply the merging. In Figure 5.2(d), each agent only keeps the state of itself, i.e. Agent 1 has $bel(\mathbf{x}_{1.3})$ and Agent 3 has $bel(\mathbf{x}_{3.3})$. It seems that they can merge their beliefs as Equation 5.4 does. However, $bel(\mathbf{x}_{1.3})$ and $bel(\mathbf{x}_{3.3})$ are actually correlated with each other, and agents are not aware of it. The harm of the implicit conflict is to cause the *over-confidence problem* [237], i.e. the resultant uncertainty (represented as the covariance matrix) is "smaller" than the actual one. An extreme case is that two agents become stationary and communicate with each other for a long time. They keep updating their states by averaging their estimates although there is no new information other than the measurements between them. The error covariance matrix $\mathbf{P}$ keeps decreasing exponentially, and finally is close to $\mathbf{0}$). Implicit conflicts are not detectable, since both Agent 1 and Agent 3 cannot tell they are correlated.

5.2 EDCL Overview and Extended Measurement Graph

EDCL is a fully decentralized collaborative algorithm, which relies on the model *information relation graph*. Before presenting the information relation graph model, we first propose the model–*extended measurement graph* (EMG) (see Figure 5.3), which is the basics of the information relation graph. The purpose of EDCL is to reconstruct the local EMG to be as

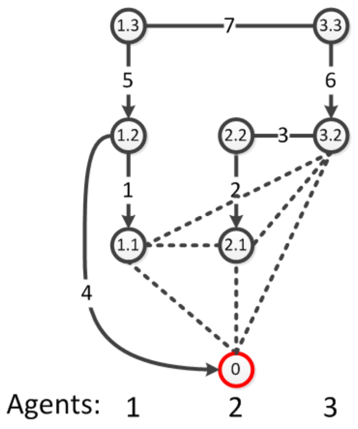


Figure 5.3: Extended Measurement graph. Each vertex represents a space-time state. Vertex 0 is the global origin. All solid lines are measurements, where the arrowed solid lines are strong measurements (GPS or relative position) and the non-arrowed solid lines are weak measurements (range or relative angle measurements). The dotted lines represent the covariance information, where the dotted line connecting a state and the origin represents self covariance $Cov(\mathbf{x}_{i,k}, \mathbf{x}_{i,k})$ (the diagonal elements of the covariance matrix), and the dotted line connecting states represents covariance of these two states $Cov(\mathbf{x}_{i,k}, \mathbf{x}_{j,l})$ (the off-diagonal elements of the covariance matrix). The vertexes connected to the global origin with dotted lines are the base nodes, which represent the base belief, i.e. $bel(\mathbf{x}_B) = (\hat{\mathbf{x}}_B, Cov(\hat{\mathbf{x}}_B, \hat{\mathbf{x}}_B))$. Here $\mathbf{x}_B = [\mathbf{x}_{1.1}^T, \mathbf{x}_{2.1}^T, \mathbf{x}_{3.2}^T]^T$

close to the global EMG as possible through collaboration. When an agent travels alone, it adds DR and GPS measurements (if it has any) onto its EMG, and updates its position estimate; when it encounters another agent, the two agents measure their relative position, share their positioning information, and update their EMGs and estimates accordingly; when the EMG grows too large, the agent fuses away the historical states.

In EDCL, a state is defined as a position of an agent at a certain time step, denoted as $\mathbf{x}_{i,k} = [x_{i,k}, y_{i,k}]$ (Agent i's state at Time k in 2D space). A measurement at time k is written in a uniform way as $\mathbf{z}_k = h_k(\mathbf{x}_{i,k}, \mathbf{x}_{j,l}) + \epsilon_k$, which could represent a DR measurement ($i = j, l = k-1$), an inter-agent measurement ($k = l$), or a GPS measurement ($\mathbf{x}_{j,l} = 0$). All these states and measurements form an extended measurement graph, which is an extension of the measurement graph defined in Section 3.1.1. The formal definition of an extended measurement graph is as follows:

Definition 1 (Extended Measurement Graph)**.** *An extended measurement graph is a directed graph $G = (V, E, F, B)$, where V is the vertex set representing all interested states, E is the edge set including all state pairs who have measurements, $F : E \to Z$ is an edge* *function which maps each edge to its corresponding measurement* $\mathbf{z}_k \in Z$*, and* B *is the base belief about the subset of states of* V*, written as* $bel(\mathbf{X}_B) = (\hat{\mathbf{x}}_B, Cov(\hat{\mathbf{x}}_B, \hat{\mathbf{x}}_B))$*.*

In EDCL, the nodes (states) involved in **B** are called *Base Nodes* (or *Base*), and are always the oldest states of all different agents in an EMG. Each state's identity is defined by {Agent Id}.{Time stamp} for simplicity. Figure 5.3 is a drawing of an extended measurement graph. Based on the belief of the base nodes $bel(\mathbf{X}_B)$ and the measurements in an EMG, we can estimate all the states by using the INOVA algorithm in Section 3.2.3.1.

Although the procedure of EDCL seems straightforward, there are several practical issues that need to be solved: i) What kind of information should be stored in each agent's local memory and what is the data structure? ii) In detail, how do two agents exchange information and update when they meet? iii) When to fuse away historical states? iv) How to resolve the structural conflict when two agents merge their EMGs? All these issues are addressed in the following sections in detail.

5.3 Information Relation Graph and Its Operations

The extended measurement graph model is easy to visualize, supports non-linear measurements, and has low complexity on updates. However, the disadvantages of EMG are: i) For a non-linear measurement update, it requires the estimation of the involved nodes (see Equations 3.21 and 3.22). However, the computational complexity of the estimation is nontrivial ($O(n^3)$, where n is the number of the states). ii) If using EMG, when two agents meet, they need to exchange not only the measurements, but also the estimation and covariance matrix, which causes a plenty of communication costs.

To overcome these drawbacks, the information relation graph model is proposed. The idea is to encode the estimation information into the measurements so that two agents do not need to redo the estimation for update when they meet, and the cost of exchanging information is much lower than using EMG. In the following, the update, augmentation and marginalization operations of an EMG are first reviewed, and then the information relation graph model which applies this idea is formally defined.

5.3.1 Update, Augmentation and Marginalization

The INOVA algorithm in 3.2.3.1 stores two pieces of information in order to calculate the position estimation and the covariance matrix. They are the Kirchhoff matrix (or information matrix) $\mathbf{L}$ and the Kirchhoff vector (or information vector) $\mathbf{b}$. They have the following relationship with the position estimation $\hat{\mathbf{x}}$ and the covariance matrix $\mathbf{\Sigma}$:

$$\mathbf{L} = \mathbf{P}^{-1}, \mathbf{b} = \mathbf{L}\hat{\mathbf{x}} \tag{5.5}$$

Assume we have the following Kirchhoff vector and matrix at some time step:

$$\mathbf{b}^{-} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \end{bmatrix}, \mathbf{L}^{-} = \begin{bmatrix} \mathbf{L}_{11} & \mathbf{L}_{12} \\ \mathbf{L}_{21} & \mathbf{L}_{22} \end{bmatrix}, \tag{5.6}$$

A measurement $\mathbf{z} = h(\mathbf{x}_2) + \epsilon$ about state $\mathbf{x}_2$ is obtained, whose error covariance matrix is $Cov(\epsilon\epsilon^T) = \mathbf{R}$. Then the *update* operation for the Kirchhoff vector and matrix is:

$$\begin{aligned} \mathbf{b}^{+} &= \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 + \mathbf{i} \end{bmatrix}, \\ \mathbf{L}^{+} &= \begin{bmatrix} \mathbf{L}_{11} & \mathbf{L}_{12} \\ \mathbf{L}_{21} & \mathbf{L}_{22} + \mathbf{I} \end{bmatrix} \end{aligned} \tag{5.7}$$

$\mathbf{i}$ and $\mathbf{I}$ are given by:

$$\begin{aligned} \mathbf{i} &= \mathbf{H}^T\mathbf{R}^{-1}(\mathbf{z} - h(\hat{\mathbf{x}}_2^-) + \mathbf{H}\hat{\mathbf{x}}_2^-) \\ \mathbf{I} &= \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H} \end{aligned} \tag{5.8}$$

where $\mathbf{H} = \frac{\partial h}{\partial \mathbf{x}_2}|_{\hat{\mathbf{x}}_2^-}$, and $\hat{\mathbf{x}}_2^-$ is the estimate of the involved joint state right before the measurement $\mathbf{z}$. Note that all the subscripts above represent the block indexes, which means $\mathbf{x}_2$ may represent more than one state. For example, if $\mathbf{z}$ is a GPS measurement, $\mathbf{x}_2$ only represents one state, but if $\mathbf{z}$ is an inter-agent measurement, $\mathbf{x}_2$ represents two states.

When an agent moves forward, DR measurements are obtained, which always connect a new state to an old state. For this type of update, it needs the *Augmentation* operation, which has the following two sub-steps. First, a new zero state is added:

$$\mathbf{b} = \left[\begin{array}{c} \mathbf{b}_1 \\ \hline \mathbf{b}_2 \\ \mathbf{0} \end{array}\right], \mathbf{L} = \left[\begin{array}{c|cc} \mathbf{L}_{11} & \mathbf{L}_{12} & \mathbf{0} \\ \hline \mathbf{L}_{21} & \mathbf{L}_{22} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{array}\right] \tag{5.9}$$

Then the Equation 5.7 is applied to the connected old state and the new state.

$$\mathbf{b}^+ = \left[\begin{array}{c} \mathbf{b}_1 \\ \hline \mathbf{b}_{2,3} + \mathbf{i}_{2,3} \end{array}\right], \mathbf{L}^+ = \left[\begin{array}{c|c} \mathbf{L}_{11} & \mathbf{L}_{12} \quad \mathbf{0} \\ \hline \begin{array}{c} \mathbf{L}_{21} \\ \mathbf{0} \end{array} & \mathbf{L}_{2,3} + \mathbf{I}_{2,3} \end{array}\right] \tag{5.10}$$

where $\mathbf{b}_{2,3} = [\mathbf{b}_2^T, \mathbf{0}^T]^T$, $\mathbf{L}_{2,3} = \begin{bmatrix} \mathbf{L}_{22} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}$, and $\mathbf{i}_{2,3}$ and $\mathbf{I}_{2,3}$ are calculated by Equation 5.8.

Another important operation is the *marginalization* operation. It is used to reduce the dimension of the joint state by forgetting uninteresting states. After marginalization, only the interesting states are left, and their sate estimates and the covariance matrix stay the same. From the probability perspective, it is the same as $P(x_1) = \int f(x_1, x_2)dx_2$. Taking Equation 5.6 for example, if State 2 is to be dropped, the marginalization operation is written as:

$$\begin{aligned} \mathbf{b}_1^m &= \mathbf{b}_1 - \mathbf{L}_{12}\mathbf{L}_{22}^{-1}\mathbf{b}_2 \\ \mathbf{L}_{11}^m &= \mathbf{L}_{11} - \mathbf{L}_{12}\mathbf{L}_{22}^{-1}\mathbf{L}_{12}^T \end{aligned} \tag{5.11}$$

5.3.2 Information Relation Graph

As aforementioned, to overcome the drawbacks of the EMG model, we need to encode the estimation information into the measurements. Therefore, instead of preserving the original measurements, the $\mathbf{i}_k$ and $\mathbf{I}_k$ in Equation 5.8 are stored, since $\mathbf{i}_k$ and $\mathbf{I}_k$ already have the estimation information. To define the information relation graph, we first define the information relation as below:

Definition 2 (Information Relation)**.** *An information relation is a four-tuple:* $\zeta_{ij}(k) = (i, j, \mathbf{i}_k, \mathbf{I}_k)$*, where* i *and* j *are the state Ids,* $\mathbf{i}_k = \mathbf{H}_k\mathbf{R}_k^{-1}(\mathbf{z}_k - h_k(\hat{\mathbf{x}}) + \mathbf{H}_k\hat{\mathbf{x}})$*,* $\mathbf{I}_k = \mathbf{H}_k\mathbf{R}_k^{-1}\mathbf{H}_k^T$*, and* $\hat{\mathbf{x}}$ *is the involved state estimates.*

In practice, the most common measurement functions have the Jacobian Matrices of the form $\mathbf{H}_k = [\mathbf{h}_k^T, -\mathbf{h}_k^T]$ (see Table 3.1), and thus $\mathbf{i}_k$ and $\mathbf{I}_k$ have the structure $\mathbf{i}_k = [\psi_k, -\psi_k]^T$ and $\mathbf{I}_k = \begin{bmatrix} \Psi_k & -\Psi_k \\ -\Psi_k & \Psi_k \end{bmatrix}$ (for GPS, $\mathbf{i}_k = \psi_k$ and $\mathbf{I}_k = \Psi_k$), where $\psi_k = \mathbf{h}_k\mathbf{R}_k^{-1}(\mathbf{z}_k - h_k(\hat{\mathbf{x}}) + \mathbf{H}_k\hat{\mathbf{x}})$ and $\Psi_k = \mathbf{h}_k\mathbf{R}_k^{-1}\mathbf{h}_k^T$. For the symmetric structure, we can further use the *compact information relation* $\zeta_{ij}(k) = (i, j, \psi_k, \Psi_k)$ instead. If not specifically pointed out, the following text uses the term information relation to refer to the compact information relation for simplicity.

After defining the information relation, it is obvious to define the information relation graph (IRG). By replacing measurements with information relations, the measurement graph is transformed to an information relation graph. All EDCL operations are actually based on IRG. The formal definition of an information relation graph is as follows:

Definition 3 (Information Relation Graph)**.** *An information relation graph is a directed graph* $\Phi = (V, E, F, B)$*, where* V *is the vertex set representing all interested states,* E *is the edge set including all state pairs who have information relations,* $F : E \to \Upsilon$ *is an edge function which maps each edge to its corresponding information relation* $\zeta_k \in \Upsilon$*, and* B *is the base belief about the subset of states of* V*, written as* $bel(\mathbf{X}_B) = (\hat{\mathbf{x}}_B, Cov(\hat{\mathbf{x}}_B, \hat{\mathbf{x}}_B))$*.*

5.4 The Elastic Decentralized Collaborative Localization Algorithm

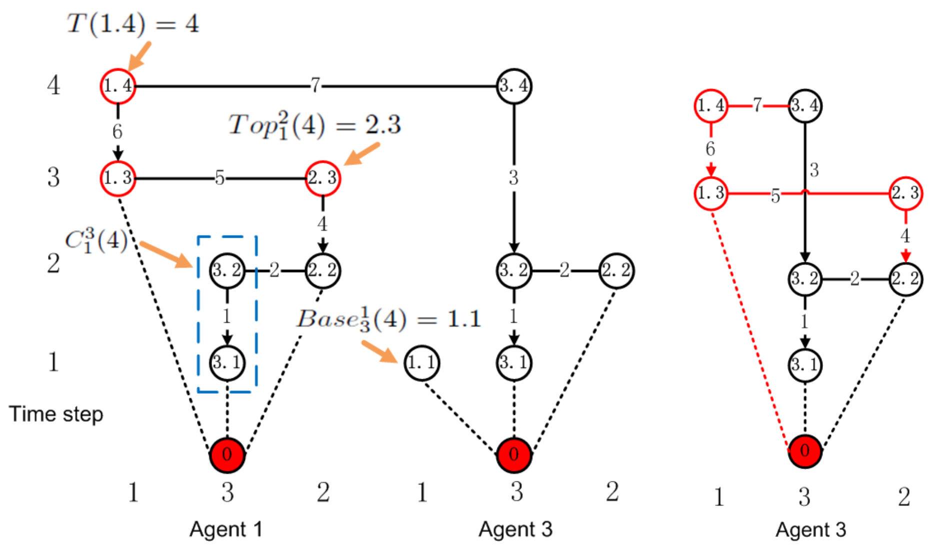
In EDCL, each agent stores five elements in its memory: the agent Id, the information relation graph $\Phi_i(k)$, the state Id list of the IRG $L_i(k)$, the marginalization factor Q, and the current belief about its latest state $bel(\mathbf{x}_i(k)) = (\hat{\mathbf{x}}_i(k), \mathbf{P}_i(k))$. The IRG includes two parts, $\Phi_i(k) = (\Upsilon_i(k), bel(\mathbf{x}_B))$, where $\Upsilon_i(k)$ is the information relation set, and $bel(\mathbf{x}_B) = (\mathbf{b}_B, \mathbf{L}_B)$ is the belief of the base nodes. The state Id list is organized in chains. The state Ids about the same agent group together, and are sorted in time order within the chain (recall that the state Id is denoted as {agent Id}.{time stamp}). The marginalization factor Q indicates when the agent needs the marginalization operation. According to different types of measurements, EDCL has three update operations: the DR, the GPS and the inter-agent measurement updates. The following subsections explain each operation in detail.

5.4.1 DR and GPS Measurement Update

When an agent obtains an egocentric measurement, such as a GPS or DR measurement, it processes it locally. Three actions take place: the new information relation is added to the IRG, the state Id list is modified, and the latest self state (position) belief is updated.

The obtained measurement is first transformed to the information relation based on the *compact information relation* definition and Table 3.1. Then this information relation is appended to the information relation set $\Upsilon_i(k)$. For a DR measurement, a new state is added into the state Id list $L_i(k)$, and for a GPS measurement, $L_i(k)$ does not change since no new state is generated. Assume that the belief of the self-state before the measurement is $bel(\mathbf{x}_i^-(k)) = (\hat{\mathbf{x}}_i^-(k), \mathbf{P}_i^-(k))$. For a DR measurement $\mathbf{z}_i(k)$ with the error covariance matrix $\mathbf{R}_i(k)$, the self-state belief is updated to $bel(\mathbf{x}_i^+(k))$:

$$\begin{aligned} \hat{\mathbf{x}}_i^+ &= \hat{\mathbf{x}}_i^-(k) + \mathbf{z}_i(k) \\ \hat{\mathbf{P}}_i^+ &= \mathbf{P}_i^-(k) + \mathbf{R}_i(k) \end{aligned} \tag{5.12}$$



(a) Before merging (b) After merging

Figure 5.4: This figure explains different notations and the merging process when Agents 1 and 3 encounter. The red circles and lines represent the new information obtained by Agent 3. (a) shows the local IRGs of Agents 1 and 3 before merging. (b) is the resultant IRG of Agent 3 after merging.

If the measurement is a GPS measurement $(\mathbf{z}_i(k), \mathbf{R}_i(k))$, the new self-state belief $bel(\mathbf{x}_i^+(k))$ is:

$$\begin{aligned} \hat{\mathbf{x}}_i^+(k) &= \mathbf{P}_i^-(k)(\mathbf{P}_i^-(k) + \mathbf{R}_i(k))^{-1}\mathbf{z}_i(k) \\ &\quad + \mathbf{R}_i(k)(\mathbf{P}_i^-(k) + \mathbf{R}_i(k))^{-1}\hat{\mathbf{x}}_i^-(k) \\ \hat{\mathbf{P}}_i^+(k) &= \mathbf{P}_i^-(k)\mathbf{R}_i(k)(\mathbf{P}_i^-(k) + \mathbf{R}_i(k))^{-1} \end{aligned} \tag{5.13}$$

5.4.2 Inter-agent Measurement Update

When two agents encounter each other, three things occur: i) two agents obtain the inter-agent measurement between them; ii) their positioning information is shared with each other; iii) and the data structure in memory is updated accordingly.

In practice, the inter-agent measuring range r_m is usually smaller than the communication range r_c. The above updates only occur when they obtain the inter-agent measurement. When the inter-agent measurement is obtained, it is first transformed into the information relation using Equation 5.8, and then appended onto the information relation set $\Upsilon(k)$.

After obtaining the inter-agent measurement, the two agents need to exchange their positioning information for collaboration. To simplify the explanation, we introduce the following notations. In Agent i's IRG $\Phi_i(k)$, the state Ids $L_i(k)$ are grouped by agent Id. Let $C_i^j(k)$ denote the state Id chain of Agent j in Agent i's IRG $\Phi_i(k)$. The state Ids in a chain is sorted in time order. We use $Base_i^j(k)$ to denote the oldest state Id in $C_i^j(k)$, which is always a base node. Similarly, we use $Top_i^j(k)$ to denote the latest state Id of $C_i^j(k)$. Since each state Id has the form of {Agent Id}.{Time stamp}, we use $T(Id)$ to denote the time stamp of the state. For example, in Agents 1's and 3's IRGs at time step 4 as shown in Figure 5.4(a), $C_1^3(4) = \{3.1, 3.2\}$, $Base_3^1(4) = 1.1$, $Top_1^2(4) = 2.3$ and $T(1.4) = 4$.

To reduce the communication cost, each agent should only send the necessary information to the other, which requires the sender to know the IRG structure of the receiver. Therefore, the exchanging process has two steps: one is to exchange the IRG meta-data, and the other is to send the real information relations.

An IRG represents the knowledge the agent has about the other agents' time space states. Furthermore, an IRG is grouped by chains, and each chain represents the knowledge of one agent. Therefore, when two agents meet, they need to know the top and the base of each chain from the other. The top represents how new the information the agent has, and the base represents the degree of the marginalization. All states between base and top have the original information relations (not marginalized). Hence, the IRG meta-data is defined as a top-base set for all chains. Take Figure 5.4(a) for example. At Time 4, Agent 3's IRG meta-data is $\{(1.1, 1.1), (2.2, 2.2), (3.1, 3.4)\}$.

After receiving the meta-data, the sender compares it with its own chains, and figures out which information relations and base node beliefs to send. Assuming the sender Id is s and the receiver Id is r, there are four cases for Chain j:

- $T(Top_s^j(k)) \leq T(Top_r^j(k))$: the sender has no new information for the receiver, and thus no action needs to be taken for receiver about Chain j.
- $T(Top_s^j(k)) > T(Top_r^j(k))$ and $T(Base_s^j(k)) \leq T(Top_r^j(k))$: the sender has new informa-

tion about Chain j and all the new information has origin information relations. Therefore, the sender needs to send all information relations whose states' time stamps are later than $T(Top_r^j(k))$.

- $T(Base_s^j(k)) > T(Top_r^j(k))$: this indicates that the sender has new information about Chain j, but it does not have the original information relations to connect to the receiver's latest state of Chain j since it already marginalizes away them. Therefore, it is an explicit structural conflict. In this case, the sender needs to send the whole chain, i.e. the base belief $bel(\mathbf{x}_B^j)$ and all the information relations of Chain j. Note that if there are multiple chains have this case, the joint base belief $bel(\mathbf{x}_B^{i,j})$ should be sent.

- The receiver has Chain j, but not the sender: this indicates that the sender does not have any knowledge about Agent j, or it fully marginalized the whole chain before (the implicit conflict case). In this case, the receiver needs to send the whole chain.

Take Figure 5.4(a) for example. After receiving Agent 3's IRG meta-data, Agent 1 compares the IRG structures chain by chain. For Chain 1, Agent 1 finds that it falls into Case 3, and the new states are 1.3 and 1.4. Hence, it puts the whole chain, i.e. information relations 5 and 6, and the belief about State 1.3 into the sending set for sending. For Chain 2, it falls into Case 2, and thus information relations 4 and 5 are put into (union) the sending set. For Chain 3, it falls into Case 1, so no new information needs to be sent.

After receiving the new information, the agent needs to update its in memory data structure. Three elements are changed, the IRG, the state Id list and the self-state estimate. For the IRG, besides adding the new information relations, the base belief is also updated if there is new base information. Take above example. Assuming before meeting, Agent 3's base belief is $bel(\mathbf{x}_{1.1}, \mathbf{x}_{2.2}, \mathbf{x}_{3.1}) = (\hat{\mathbf{x}}^-, \mathbf{P}^-)$, where

$$\hat{\mathbf{x}}^- = [\hat{\mathbf{x}}_{1.1}^T, \hat{\mathbf{x}}_{2.2}^T, \hat{\mathbf{x}}_{3.1}^T]^T, \mathbf{P}^- = \begin{bmatrix} \mathbf{P}_{1.1,1.1} & \mathbf{P}_{1.1,2.2} & \mathbf{P}_{1.1,3.1} \\ \mathbf{P}_{2.2,1.1} & \mathbf{P}_{2.2,2.2} & \mathbf{P}_{2.2,3.1} \\ \mathbf{P}_{3.1,1.1} & \mathbf{P}_{3.1,2.2} & \mathbf{P}_{3.1,3.1} \end{bmatrix}$$

The base information it obtains is $bel(\mathbf{x}_{1.3}) = (\hat{\mathbf{x}}_{1.3}, \mathbf{P}_{1.3,1.3})$. After merging, the resultant base belief is $bel(\mathbf{x}_{1.3}, \mathbf{x}_{2.2}, \mathbf{x}_{3.1}) = (\hat{\mathbf{x}}_B^+, \mathbf{P}_B^+)$, where

$$\hat{\mathbf{x}}_B^+ = [\hat{\mathbf{x}}_{1.3}^T, \hat{\mathbf{x}}_{2.2}^T, \hat{\mathbf{x}}_{3.1}^T]^T, \mathbf{P}_B^+ = \begin{bmatrix} \mathbf{P}_{1.3,1.3} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{P}_{2.2,2.2} & \mathbf{P}_{2.2,3.1} \\ \mathbf{0} & \mathbf{P}_{3.1,2.2} & \mathbf{P}_{3.1,3.1} \end{bmatrix}$$

Note that the explicit conflict is resolved by treating the covariance between the new bases and the old bases as $\mathbf{0}$, so the correlation information is lost. Furthermore, for the explicit conflict case (Case 3), all its old information relations is removed.

After updating the IRG, the state Id list $\Upsilon_i(k)$ is modified accordingly. For the self-state estimate $bel(\mathbf{x}_i)$, based on the new IRG and Equations 5.5–5.10, it is also updated. The before and after meeting IRGs of Agent 3 are shown in Figures 5.4(a) and 5.4(b), respectively.

5.4.3 Marginalization

In EDCL, as agents collaborate over time, the IRG keeps growing without bound. Therefore, we need to marginalize some of the past states in order to prevent EDCL from exhausting memory and computation resources. But the questions are: i) when should the marginalization happen? ii) And which states should be marginalized?

In EDCL, each agent keeps a parameter, called marginalization factor Q, which is a threshold to indicate when the marginalization needs to happen. The metric for Q is flexible. It could be the current memory usage, the number of states in the IRG or the number of information relations in the IRG. This paper uses the metric of $M_i(k) = |\Upsilon_i(k)| + |Base_i(k)|$, i.e. the number of the information relations in the IRG plus the number of base states. Whenever the IRG changes, $M_i(k)$ is checked and if $M_i(k) > Q$, the marginalization process is triggered.

For the second question, EDCL marginalizes states using the following rules: 1. it marginalizes states in time order–from the oldest to the latest; 2. for states having the same time stamp, the states with the smaller agent Ids are marginalized first; 3. but for states having the same time stamp, the self-state is always marginalized last. Take Agent 1's IRG in Figure 5.4(a) for example. The marginalization order is states 3.1, 2.2, 3.2, 2.3, 1.3 and 1.4.

Since the marginalization process always chooses the oldest states first, and the oldest states are the base states in chains, the base belief of the IRG also needs to be updated. When there are newer states than the marginalized state existing, the one right above (later than) it becomes the new base state of that chain, and all information relations about this new base state and the other base states are fused. Equations 5.8–5.10 are used for fusing information relations, and Equation 5.11 is used for marginalizing the old base state. After the marginalization, the IRG (including $\Upsilon_i(k)$ and $bel(\mathbf{x}_B(k))$) and the state Id list are updated, but the latest self-state estimate does not change. The marginalization process fuses states one by one. It checks $M_i(i)$ after each fusion, and does not stop until $M_i(k) \leq Q$.

Take Agent 1's IRG in Figure 5.4(a) for example. Assuming $Q = 5$, in current IRG $M_1(4) = 3 + 6 = 9 > Q$, therefore, the marginalization process is triggered. Following the marginalization rule, State 3.1 is first marginalized. After making State 3.2 as the new base state and fusing the information relations 1 and 2, the new base states are 1.3, 2.2 and 3.2, and the information relations are 4, 5, and 6. Now $M_1(4) = 6 > Q$. So State 2.2 is further marginalized. Finally, the base states become 1.3, 2.3, 3.2, and the remaining information relation is 6. Now $M_1(4) < Q$, so the marginalization process stops.

5.5 EDCL Analysis

Optimality analysis: when agents only exchange information relations without involving base node beliefs, the estimate of EDCL is optimal, which means that it achieve the exact probability dense function (pdf) if all the random error vectors follow multi-variant Gaussian distribution, or the minimum mean square error if the error vectors are non Gaussian distribution. The approximation only occurs when there exist explicit (Case 3 in IRG meta-data exchanging) or implicit (a part of Case 4) conflicts, because in these cases, agents lose the correlation information (explicit conflicts) or use the same information multiple times (implicit conflicts). The optimality is determined by parameter Q. The bigger Q is, the less probability the conflicts occur, and when Q is big enough, Cases 3 and 4 would not occur at all, and thus no conflict would occur.

Complexity analysis: Q determines not only the optimality, but also the complexity of EDCL. Assuming there are n states in an IRG, the complexity of Equations 5.7–5.10 is $O(1)$. For Equation 5.11, if $\mathbf{L}$ is sparse, it is only $O(1)$, but if $\mathbf{L}$ is dense, it is $O(n^2)$. For the estimation Equation 5.5, it needs $O(n^3)$. For an agent with Q, the memory usage would never exceed $O(Q)$. For the computational complexity, a GPS or DR measurement needs the complexity of $O(1)$; an inter-agent measurement, assuming two agents have Q_i and Q_j, needs the maximum complexity of $\max(O(Q_i^2 Q_j), O(Q_i^3))$ for Agent i. For the communication cost, an agent would never send more than $O(Q)$ number of information relations to the other. Therefore, parameter Q bounds the resource cost. The practical choice of Q depends on the agent's hardware capability and the encountering pattern (the number of agents it meets and the encountering frequency).

5.6 Evaluation

In this section, we evaluate EDCL with two cases. One is a simple simulation which only involves four agents. The purpose is to study how parameters influence EDCL's performance. The other is an emulation on a real trace, which involves a large number of taxis based on a real GPS data set. The purpose of this evaluation is to show the EDCL performance with real agent trajectories.

5.6.1 Simple Simulation

For this simulation, there are only four agents with a full networking connection. They start at $(1, 0)$, $(2, 0)$, $(3, 0)$ and $(4, 0)$, respectively, and keep moving in the y-axis direction step by step. Each step is 20m distance, and there are totally 9 steps. They only communicate

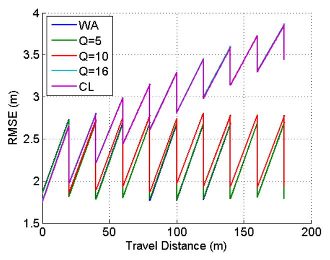


Figure 5.5: RMSE for Agent 1

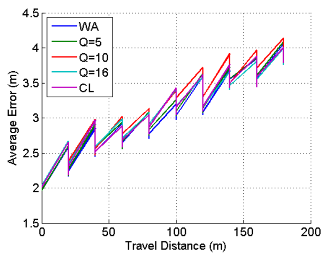


Figure 5.6: Localization error for Agent 1

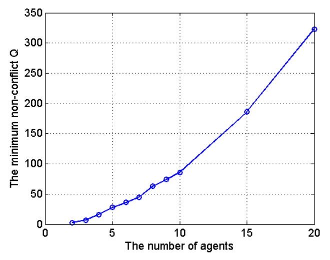


Figure 5.7: The number of agents vs. Q

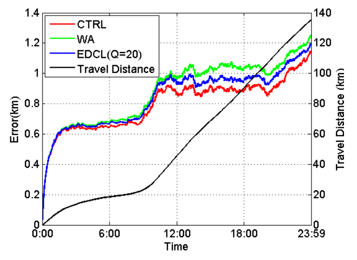


Figure 5.8: Average error over time

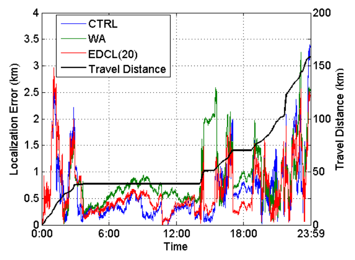


Figure 5.9: Localization error for a typical taxi

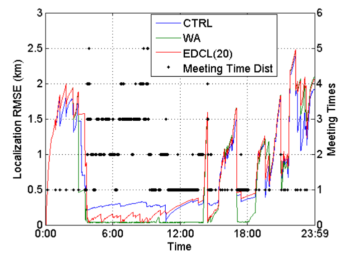


Figure 5.10: Localization RMSE for a typical taxi

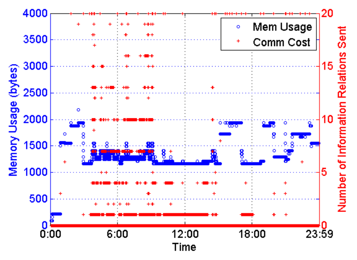


Figure 5.11: EDCL overhead for a typical taxi

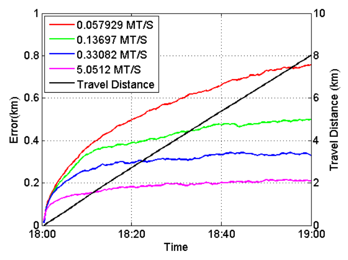


Figure 5.12: Average error vs. network density

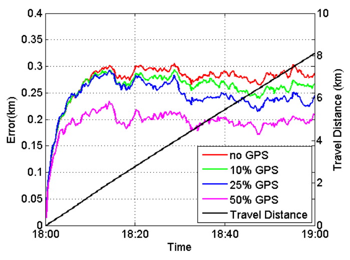


Figure 5.13: Average error of non-GPS taxis

with each other after each step to share their positioning information. The error covariance of each axis of the DR module is 10% of the travel distance d, i.e. $\mathbf{R}_{DR} = \begin{bmatrix} 0.1d & 0 \\ 0 & 0.1d \end{bmatrix}$. The inter-agent measurement module measures the relative distance between two agents, whose error covariance is 5% of the real distance, i.e. $R_{IA} = 0.05d$. We generate measurements corrupted by random errors, feed them into different filters and compare their performance. The filters we compare include weighted average filter (WA, i.e. EDCL with $Q = 1$), EDCL with different Qs, and the centralized optimal filter (CTRL).

Figure 5.5 shows the root mean square error (RMSE) of these filters' estimation. The RMSE, which is defined as $\sqrt{trace(\mathbf{P})}$, is an indicator of how large the estimation uncertainty ($\mathbf{P}$) is. This uncertainty decides the weights when two agents merge their positioning estimates, and the confidence region [238] of the estimate. When the filter is consistent, the RMSE represents the mean estimation error. But when the filter is not consistent, the RMSE is not the indicator any more, and thus the above properties do not hold. In Figure 5.5, we see that the RMSE of WA and EDCL with $Q = 5, 10$ is below the optimal RMSE (CTRL). This is because they encounter some correlation structural conflicts, and thus suffer from the over-confidence problem. When $Q = 16$, the RMSE is almost overlapped with CTRL, which means Q is big enough to avoid the conflicts. Another observation is that the RMSE of EDCL with $Q = 10$ is better than $Q = 5$, and $Q = 5$ is better than WA. This is because the bigger Q is, the less conflicts an agent would have, although the difference in this scenario is very small (since their moving patterns stay the same).

Figure 5.6 shows the localization errors of these filters. The curves are very similar to the optimal RMSE curves in Figure 5.5. We see that the error grows as agents move forward, but suddenly reduces when agents communicate with each other. The performances of the 5 filters are similar. This is because the 4 agents' moving patterns are almost the same, and thus their uncertainties are very close. Therefore, the error can be averaged out without considering the correlation between them.

Figure 5.7 shows the minimum Qs for avoiding the conflicts versus different numbers of agents. Since the communication number increases quadratically as the number of agents increases, the minimum Q also increases quadratically. When the number of agents is 20, the non-conflict minimum Q is 323. However, in reality, if agents do not stick together all the time or the application does not require a strict consistency, Q does not need to be that large. Furthermore, Q not only can be a permanent parameter stored in memory according to a device's capability, but also can change dynamically if EDCL runs in a multi-application platform (e.g. cell phones) and the other applications have fluctuating requirements on resources, or if the encountering pattern changes from time to time.

5.6.2 Taxi Localization

This subsection evaluates EDCL with a real taxi GPS data set. The data set we use is from Microsoft Research [239], which contains the GPS trajectories of 10,357 taxis during the period of Feb. 2nd to Feb. 8th, 2008 in Beijing. Since the data set is very large, we only focus on those taxis which are active in the urban area on Feb. 3rd. The area is $20km \times 25km$, which includes 1,256 taxis. The GPS data is interpolated so that the resolution is 15s. The GPS data are used for generating measurements with random errors and also for the ground truth. The uncertainty of the DR module and the inter-agent measurement module are set the same as in the above simulation. By default each agent does not have GPS measurements. The range of the inter-agent measurement module is set to 20m by default. The EDCL parameter Q is set to 20. Note that we do not expect to localize taxis without using GPS, but only aim to demonstrate the concept of EDCL in a large scale network, and compare the performance of EDCL with the optimal centralized filter.

Figure 5.8 shows the average localization errors for all 1,256 taxis during the day. Most of the time, CTRL is 7.3% better than EDCL (Q=20), and EDCL (Q=20) is 8.2% better than WA. At the end of the day, the average localization error of EDCL is 1.18km, which is only 0.88% of the travel distance.

Figures 5.9, 5.10 and 5.11 show the localization error, RMSE and the overhead for a typical taxi during the day. The curve of Figure 5.9 is a little bit noisy, but we still see that CTRL performs the best and WA is the worst most of the time. The RMSE curves in Figure 5.10 look very similar as the localization error curves in Figure 5.9. However, the curves of EDCL and WA are below that of CTRL, which indicates the occurrence of the over-confidence problem. Furthermore, the RMSE of EDCL is larger than WA. This is because EDCL preserves more historical measurements than WA, and thus keeps more correlation information. The second axis of Figure 5.10 shows the meeting times of the taxi. We see that whenever there is an encounter, the RMSE and localization errors are reduced.

The more the encounters, the more the error is reduced.

Figure 5.11 shows the EDCL overheads for the taxi. Since Q bounds the memory usage and the communication cost, we see that when $Q = 20$, the memory usage never exceeds 2k bytes, and the number of information relations it sends is always no more than 20. This shows that the overhead of EDCL is controllable, which is a very useful property for portable devices in practice.

Figure 5.12 shows how network density influences the EDCL performance. In order to increase the network density, we increase the inter-agent measurement range from 20m, to 50m, 100m, and 500m. Consequently, the encountering times increase from 0.058 encounter per taxi per time step (MT/S), to 0.137 MT/S, 0.331 MT/S, and 5.051 MT/s during 18:00 to 19:00. We see that the localization accuracy is dramatically improved. Since we only consider 1,256 taxis, in practice there are many more taxis, and thus the encounters are very likely more than 0.331 MT/S. Hence we expect an even better performance in practice.

Figure 5.13 shows the EDCL (Q=20) localization accuracy when we insert some GPS data into the network. During time 18:00 to 19:00, for all 1,256 taxis, we equip 10%, 25%, or 50% of the taxis with GPS, and calculate the average localization error for the non-GPS taxis. The inter-agent measurement range of this experiment is set to 100m. We see that as the number of GPS measurements increases, the localization accuracy is dramatically improved, since the non-GPS taxis have more chances to obtain the accurate estimates from GPS taxis.

5.7 Discussion

In this chapter, we proposes an elastic, decentralized, and collaborative localization approach (EDCL) for localizing mobile agents accurately and efficiently. Different from the optimal and the weighted average (WA) distributed localization algorithms, EDCL is able to make a trade-off between consistency and overhead by adjusting the marginalization factor Q which indicates the number of historical measurements that can be stored in memory.

The efficient implementation design of EDCL is realized through the INOVA algorithm along with the *information relation graph* concept and a *chain based data structure*. The evaluation of EDCL is based on a real data set simulation involving 1,256 taxis in Beijing, China. The results show that EDCL achieves not only a close to optimal localization accuracy, but also much better RMSE than the WA method, and the overhead is controllable. In the future, how to set the marginalization factor Q and how dynamic Q impact the whole network need to be studied.

Chapter 6

Indoor Pedestrian Localization System

This chapter presents the implementation of an indoor pedestrian localization system. As described in Section 2.3, a basic indoor pedestrian localization system is the inertial only navigation system, where the inertial data (the acceleration and the angular rate of the pedestrian body) are collected and integrated to calculate the continuous positions. To outperform the inertial-only navigation system, hybrid sources of position information are needed. Such systems either require infrastructure (such as RFID and Wifi) or auxiliary information (e.g., maps). However, in many scenarios, such as fire fighting and the underground mining, these infrastructures or auxiliary information are often unavailable, and thus a fully self-contained localization system is required.

In this chapter a novel self-contained system is presented. The system uses a foot-mounted inertial navigation unit as the basic personal tracking module, a RSSI estimator as the inter-person distance measurement module, and the EDCL algorithm (see Chapter 5) as the filter to localize each pedestrian indoors. It is fully decentralized, and uses pedestrian encountering opportunities to share position information to collaboratively localize each pedestrian indoors. The practical usage includes firefighting, underground mining, indoor officer or robotics localization etc.

Note that for concept-proof and cost-saving purposes, the foot-mounted inertial navigation unit and the RSSI estimator are adopted. However, these modules can be replaced by others with the similar functionality in practice. For example, the foot-mounted navigation module can be replaced by a mobile-phone based navigation module in the officer indoor localization scenario, and the RSSI inter-person distance estimator can be replaced by a laser-based distance estimator in scenarios like robotics indoor operations, which require more accuracy.

The chapter is organized as follows. Section 6.1 gives the overview of the whole system design. Sections 6.2 and 6.3 describe implementations of the inter-agent RSSI distance estimator and the foot-mounted inertial navigation module, respectively (for the EDCL module, please refer to Chapter 5). Section 6.4 presents the integration of the whole system, and Section 6.5 gives the evaluation results. A further discussion is presented in Section 6.6.

6.1 System Overview

As Figure 6.1 shows, the system has three components: the inter-agent distance estimator, the inertial navigator and the EDCL filter module. Each agent wears such a system. The inter-agent distance estimator outputs the distance estimation between agents. In our implementation, a radio is used to collect RSSIs (Received Signal Strength Indicator) from other agents, and map them to distance values. The inertial navigator outputs the displacement of the agent between two time points. It contains an accelerometer and a gyroscope to get acceleration and angular rate values, and then uses the ZUPT (Zero Velocity Update) algorithm to calculate the agent's position and velocity in the navigation coordinate system (assuming the initial position is known). Given the inter-agent distance estimation and the self displacement estimation, EDCL is able to estimate the agent's position based on a graph structure (see Chapter 5) over time. The output position from EDCL is also in

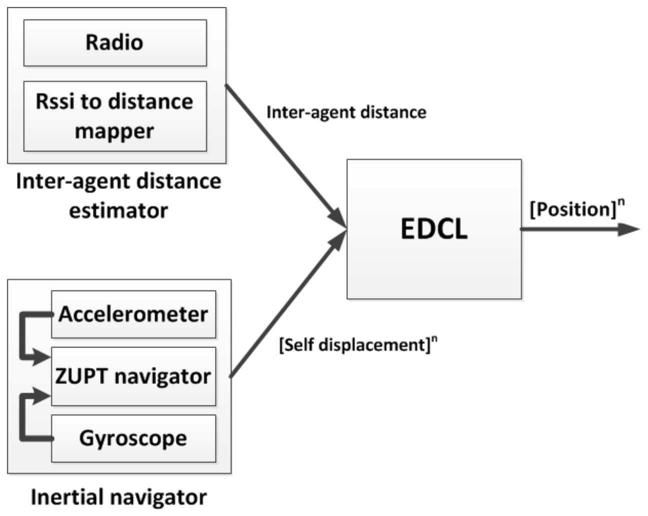


Figure 6.1: Indoor pedestrian localization system diagram.

the navigation coordinate system. Note that the radio in the diagram is used to not only estimate the inter-agent distance but also share the position information between agents.

Two things need to be emphasized. First, the whole system is fully self-contained and decentralized. The inertial navigator does not rely on any external infrastructure or map information. The RSSI based inter-agent distance estimator and the information sharing work by opportunity. If there are no other agents nearby, the performance of the system is downgraded to the inertial-only navigation system. Second, the implementation of each module in the diagram is flexible. The inter-agent distance estimator can be implemented by a laser based range finder, and the inertial navigator can be implemented by either a foot-mounted IMU (Inertial Measurement Unit) or a cell phone based IMU. The communication between the EDCL module and the other two modules can be either wireless (such as bluetooth or zigbee) or wired (it is possible to integrate everything in one device). Since the output rate of the inter-agent distance estimator and the inertial navigator is not high (one output per second is good enough for most applications), the wireless bandwidth requirement is not high.

6.2 Inter-agent RSSI Distance Estimator

As described in Section 2.1.1, there are many means for measuring range, such as using the TOA of light, the TDOA between electromagnetic wave and sound, and the Doppler effect for moving objects. For cost consideration and the proof of concept, our implementation chooses RSSI to estimate the distance between agents. Obviously, the most advantage of this method is that the RSSI comes almost for free on radio devices, but the down side is that the distance estimation is not accurate, especially indoors. Therefore, this section analyzes the reasons of the inaccuracy, and proposes several methods to improve it.

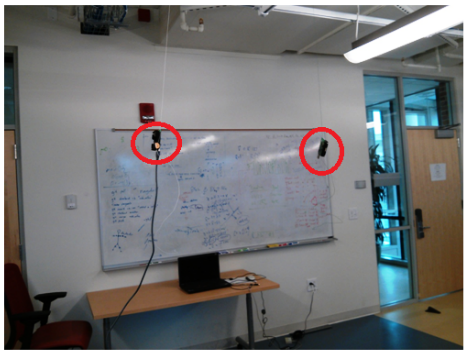
6.2.1 RSSI study

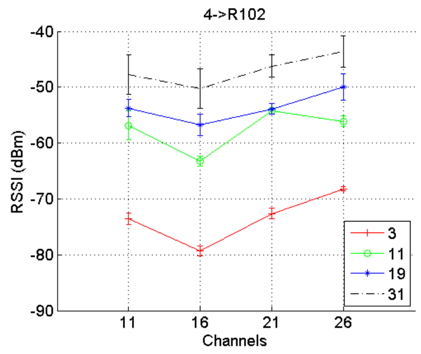
Ideally, in a free space, the received signal strength is given as the following formula [90].

$$P_r(d) = \frac{P_t G_t G_r \lambda^2}{(4\pi)^2 d^2 L} \tag{6.1}$$

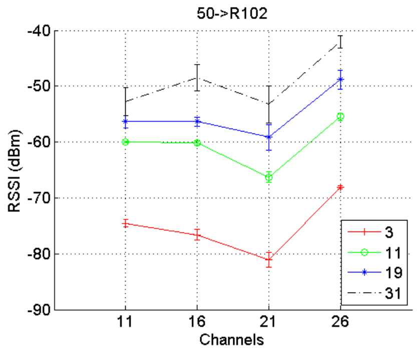
where P_t is the transmitted signal power. G_t and G_r are the antenna gains of the transmitter and the receiver, respectively. $L(L \geq 1)$ is the system loss, and λ is the wavelength.

However, in reality there are many factors impacting the RSS. The most significant ones are: multi-path effect, shadowing effect and the radio noise. Multi-path effect means that the receiver may receive the signal from multiple paths because of the reflection, refraction and scattering of the environments. Signal arrives the receiver through different paths. Some of the paths have the constructive effect on RSS, while others have the destructive effect, which may result in the significant deviation from the ideal value deduced by Formula 6.1. The multi-path propagation causes two effects in practice. One is that by moving the receiver's position a little bit, the RSS changes significantly. Another is that different frequencies have very different RSS values, and one of the frequency has the deepest fading effect (having the least RSS value), which is called selective fading effect. The shadowing effect means that the RSS fluctuates due to the objects obstructing the propagation path

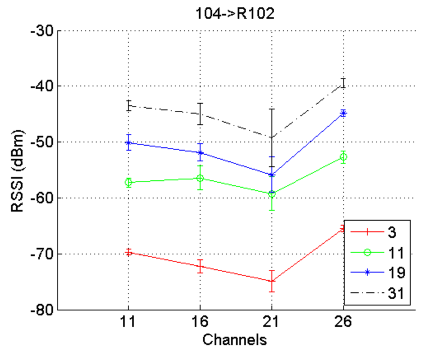

(a) Spacial lab environment



(b) Sending from Node 4 to R102



(c) Sending from Node 50 to R102



(d) Sending from Node R104 to R102

Figure 6.2: The study of frequency, transmission power and RSS. (a) is the experiment setup. The nodes are hung in the air as the red circles show. (b) is the RSSI vs. frequency and transmission power from Node 4 to Node R102. (c) is the RSSI vs. frequency and transmission power from Node 50 to Node R102. (d) is the RSSI vs. frequency and transmission power from Node 104 to Node R102.

between the transmitter and the receiver. For example, if there is a person or a wall between the transmitter and the receiver, the signal may be refracted or absorbed by this person or the wall, and thus differs much from the deduced value. The third impacting factor is the radio noise, which means that the RSS is a random value even it is transmitted in the free space. The discrepancy of hardware also causes the static or dynamic deviations.

We studied the multipath effect regarding with the factors of radio frequency, distance, transmission power, and different environments. Figure 6.2 shows the study of frequency, transmission power and RSS. Two TelosB [13] Nodes are hung in the air with the distance of 63 $\pm$ 3cm apart in a spacial lab environment. One node sends packets to the other at

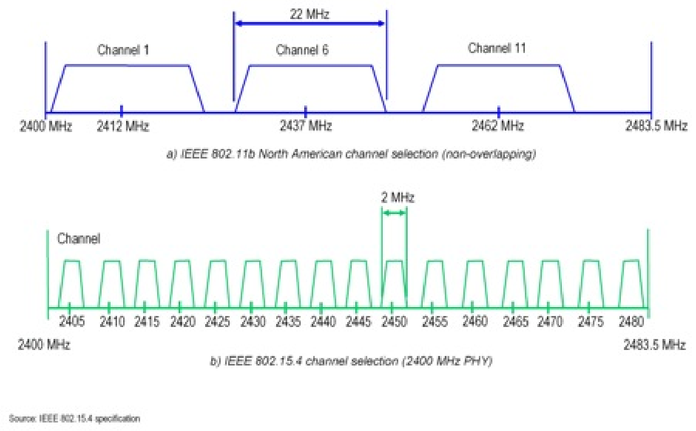
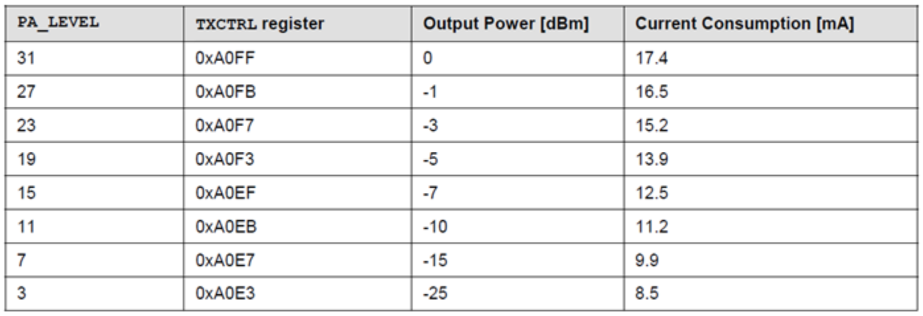


Figure 6.3: Wifi channel frequencies vs. 802.15.4 channel frequencies.

the rate of 50 packets/second. The receiver is fixed (Node R102), while different senders are tested (Nodes 4, 50, and 104). Different frequencies (Channel 11, 16, 21 and 26) and different transmission powers (power levels 3, 11, 19 and 31) are studied. The channel frequencies and the power level table are shown in Figures 6.3 and 6.4, respectively.

From Figures 6.2(b) to 6.2(d), we see that although the distance is the same, different frequencies have different RSS. Furthermore, some frequency has most significant fading effect, e.g. Channel 16 in Figure 6.2(b), and Channel 21 in Figures 6.2(c) and 6.2(d). This indicates that different frequency impacts the RSS differently because of the multipath effect. The second observation from these graphs is that the transmission power does not change the multipath effect (the shape of the plot for each transmission power is similar). This is intuitive because the transmission power only changes the strength of the signal, but cannot change the path it travels. The third observation is that with different senders, the receiver gets different RSS. This is mostly because of the hardware discrepancy, and some minor position changes may contribute to this error too.

We also studied the RSS under different environments, distances and frequencies. Figures 6.5 and 6.6 show the experimental results in the corridor and the studying lobby, respectively. The transmission rate is still 50 packets/second. Several observations are made from these plots. First, the RSS is not monotonically decreasing as distance increas-



PA_LEVEL	TXCTRL register	Output Power [dBm]	Current Consumption [mA]
31	0xA0FF	0	17.4
27	0xA0FB	-1	16.5
23	0xA0F7	-3	15.2
19	0xA0F3	-5	13.9
15	0xA0EF	-7	12.5
11	0xA0EB	-10	11.2
7	0xA0E7	-15	9.9
3	0xA0E3	-25	8.5

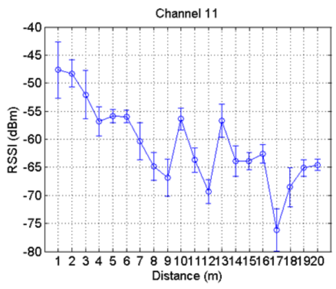
Figure 6.4: CC2420 transmission power configuration [13].

es. Second, there is a relative large noise in the real environment (see the error bar at each point). Third, in some place, by changing a little position, RSSI differs a lot. For example, the RSSI at distances 9m and 10m of Channel 11 in the corridor, and the distances of 4m and 5m of Channel 16 in the studying lobby. Third, different channel shows different shapes. For example, in corridor environment, Channel 11 seems to vary much from distances 9m to 14m, while Channel 21 RSSI drops quickly from distance 13m to 16m. Fourth, the overall RSSI in the corridor is lower than that in the studying lobby environment.

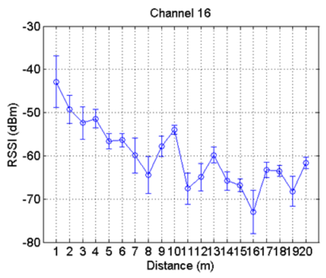
This experiment tells us that the multipath effect has a strong impact on RSS values. We cannot simply use Formula 6.1 to map the RSS to distance. Additionally, different environment has different path loss effect, and the RSS noise cannot be neglected. Figure 6.7 shows the distribution of RSSIs of channel 21 in the lobby. We see that the it is single modal and could be treated as a normal distribution.

The body shadowing effect is also studied. We conducted a similar experiment in the corridor from distance 1m to 5m under the transmission power level 5. But this time the node is attached to a person's chest. In one experiment, two people are face to face, but in another experiment people are face to back, which means there is a body shadowing in the study. Figure 6.8 clearly shows that the RSSI with body shadowing is significantly reduced. This experiment tells us that in practice, if the device is wore on the human body, we should either consider this shadowing effect into our model or use some method to avoid the body shadowing.

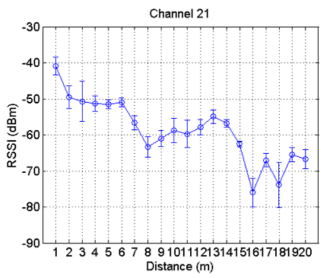
(a) Corridor environment



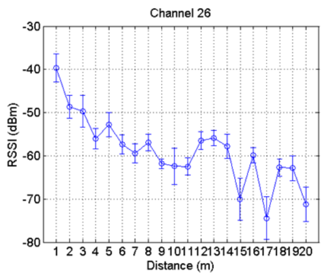
(b) Corridor Channel 11 RSSI



(c) Corridor Channel 16 RSSI

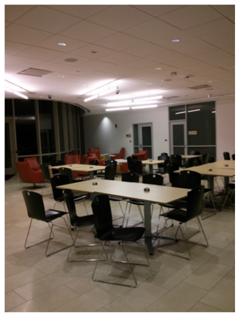


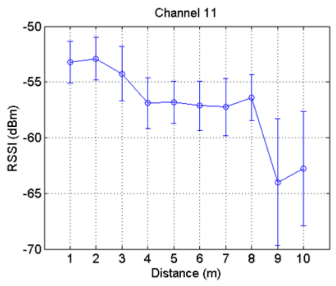
(d) Corridor Channel 21 RSSI



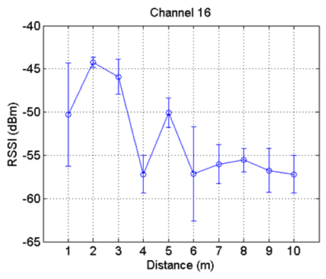
(e) Corridor Channel 26 RSSI

Figure 6.5: RSSI under different transmission power and distances in the corridor.

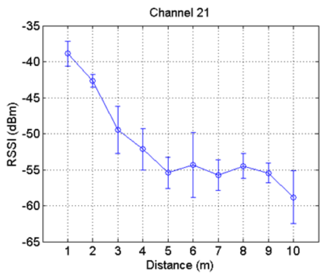

(a) Studying lobby environment



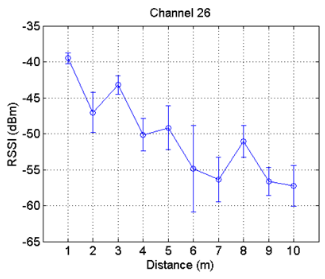
(b) Studying lobby Channel 11 RSSI



(c) Studying lobby Channel 16 RSSI



(d) Studying lobby Channel 21 RSSI



(e) Studying lobby Channel 26 RSSI

Figure 6.6: RSSI under different transmission power and distances in the studying lobby.

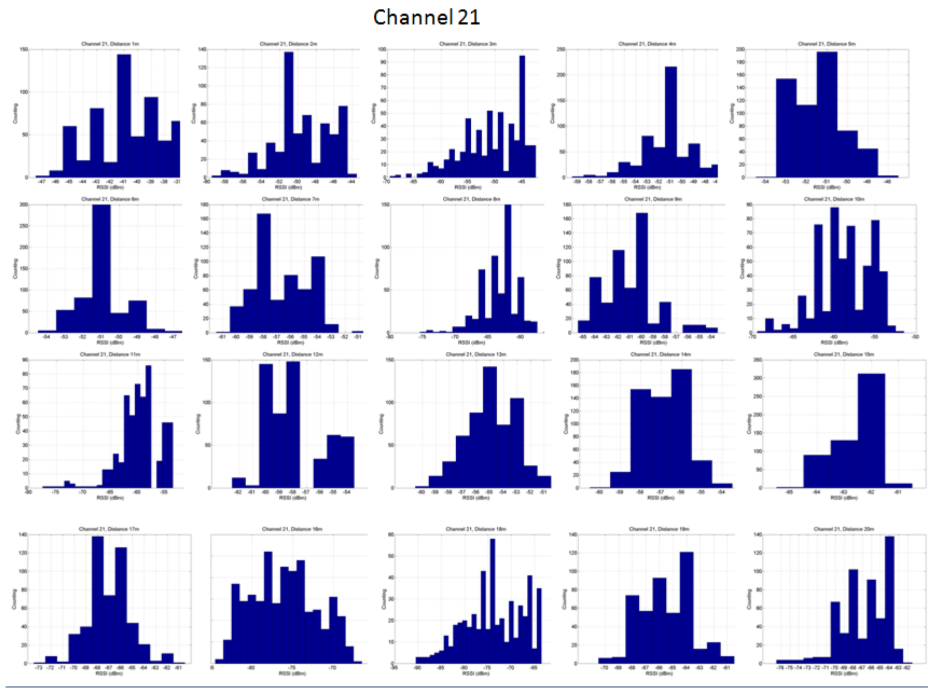


Figure 6.7: RSSI distribution

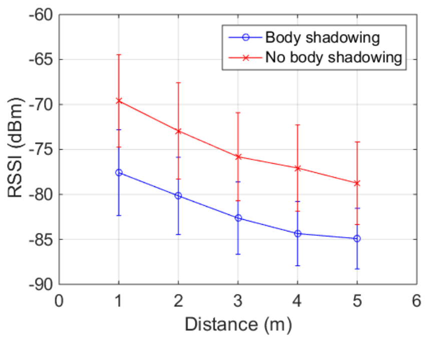


Figure 6.8: RSSI of body shadowing vs no body shadowing.

6.2.2 The RSSI to Distance Estimation Module

From the above section, we see that in order to use RSSI as a distance estimator, several issues need to be solved: 1) because of the multipah effect, stronger RSSI does not necessarily mean shorter distance, and a little change of position may cause RSSI to vary significantly. 2) Because of the multipath effect, the selected frequency may have selective fading effect at some position, which causes the RSSI to be unusable. 3) Even given the ideal environment, the RSSI has unneglectable noise. 4) If the device is wore on human body, the body shadowing effect reduces the RSSI.

To solve the above issues, we use the following methods accordingly. 1) If the device is still, averaging cannot help remove the multipah effect. However, if the device is moving, the signal traveling path is also changing, and thus averaging can help reduce the deviation. 2) One frequency may have a selective fading effect. However, if multiple frequencies are used and only the median RSSI is taken, the selective fading effect would be improved significantly. 3) Modeling the probabilistic uncertainty and averaging the RSSI are good means to deal with noise. Furthermore, we propose a novel filter which weights more on high RSSI values. Using this filter, the distance estimation is restricted in the short distance, which results in a better performance. 4) Because of the dynamics of the environment and human actions, the body shadowing effect is hard to model in practice. We avoid the body shadowing effect by putting the radio on the top of the head (equipping the radio on a hat).

This section mainly focuses on developing the novel filters aforementioned. The frequency hopping protocol is described in the next section. The performance of the filters with the frequency hopping protocol is evaluated in Section 6.2.4.

6.2.2.1 The Corrected Maximum Likelihood Distance Estimator

Neal Patwari et al. [86] studied the relationship between RSS and range, and proposed a corrected maximum likelihood distance estimator. First, the ensemble mean power $\bar{P}(d)$ at distance d is typically modeled as

$$\bar{P}(d) = \Pi_0 - 10 n_p \log_{10} \frac{d}{\Delta_0} \tag{6.2}$$

where Π_0 is the received power (dBm) at the reference distance Δ_0, and n_p is the path-loss exponent whose typical value is between 2 and 4. According to many years' measurement results [240–242], the error of the received power $P(d) - \bar{P}(d)$ follows the normal distribution.

Therefore, it can be written as:

$$f(P_{i,j} = p \mid \{\mathbf{z}_i\}_{i=1}^{N}) = \mathcal{N}(p; \bar{P}(\parallel \mathbf{z}_i - \mathbf{z}_j \parallel), \sigma_{dB}^2) \tag{6.3}$$

where $P_{i,j}$ represents the received power at node i transmitted by sender j, $\mathbf{z}_i$ is the position of Node i, $\mathcal{N}(x; \mu, \sigma^2)$ is the notation of the value x of a normal distribution $p.d.f.$ with the mean μ and the deviation σ, σ_{dB}^2 is the variance in the unit dB, whose value is usually constant, and $\parallel \mathbf{z}_i - \mathbf{z}_j \parallel$ is the distance between nodes i and j.

The log-likelihood of $P_{i,j}$ is

$$\log f(P_{i,j} \mid \{\mathbf{z}_{i,j}\}_{i=1}^{N}) = c_1 - \frac{[P_{i,j} - \bar{P}(\parallel \mathbf{z}_i - \mathbf{z}_j \parallel)]^2}{2\sigma_{dB}^2} \tag{6.4}$$

where c_1 is the constant. Obviously, the likelihood function has the maximum when $\bar{P}(\parallel \mathbf{z}_i - \mathbf{z}_j \parallel) = P_{i,j}$. According to Equation 6.2, we have:

$$\begin{aligned} P_{i,j} &= \Pi_0 - 10n_p \log_{10} \frac{\delta_{i,j}^{MLE}}{\Delta_0} \\ \delta_{i,j}^{MLE} &= \Delta_0 10^{\frac{\Pi_0 - P_{i,j}}{10n_p}} \end{aligned} \tag{6.5}$$

where $\delta_{i,j}^{MLE}$ is the maximum likelihood estimation for the distance of the measurement $P_{i,j}$. If we write $P_{i,j} = \bar{P}(\parallel \mathbf{z}_i - \mathbf{z}_j \parallel) + \eta_{i,j}$, then we have

$$\begin{aligned} \delta_{i,j}^{MLE} &= \Delta_0 10^{\frac{\Pi_0 - \bar{P}(\parallel \mathbf{z}_i - \mathbf{z}_j \parallel) - \eta_{i,j}}{10n_p}} \\ \delta_{i,j}^{MLE} &= \Delta_0 10^{\frac{10n_p \log_{10} \frac{\parallel \mathbf{z}_i - \mathbf{z}_j \parallel}{\Delta_0} - \eta_{i,j}}{10n_p}} \\ \delta_{i,j}^{MLE} &= \parallel \mathbf{z}_i - \mathbf{z}_j \parallel 10^{\frac{-\eta_{i,j}}{10n_p}} \end{aligned} \tag{6.6}$$

The first moment and the second moments of δ^{MLE} are:

$$\begin{aligned} E(\delta_{i,j}^{MLE}) &= C \parallel \mathbf{z}_i - \mathbf{z}_j \parallel && (6.7) \\ var(\delta_{i,j}^{MLE}) &= (C^4 - C) \parallel \mathbf{z}_i - \mathbf{z}_j \parallel^2 && (6.8) \\ \text{where } C &= exp\frac{1}{2\gamma}, \\ \text{and } \gamma &= (\frac{10n_p}{\sigma_{dB}\log 10})^2 \end{aligned}$$

The parameter C is a multiplicative bias factor, and thus needs to be corrected. Hence the bias corrected version of the estimator is

$$\delta_{i,j}^{CMLE} = \frac{\Delta_0}{C} 10^{\frac{\Pi_0 - P_{i,j}}{10n_p}} \tag{6.9}$$

Therefore, the first two moments of $\delta_{i,j}^{CMLE}$ are

$$\begin{aligned} E(\delta_{i,j}^{CMLE}) &= \parallel \mathbf{z}_i - \mathbf{z}_j \parallel && (6.10) \\ var(\delta_{i,j}^{CMLE}) &= (C^2 - C^{-1}) \parallel \mathbf{z}_i - \mathbf{z}_j \parallel^2 && (6.11) \end{aligned}$$

The above estimator is for only one RSS value. If multiple RSS values about the same distance are collected, we can modified the above equations accordingly. Assuming each RSS measurement is i.i.d. (independent and identically distributed), the joint p.d.f. is written as

$$f(\mathbf{P}_{i,j} = \{p\}_{k=1}^M \mid \{\mathbf{z}_i\}_{i=1}^N) = \prod_{k=1}^M \mathcal{N}(p; \bar{P}(\parallel \mathbf{z}_i - \mathbf{z}_j \parallel), \sigma_{dB}^2) \tag{6.12}$$

The log-likelihood function is:

$$\log f(\mathbf{P}_{i,j} = \{p\}_{k=1}^M \mid \{\mathbf{z}_{i,j}\}_{i=1}^N) = c_1' - \frac{1}{2\sigma_{dB}^2} \sum_{k=1}^M [P_{i,j}^{(k)} - \bar{P}(\parallel \mathbf{z}_i - \mathbf{z}_j \parallel)]^2 \tag{6.13}$$

The maximum value is reached when $\bar{P}(\parallel \mathbf{z}_i - \mathbf{z}_j \parallel) = \bar{\mathbf{P}}_{i,j} = \frac{1}{M}\sum_{k=1}^{M} P_{i,j}^{(k)}$. Hence, the estimator is:

$$\delta_{i,j}^{MMLE} = \Delta_0 10^{\frac{\Pi_0 - \bar{\mathbf{P}}_{i,j}}{10n_p}} \tag{6.14}$$

The first two moments are:

$$\begin{aligned}
E(\delta_{i,j}^{MMLE}) &= C_M \parallel \mathbf{z}_i - \mathbf{z}_j \parallel &(6.15)\\
var(\delta_{i,j}^{MMLE}) &= (C_M^4 - C_M) \parallel \mathbf{z}_i - \mathbf{z}_j \parallel^2 &(6.16)\\
\text{where } C_M &= exp\frac{1}{2\gamma_M},\\
\text{and } \gamma_M &= (\frac{10\sqrt{M}n_p}{\sigma_{dB}\log 10})^2
\end{aligned}$$

The bias corrected estimator version is

$$\delta_{i,j}^{CMMLE} = \frac{\Delta_0}{C_M} 10^{\frac{\Pi_0 - \bar{\mathbf{P}}_{i,j}}{10n_p}} \tag{6.17}$$

And the first two moments of $\delta_{i,j}^{CMMLE}$ are

$$\begin{aligned}
E(\delta_{i,j}^{CMMLE}) &= \parallel \mathbf{z}_i - \mathbf{z}_j \parallel &(6.18)\\
var(\delta_{i,j}^{CMMLE}) &= (C_M^2 - C_M^{-1}) \parallel \mathbf{z}_i - \mathbf{z}_j \parallel^2 &(6.19)
\end{aligned}$$

6.2.2.2 The Corrected Flattened Maximum Likelihood Distance Estimator

From Equations 6.15 and 6.18 we see that the longer the distance two nodes have, the bigger the estimation variation it has. Additionally, from our RSS studies, we found that the multipath effect is more likely to have the destructive impact on RSS than to have the constructive impact. If the RSS is lower than some threshold, it is not meaningful to differentiate the contributions of these RSS values to the distance mapping. For example, if one RSSI is -85 dBm (-92 dBm is the lower limit for the device to detect) and another is -87 dBm, it does not really indicate that -85 dBm is more likely to have the shorter distance than -87 dBm does. Therefore, the idea of the flattened maximum likelihood distance estimator is to treat all the RSS values lower than the threshold T the same. Its p.d.f. for a single RSS value is written as:

$$f(P_{i,j} = p \mid \{\mathbf{z}_i\}_{i=1}^{N}) = \begin{cases} \frac{1}{\Phi(T;\bar{P}(\|\mathbf{z}_i-\mathbf{z}_j\|),\sigma_{dB}^2)} & p < T \\ \mathcal{N}(p;\bar{P}(\| \mathbf{z}_i - \mathbf{z}_j \|),\sigma_{dB}^2) & p \geq T \end{cases} \tag{6.20}$$

Assume that M RSS measurements for the same distance are collected. By rearranging them, we have the first K values are less than T, and the remaining $M-K$ are bigger than or equal to T. The joint p.d.f. of this group of RSS is:

$$f(\mathbf{P}_{i,j} = \{p\}_{k=1}^{M} \mid \{\mathbf{z}_i\}_{i=1}^{N}) = \frac{\prod_{k=1}^{M-K} \mathcal{N}(p;\bar{P}(\| \mathbf{z}_i - \mathbf{z}_j \|),\sigma_{dB}^2)}{\Phi^K(T;\bar{P}(\| \mathbf{z}_i - \mathbf{z}_j \|),\sigma_{dB}^2)} \tag{6.21}$$

The log maximum likelihood function is

$$\begin{aligned}\mathcal{L}_F = \log f(\mathbf{P}_{i,j} = \{p\}_{k=1}^{M} \mid \{\mathbf{z}_{i,j}\}_{i=1}^{N}) = \quad & c_F - \tfrac{1}{2\sigma_{dB}^2}\textstyle\sum_{k=1}^{M-K}[P_{i,j}^{(k)} - \bar{P}(\| \mathbf{z}_i - \mathbf{z}_j \|)]^2 \\ & -K \ln \Phi(T;\bar{P}(\| \mathbf{z}_i - \mathbf{z}_j \|),\sigma_{dB}^2)\end{aligned} \tag{6.22}$$

Denoting $\bar{P}(\| \mathbf{z}_i - \mathbf{z}_j \|)$ as μ_F, the first derivative of $\mathcal{L}_F$ is

$$\frac{\partial \mathcal{L}_F}{\partial \mu_F} = -\frac{1}{\sigma_{dB}^2}\sum_{k=1}^{M-K}(\mu_F - P_{i,j}^{(k)}) - \frac{K\mathcal{N}(T;\mu_F,\sigma_{dB}^2)}{\Phi(T;\mu_F,\sigma_{dB}^2)} \tag{6.23}$$

The second derivative of $\mathcal{L}_F$ is

$$\frac{\partial^2 \mathcal{L}_F}{\partial \mu_F^2} = -\frac{M-K}{\sigma_{dB}^2} + \frac{K\mathcal{N}^2(T;\mu_F,\sigma_{dB}^2)}{\Phi^2(T;\mu_F,\sigma_{dB}^2)} + \frac{K(T-\mu_F)\exp\left(-\frac{(T-\mu_F)^2}{2\sigma_{dB}^2}\right)}{\sigma_{dB}^3\sqrt{2\pi}\Phi(T;\mu_F,\sigma_{dB}^2)} \tag{6.24}$$

Since the first derivative and the second derivative of $\mathcal{L}_F$ are supplied, Newton's method can be used to find the maximum. Assuming the estimation of μ_F is $\hat{\mu}_F$, it has the following properties:

$$\sqrt{M}(\hat{\mu}_F - \mu_F) \rightarrow \mathcal{N}(x; 0, \mathbf{I}_F^{-1}) \tag{6.25}$$
$$\mathbf{I}_F = -E(\frac{\partial^2 \mathcal{L}_F}{\partial \mu_F^2}) \approx -\frac{1}{M}\frac{\partial^2 \mathcal{L}_F}{\partial \mu_F^2}|_{\mu_F=\hat{\mu}_F} \tag{6.26}$$

Therefore, the distance estimation is:

$$\delta_{i,j}^F = \Delta_0 10^{\frac{\Pi_0 - \hat{\mu}_F}{10 n_p}} \tag{6.27}$$

The first two moments are:

$$\begin{aligned} E(\delta_{i,j}^F) &= C_F \parallel \mathbf{z}_i - \mathbf{z}_j \parallel && (6.28) \\ var(\delta_{i,j}^F) &= (C_F^4 - C_F) \parallel \mathbf{z}_i - \mathbf{z}_j \parallel^2 && (6.29) \\ \text{where } C_F &= exp\frac{1}{2\gamma_F}, \\ \text{and } \gamma_F &= (\frac{10 n_p \sqrt{M \mathbf{I}_F}}{\log 10})^2 \end{aligned}$$

The bias corrected estimator version is

$$\delta_{i,j}^{CF} = \frac{\Delta_0}{C_F} 10^{\frac{\Pi_0 - \hat{\mu}_F}{10 n_p}} \tag{6.30}$$

And the first two moments of $\delta_{i,j}^{CF}$ are

$$\begin{aligned} E(\delta_{i,j}^{CF}) &= \parallel \mathbf{z}_i - \mathbf{z}_j \parallel && (6.31) \\ var(\delta_{i,j}^{CF}) &= (C_F^2 - C_F^{-1}) \parallel \mathbf{z}_i - \mathbf{z}_j \parallel^2 && (6.32) \end{aligned}$$

6.2.2.3 The Corrected Truncated Maximum Likelihood Distance Estimator

The above corrected flattened maximum likelihood Distance estimator treats all RSS values below the treshold T the same. An even more extreme idea is that we do not trust the low value RSS value at all. So we just throw them away. With this idea, the p.d.f. of the RSS below the threshold T should be normalized as

$$f(P_{i,j} = p \mid \{\mathbf{z}_i\}_{i=1}^N) = \frac{\mathcal{N}(p; \bar{P}(\| \mathbf{z}_i - \mathbf{z}_j \|), \sigma_{dB}^2)}{1 - \Phi(T; \bar{P}(\| \mathbf{z}_i - \mathbf{z}_j \|), \sigma_{dB}^2)} \tag{6.33}$$

Assume that M RSS measurements for the same distance are collected, and their values are all bigger than threshold T. The joint p.d.f. is written as:

$$f(\mathbf{P}_{i,j} = \{p\}_{k=1}^M \mid \{\mathbf{z}_i\}_{i=1}^N) = \prod_{k=1}^{M} \frac{\mathcal{N}(p; \bar{P}(\| \mathbf{z}_i - \mathbf{z}_j \|), \sigma_{dB}^2)}{1 - \Phi(T; \bar{P}(\| \mathbf{z}_i - \mathbf{z}_j \|), \sigma_{dB}^2)} \tag{6.34}$$

The log maximum likelihood function is

$$\begin{aligned}\mathcal{L}_T = \log f(\mathbf{P}_{i,j} = \{p\}_{k=1}^M \mid \{\mathbf{z}_{i,j}\}_{i=1}^N) = \quad & c_T - \tfrac{1}{2\sigma_{dB}^2} \textstyle\sum_{k=1}^M [P_{i,j}^{(k)} - \bar{P}(\| \mathbf{z}_i - \mathbf{z}_j \|)]^2 \\ & -M \ln(1 - \Phi(T; \bar{P}(\| \mathbf{z}_i - \mathbf{z}_j \|), \sigma_{dB}^2)) \end{aligned} \tag{6.35}$$

Denoting $\bar{P}(\| \mathbf{z}_i - \mathbf{z}_j \|)$ as μ_T, the first derivative of $\mathcal{L}_T$ is

$$\frac{\partial \mathcal{L}_T}{\partial \mu_T} = -\frac{1}{\sigma_{dB}^2} \sum_{k=1}^M (\mu_T - P_{i,j}^{(k)}) + \frac{M\mathcal{N}(T; \mu_T, \sigma_{dB}^2)}{1 - \Phi(T; \mu_T, \sigma_{dB}^2)} \tag{6.36}$$

The second derivative of $\mathcal{L}_T$ is

$$\frac{\partial^2 \mathcal{L}_T}{\partial \mu_T^2} = -\frac{M}{\sigma_{dB}^2} + \frac{M\mathcal{N}^2(T; \mu_T, \sigma_{dB}^2)}{(1 - \Phi(T; \mu_T, \sigma_{dB}^2))^2} - \frac{M(T - \mu_T)\exp\left(-\frac{(T-\mu_T)^2}{2\sigma_{dB}^2}\right)}{\sigma_{dB}^3\sqrt{2\pi}(1 - \Phi(T; \mu_T, \sigma_{dB}^2))} \tag{6.37}$$

The maximum point for $\mathcal{L}_T$ can be found by using Newton's method. Assuming it is $\hat{\mu}_T$, the following properties hold:

$$\sqrt{M}(\hat{\mu}_T - \mu_T) \to \mathcal{N}(x; 0, \mathbf{I}_T^{-1}) \tag{6.38}$$

$$\mathbf{I}_T = -E(\frac{\partial^2 \mathcal{L}_T}{\partial \mu_T^2}) \approx -\frac{1}{M}\frac{\partial^2 \mathcal{L}_T}{\partial \mu_T^2}|_{\mu_T = \hat{\mu}_T} \tag{6.39}$$

Therefore, the distance estimation is:

$$\delta_{i,j}^{T} = \Delta_0 10^{\frac{\Pi_0 - \hat{\mu}_T}{10 n_p}} \tag{6.40}$$

The first two moments are:

$$\begin{aligned} E(\delta_{i,j}^{T}) &= C_T \parallel \mathbf{z}_i - \mathbf{z}_j \parallel & (6.41) \\ var(\delta_{i,j}^{T}) &= (C_T^4 - C_T) \parallel \mathbf{z}_i - \mathbf{z}_j \parallel^2 & (6.42) \\ \text{where } C_T &= exp\frac{1}{2\gamma_T}, \\ \text{and } \gamma_T &= (\frac{10 n_p \sqrt{M \mathbf{I_T}}}{\log 10})^2 \end{aligned}$$

The bias corrected estimator version is

$$\delta_{i,j}^{CT} = \frac{\Delta_0}{C_T} 10^{\frac{\Pi_0 - \hat{\mu}_T}{10 n_p}} \tag{6.43}$$

And the first two moments of $\delta_{i,j}^{CT}$ are

$$\begin{aligned} E(\delta_{i,j}^{CT}) &= \parallel \mathbf{z}_i - \mathbf{z}_j \parallel & (6.44) \\ var(\delta_{i,j}^{CT}) &= (C_T^2 - C_T^{-1}) \parallel \mathbf{z}_i - \mathbf{z}_j \parallel^2 & (6.45) \end{aligned}$$

6.2.3 Frequency Hopping TDMA

As explained in Section 6.2.1, the frequency selective fading effect may affect the accuracy of RSS to distance mapping significantly. Additionally, when multiple nodes participate in the packet transmission and RSS sensing, CSMA (carrier sense multiple access) may not fully prevent the transmission collision, and thus impact RSS accuracy. Furthermore, to accurately estimate distance from RSS, a large number of samples need to be collected. Therefore, under the high transmission frequency, some node may not get enough chance to send the packet by using CSMA.

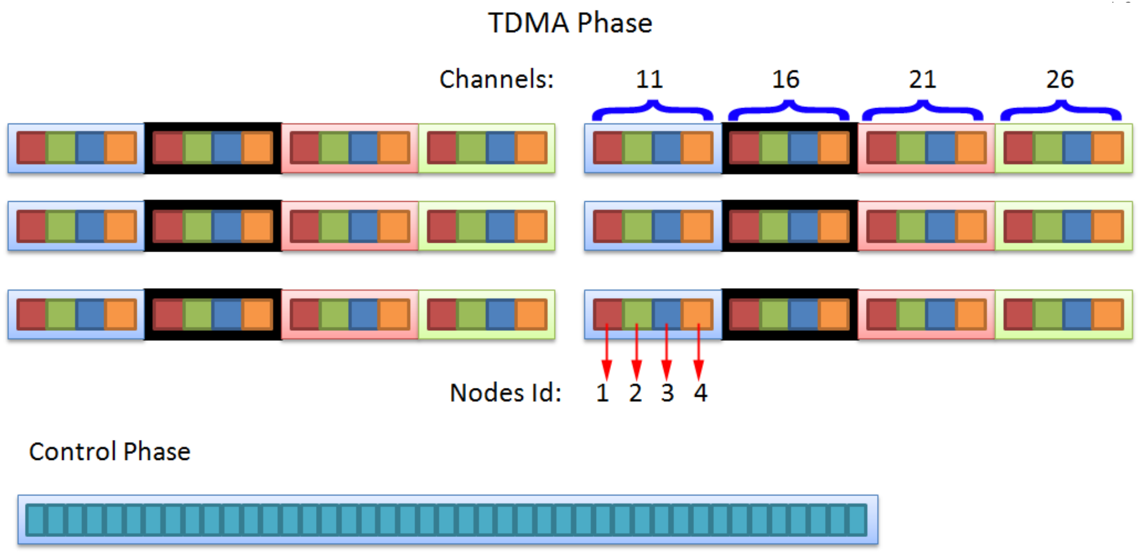


Figure 6.9: Frequency Hopping TDMA Design. There are two time phases: the TDMA time phase and the control phase. In the control phase, the frequency is fixed. The purpose of the control phase is to coordinate the channels and sending time for each participant in the following TDMA phase. In the TDMA phase, it has multiple TDMA periods. Each period has $N \times C$ sending time slot, where N is the number of participants, and C is the number of channels. The different filling colors of time slots represent the different participants (the corresponding participant should send in its own time slot). The different frame colors for a group of time slots represents the different channels.

To solve the above issues, the best way for packet transmission is to combine frequency hopping and TDMA, i.e. nodes keep changing the channel (radio frequency) to send packets to overcome selective fading effect, and each node has its own time slot to send to fully avoid the contention (TDMA). However, there are two challenges for this method: how to coordinate which channel to use so that when a node sends a packet, the receivers are in the correct channel to receive? Due to the discrepancy of hardware, the local clock of each node drifts differently, how to keep all participants' clock synchronized so that they behave as expected?

We design the protocol as Figure 6.9 shows. There are two time phases: one is the TDMA phase and the other is the control phase. In the TDMA phase, each node sends packet in its own time slot under assigned channel, and the channel keeps hopping. In the control phase, a master node sends out control message to tell each node when and under which channel it sends. In addition, the control phase is also for correcting the time drift for all participants. For this design, we need to consider the following questions:

i) How long should each time slot be?

ii) How long should the whole TDMA phase be?

iii) Who should be the master node?

iv) What if nodes join and leave dynamically?

v) What message content should be sent in TDMA and control phases?

6.2.3.1 Protocol Time Setting

For the first two questions, it depends on how fast a message can be sent, and how large the time drift of the node is. We did some studies with TelosB nodes and TinyOS. In TinyOS, there are two important events for a sender and a receiver, respectively. When a sender calls Send() method, it needs to prepare all the data for the radio to transmit. After that, an event (interrupt) called SFD (start-of-frame delimiter) is triggered, which represents the beginning of sending the first bit. After all the data are sent, there is a Sendone() event triggered, which represents the ending of this sending. Correspondingly, there is a SFD for a receiver, which represents the beginning of receiving the first bit of data, and a ReceiveDone() event which represents the ending of the receiving. The SFD timestamps for both the sender and the receiver are automatically recorded locally, but the sender's SFD timestamp cannot be included in this transmission. The most important thing is that the time difference of these two SFD events is usually less than $1\mu s$, which can be used for time synchronization.

By keeping sending different size packets and recording the timestamps for Send(), SFD (both sender and receiver sides), SendDone() and ReceiveDone(), we get the measurement of data transmission rate and the time drift. Figure 6.10 shows the time cost for channel

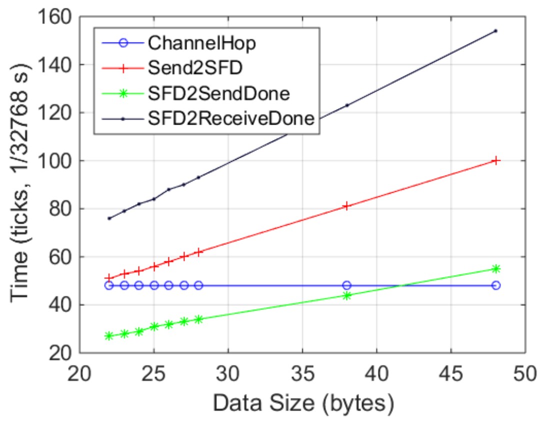


Figure 6.10: TelosB transmission and channel hopping delay

hopping, sending preparation, sending and receiving versus different data size. The deviation is less than 1 tick (1/32768s). Note that there are 6 bytes for the physical layer, 10 bytes for the MAC layer, and 2 bytes for the CRC. So the actual payload size is 18 bytes less than the total data size. We see that the channel hopping needs 48 ticks (1.5 ms), the sending preparation rate is 15.36 kB/s, the sending rate is 27.93 kB/s, and the receiving rate is 9.97 kB/s. Furthermore, by comparing the SFD of the sender side and that of the receiver side, we find the local time difference increases 1 tick for every 1.1 second, which is 27.7μs per second.

As aforementioned, the frequency hopping TDMA protocol has two phases: the TDMA phase and the control phase. In the TDMA phase, each node sends in its own specific time slot under a specific channel. All the nodes hop to the next channel at the same time after they all finish their time slots under the current channel. We call the period where nodes are under the same channel as **Channel Period**. When all the nodes finish all the channel periods, they start over from the first channel again. We call this time period as **TDMA Period**. Nodes experience several TDMA periods before they go to the control phase. Take Figure 6.9 for example, there are four nodes and four channels. In the first channel period, where the channel is Channel 11, Nodes 1, 2, 3 and 4 send one by one in their own TDMA time slots. After that, all nodes hop to Channel 16–the second channel period. When all nodes finish all channel periods 11, 16, 21 and 26, they start over from Channel 11. So we see that there are totally 6 TDMA periods in Figure 6.9. The control phase has the same channel as the first channel, and the time duration should be long enough so that even if a node is out of sync, it can always has chance to hop to the control phase's channel and get synced again.

Assuming the maximum payload size in the TDMA phase is s, the sending to SFD rate is r_s, the SFD to receiveDone rate is r_r, the channel hopping time is c, the drifting rate is d, and there are m channels and n nodes. Each TDMA sending time slot should satisfy

$$T_s \geq c + \frac{s}{r_s + r_r} \tag{6.46}$$

The control phase time should satisfy

$$T_c \geq mT_s \tag{6.47}$$

Hence, even with the maximum drifting, a node can still hop into the same channel as the control phase to get time synced. Furthermore, the TDMA phase cannot be too long, otherwise because of time drifting, nodes may hop to different channels and thus cannot receive the packets from a sender for a while. For simplicity, we do not allow the node's time to drift bigger than a half of a TDMA sending time slot T_s. So it should satisfy

$$\begin{aligned} pnmdTs &\leq \frac{T_s}{2} \\ \Rightarrow p &\leq \frac{1}{2nmd} \end{aligned} \tag{6.48}$$

where p is the number of periods.

Take Figure 6.9 as example and use the above experimental result, we have $n = 4$, $m = 4$, $r_s = 15.36kB/s$, $r_r = 9.97kB$, $c = 1.5ms$ and $d = 27.7\mu s/s$. Assuming in each TDMA sending time slot, each node only sends its node id, which is 4 bytes, we have $T_s \geq 2.3ms$, $T_c \geq 9.4ms$, and $p \leq 11282.2$.

6.2.3.2 Master Node Selection

For Question iii), we should consider the initial time when no node is a master node, the case when multiple nodes claim they are the master, and the case when the master node leaves. The rule is simple: a node treats the node with the smallest node id as the master node, keeps this master node id in the local memory, and sends control message if the master node id is itself in the control phase. The master node id variable changes in the following conditions:

- initially, each node sets the master node to itself.
- If Node C is the master node of Node A, and Node A receives a message from Node B whose id is smaller than Node C, then Node A sets its master node to Node B.
- In the control phase, if a node does not hear from the master node for a half time of the control phase, it sets itself as the master node and begins to broadcast control messages.

6.2.3.3 Network Dynamics and Protocol Packet format

Questions iv) and v) should be answered together. Questions iv) is about the network dynamics, and it is solved by the carefully designed packet format and the protocol behaviors.

Nodes may leave or join the network dynamically, and be out of sync at any time. The number of the participant nodes decides the time of the channel period and the T-DMA period. Therefore, to track these dynamics, each node keeps the following variables in its memory {node_id, master_node_id, phase_mode, slot_id, known_node_id_list[], node_msg_num_list[], channel_list[], tdma_period_num}. node_id and master_node_id are the node itself id and the current master node id, respectively. phase_mode indicates that currently it is in the TDMA phase (0) or control phase (1). slot_id means which sending time slot the node is current in (there are totally $n \times m \times p$ slots). known_node_id_list is the list of all the nodes it hears in this TDMA phase and the control phase. node_msg_num[]

array records how many messages it receives from each of the known nodes. channel_list[] is the channel order list it needs to hop. tdma_period_num is the number of TDMA periods it needs to experience.

The format of the TDMA message is:

16	1	15	64	TBD
sender_id	phase_mode	slot_id	seq	[content]

The format of the control message is:

16	1	64	16	TBD	
node_id	phase_mode	time_to_tdma	node_num	node_id_list[]	
8	TBD	8	64	64	TBD
channel_num	channel_list[]	tdma_period_num	last_sdf_timestamp	seq	[content]

seq is the packet sequence number. *time_to_tdma* is the time to the beginning of the next TDMA cycle. *last_sdf_timestamp* is the last sending SDF timestamp (since the current SDF time cannot be included in the current sending message). Both *time_to_tdma* and *last_sdf_timestamp* are for time synchronization purposes.

We define one TDMA phase plus one control phase as one cycle. During one cycle, each node records the node ids it hears and the number of messages it receives from each of these nodes. If the message number of a hearing node is bigger than some threshold t_m, then this node is considered as a participant. The master node needs to do so in order to send the control message to coordinate the next cycle. The other nodes also need to do so in order to replace the master if the master node leaves.

By using this protocol, if the master node is in the control phase, any node, as long as it is in the same channel of the control phase (regardless which phase it is in), gets the time synchronized through knowing the time difference between the local clock and the master's clock, and also knows how soon the next cycle begins. Furthermore, with predefined (or dynamically decided) channels and the TDMA period number, and the collected information about the participants, master decides the full sending order of the next cycle. By receiving the control message from the master node, each node also knows its sending time slots of

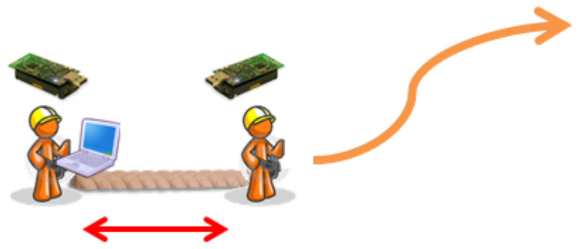


Figure 6.11: The RSS to distance estimation mobile case setting

the next cycle.

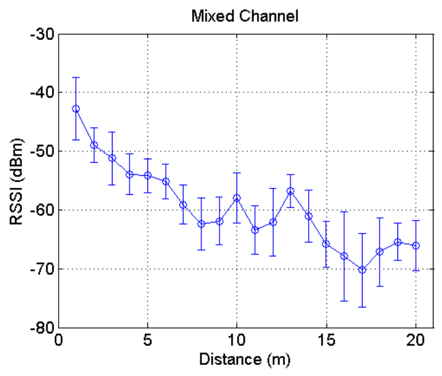
6.2.4 Evaluation

This section presents the experimental results for RSS to distance estimation. We first give the experimental configuration. Then we compare the estimation accuracies between using one channel and using the frequency hopping TDMA protocol under the mobile case. Finally, the performances of different filters described in Section 6.2.2 are evaluated.

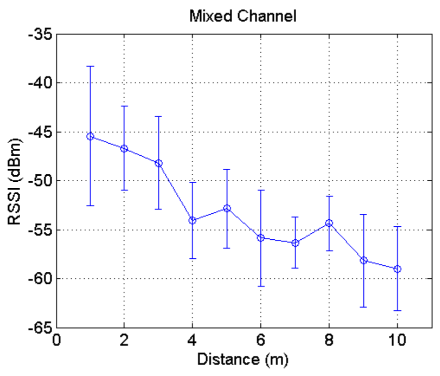
6.2.4.1 Configuration

TelosB [13] motes are used in our experiment. All data are collected through a serial port from TelosB to a laptop. The mote is put on the top of people's head in order to avoid the body shadowing effect. For one channel experiment, the sending rate is 50 packets/second, and the testing channels are 11, 16, 21 and 26. For the frequency hopping TDMA case, the sending time slot is 10ms, the selected channels are 11, 16, 21 and 26, the number of TDMA period is 12, and the control phase is 60ms with the sending rate 10ms.

For the mobile case, in order to study the estimation accuracy under a specific distance D, two people are tied with a rope of a fixed length D. When two people are walking, they always keep the rope tight so that their distance is fixed. Figure 6.11 illustrates the setting. Furthermore, in order to know which environment the RSS value is collected, the people in the experiment press a number key on the laptop whenever he/she enters a new environment. This operation inputs a special record in the RSS series data, and thus we



(a) Frequency Hopping RSS in Corridor



(b) Frequency Hopping RSS in Studying Lobby

Figure 6.12: Frequency Hopping TDMA RSS in Corridor and Studying Lobby

know the environment when we post process the RSS values. For example, if a person enters the corridor, "1" is pressed. If it is lobby, "2" is pressed. So in the RSS records, a line of "1,1,1,1,1,1,1,1" and a line of "2,2,2,2,2,2,2,2" are inserted. So we know all RSS values between the line of "1,1,1,1,1,1,1,1" and the line of "2,2,2,2,2,2,2,2" are collected in the corridor environment.

All the experiments were conducted inside the building of the department of computer science (Rice Hall) of University of Virginia. For both the static case and mobile case, the experiment covers the environments of corridors, lobbies with crowd desks and chairs, basements and stairs.

6.2.4.2 One Channel versus Channel Hopping

Figures 6.5 and 6.6 show the static RSS versus distance in the corridor and studying lobby environments, respectively. We see that the RSS mean value fluctuates as distance increases. By using the Frequency hopping TDMA protocol, the RSS in the same environments is studied. The results are shown in Figure 6.12. The first observation is that the variation of each point is bigger than that of Figures 6.5 and 6.6. This is because the RSS values of all channels are collected (the RSS values under different channels differ a lot). However, the trend of the mean value of RSS is more stable than that in Figures 6.5 and 6.6. Although

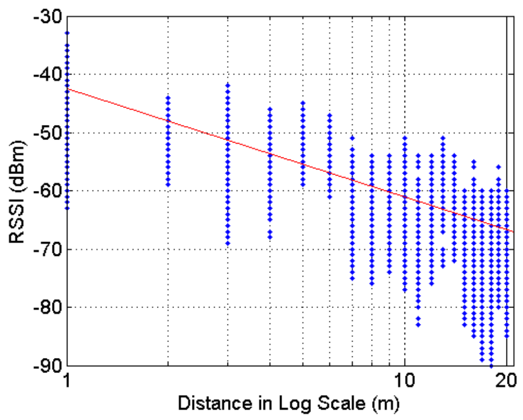


Figure 6.13: Linear regression fitting for multi-channel RSS values versus distances

it is still not monotonically decreasing as the distance increases, the overall curve has this trend.

In order to map RSS to distance, the parameters Π_0, Δ_0 and n_p in Equation 6.2 are needed. By collecting all RSS values in different distances, the linear regression is used to estimate these values. Figure 6.13 shows the fitting result, where we get $\Pi_0 = -42.49dBm$, $\Delta_0 = 1m$ and $\hat{n}_p = 1.86$. Another parameter is σ_{dB} in Equation 6.3, which represents the deviation of the RSS value. Based on our experimental data, we have $\sigma_{dB} = 5.05dB$. Note that these parameters are estimated based on all the data of different channels in different distances under different environments. Therefore, it represents the property of the indoor environment of Rice Hall.

After having these parameters' estimation, we study the RSS to distance estimation under the mobile case. The hardware setting is as Figure 6.11 shows. The ropes between the two people are made in different lengths in order to test the performance of the estimator under different distances. The two people in the experiment keep the rope tight and walk through the predefined routes, which covers different environments, such as the corridor, the basement, the stairs and the lobby.

To have better performance, we first filter the RSS series with a moving average window, and then feed them into the estimator. Figure 6.14 shows an example of such an RSS series.

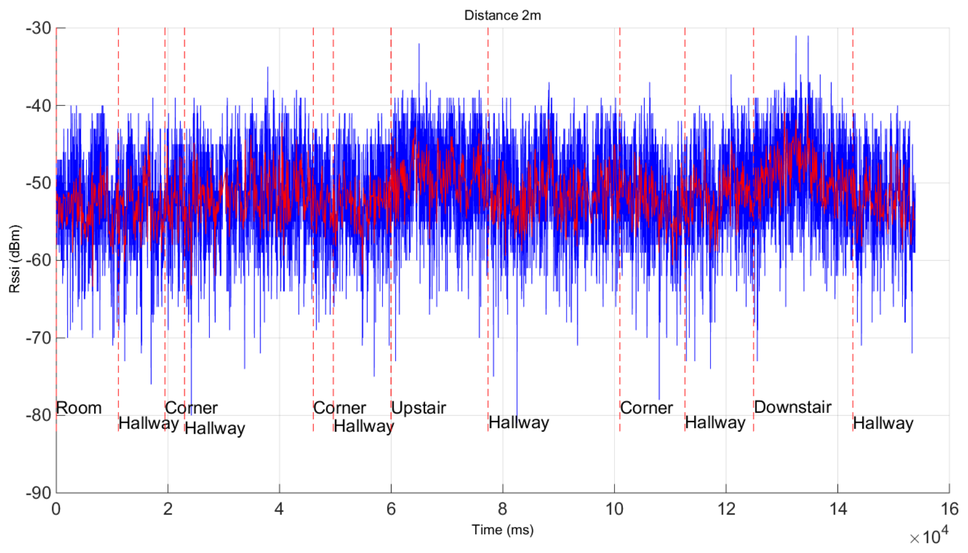


Figure 6.14: RSS series of distance 2m with frequency hopping TDMA

The data are collected with the distance of 2m using the frequency hopping TDMA protocol. We see that different parts of the series are tagged with different environments. An moving average window with the size of 5 samples are applied. The result (red lines) is more stable than the original one (blue lines).

After the averaging, we feed these RSS values into the maximum likelihood distance estimator (see 6.2.2.1) for different channels. Figure 6.15 shows the estimation error for different channels and the channel hopping. For a single channel, Channel 11 performs worst, while the Channel 26 performs best. For the frequency hopping TDMA, it outperforms Channels 11 and 16, but is worse than Channels 21 and 26. The reason for the good performance of channels 21 and 26 is because they are at the edge of the whole 802.14.5 channel spectrum, which does not have any (or have a little) contention with Wifi signals. Figure 6.16 shows the Wifi signals in Rice Hall. We see that Wifi Channel 1 has the strongest signal strength in the building, which is at the same frequency band as the Channel 11 of 802.15.4, and Channel 16 is also impacted by the other channels of Wifi (see Figure 6.3). For frequency hopping TDMA protocol, since it uses all the data from all channels, the accuracy is in the middle. At the distance 1m, it has the estimation error of 0.9588m,

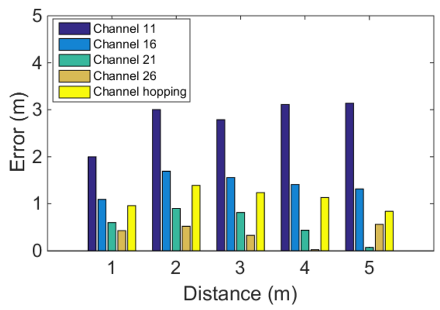
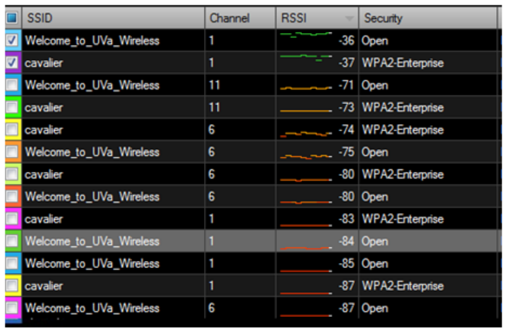


Figure 6.15: MLE estimation accuracy with different distances under different channels



	SSID	Channel	RSSI	Security
✓	Welcome_to_UVa_Wireless	1	-36	Open
✓	cavalier	1	-37	WPA2-Enterprise
	Welcome_to_UVa_Wireless	11	-71	Open
	cavalier	11	-73	WPA2-Enterprise
	cavalier	6	-74	WPA2-Enterprise
	Welcome_to_UVa_Wireless	6	-75	Open
	cavalier	6	-80	WPA2-Enterprise
	Welcome_to_UVa_Wireless	6	-80	Open
	cavalier	1	-83	WPA2-Enterprise
	Welcome_to_UVa_Wireless	1	-84	Open
	Welcome_to_UVa_Wireless	1	-85	Open
	cavalier	1	-87	WPA2-Enterprise
	Welcome_to_UVa_Wireless	6	-87	Open

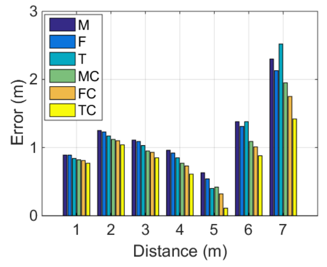
Figure 6.16: Wifi signals in Rice Hall

and at the distance 5m, the error is 0.8386m. If we don't know the signal spectrum of a building, the frequency hopping TDMA protocol is the safest choice, and its performance is acceptable.

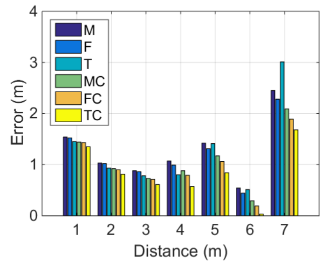
6.2.4.3 Estimator Performances Evaluation

Several experiments are done in order to compare the performance of the estimators proposed in Section 6.2.2. First experiment is to compare these estimators in the relative short distances under different environments. The distances are from 1m to 7m, and the environments include the corridor and the basement, respectively. We repeat 3 times for each environment and collet the RSS data, and then post process them with different estimators. The second experiment is to test the performance of these estimators with long distances. The distances are 10m, 15m and 20m. Because the flattened maximum likelihood distance estimator treats all low RSS values the same, and the truncated maximum likelihood distance estimator drops the low RSS values, it is necessary to study their performances under long distances which usually have low RSS values. In these experiments, the thresholds in Equations 6.20 and 6.33 are set to $T = -65dBm$. Note that the frequency hopping TDMA protocol is turned on for all these experiments.

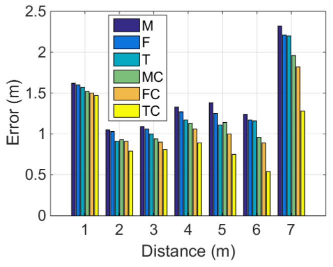
Figures 6.17(a), 6.17(b) and 6.17(c) show the estimation accuracy for 6 estimators in the corridor environment. "M" represents the maximum likelihood distance estimator, "F" represents the flattened maximum likelihood distance estimator, "T" represents the



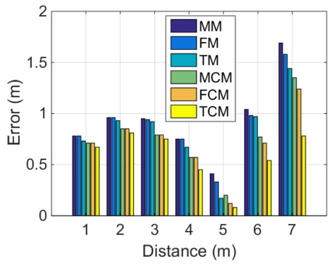
(a) Estimation accuracy in the corridor 1



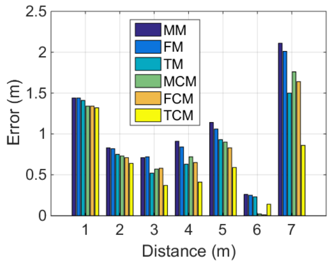
(b) Estimation accuracy in the corridor 2



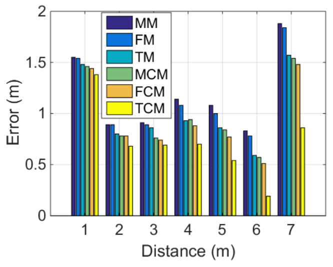
(c) Estimation accuracy in the corridor 3



(d) Estimation with post median filter accuracy in the corridor 1



(e) Estimation with post median filter accuracy in the corridor 2

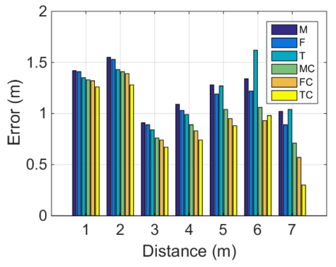


(f) Estimation with post median filter accuracy in the corridor 3

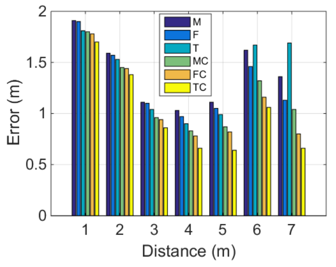
Figure 6.17: The estimation accuracy of different filters in the corridor environment.

truncated maximum likelihood distance estimator, “MC” is the corrected version of the estimator “M”, “FC” is the corrected version of the estimator “F”, and “TC” is the corrected version of the estimator “T”. The figures shows that the corrected version of the estimators are better than the uncorrected versions. For the corrected estimators, the best estimator is “TC”, then is “FC”, and the last is “MC”. “TC” achieves sub-meter accuracy at most time.

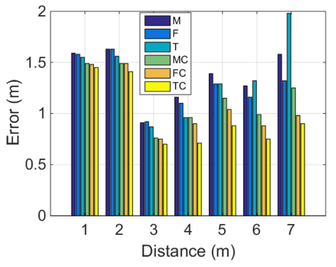
Figures 6.17(d), 6.17(e) and 6.17(f) show the results of all these estimators after a post median filter processing. That is when the data are collected, they are like the blue curves in Figure 6.14. After the first moving averaging processing, we have the red line in Figure 6.14. Then these data are fed into different filters, and the output is a series of distance estimations, whose mean error is shown in Figures 6.17(a), 6.17(b) and 6.17(c). After that, these distance estimates are further processed with a moving median filter, whose window size is 3 samples. From Figures 6.17(d), 6.17(e) and 6.17(f) we see that the post median



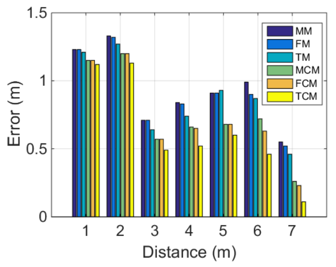
(a) Estimation accuracy in the basement 1



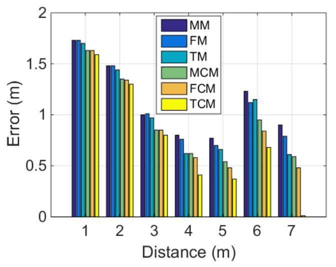
(b) Estimation accuracy in the basement 2



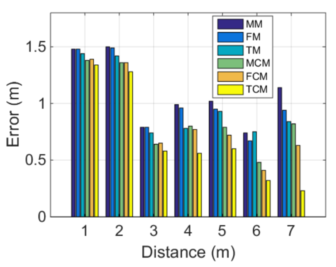
(c) Estimation accuracy in the basement 3



(d) Estimation with post median filter accuracy in the basement 1



(e) Estimation with post median filter accuracy in the basement 2



(f) Estimation with post median filter accuracy in the basement 3

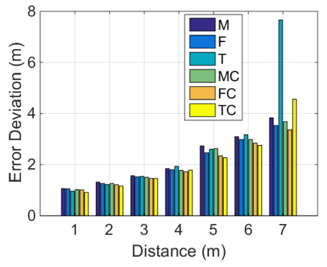
Figure 6.18: The estimation accuracy of different filters in the basement environment.

filter processed results are even better, and the performance ranking is the same as that in Figures 6.17(a), 6.17(b) and 6.17(c).

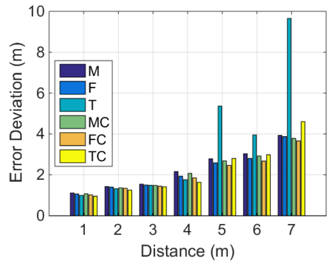
Figure 6.18 shows the RSS to distance estimation errors of different estimators in the basement environment. The performance ranking of different estimators is the same as in Figure 6.17. The errors of "TCM" are all below 1.5m, and most of the time are below 1m.

Figures 6.17 and 6.20 show the error deviations corresponding to the mean errors in Figures 6.17 and 6.18. We see that as the estimated distance increases, the deviation also increases. More importantly, the estimator "TCM" has not only the best mean estimation error, but also the smallest deviation of the estimation.

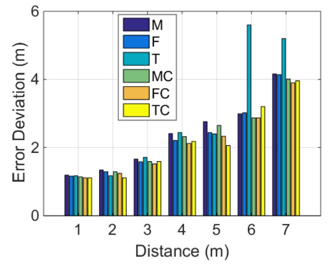
Figure 6.21 shows the estimation errors for different estimators at long distances. "NA" means unavailable, that is the estimator cannot output a valid estimation. "NA" is better than an inaccurate estimation since it would not pollute the data. From the figure we see that the estimators are very inaccurate at long distances, which proves our claim in Section



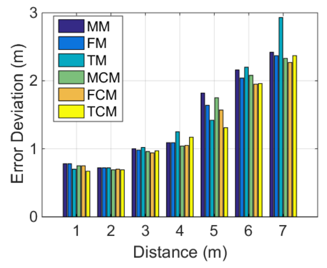
(a) Estimation error deviation in the corridor 1



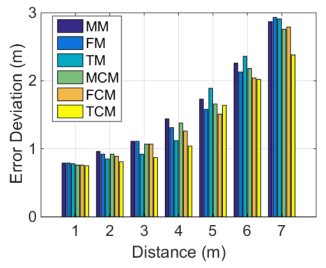
(b) Estimation error deviation in the corridor 2



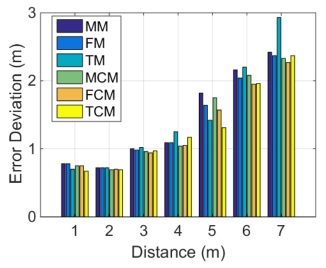
(c) Estimation error deviation in the corridor 3



(d) Estimation error deviation (with post median filter) in the corridor 1

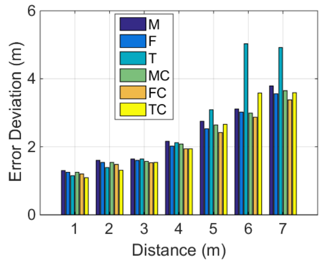


(e) Estimation error deviation (with post median filter) in the corridor 2

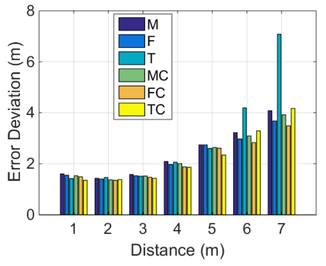


(f) Estimation error deviation (with post median filter) in the corridor 3

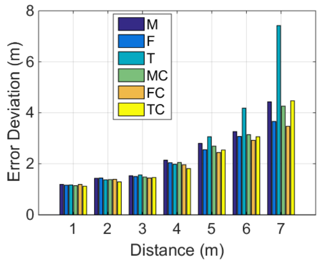
Figure 6.19: The estimation deviations in the corridor



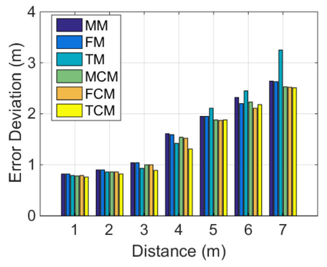
(a) Estimation error deviation in the basement 1



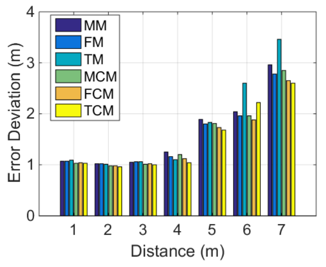
(b) Estimation error deviation in the basement 2



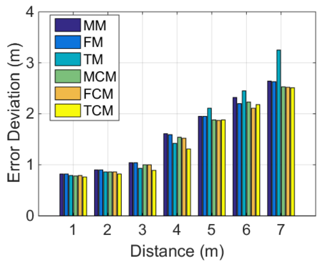
(c) Estimation error deviation in the basement 3



(d) Estimation error deviation (with post median filter) in the basement 1

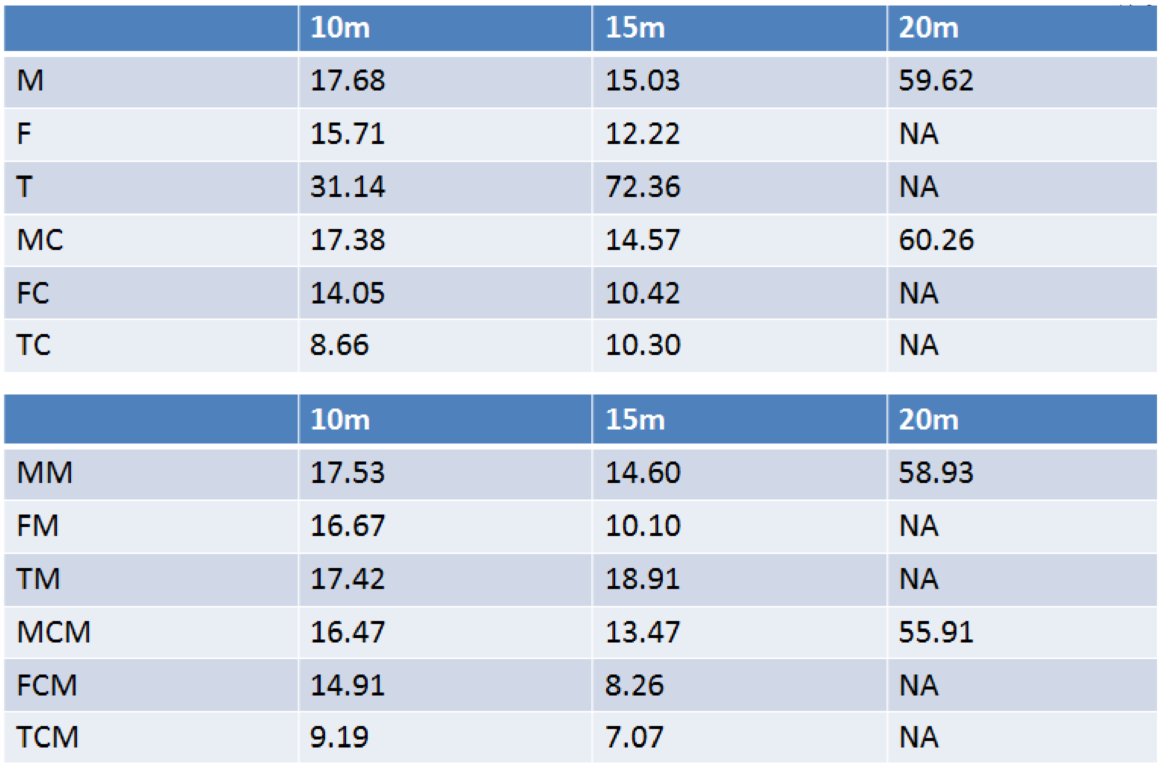


(e) Estimation error deviation (with post median filter) in the basement 2



(f) Estimation error deviation (with post median filter) in the basement 3

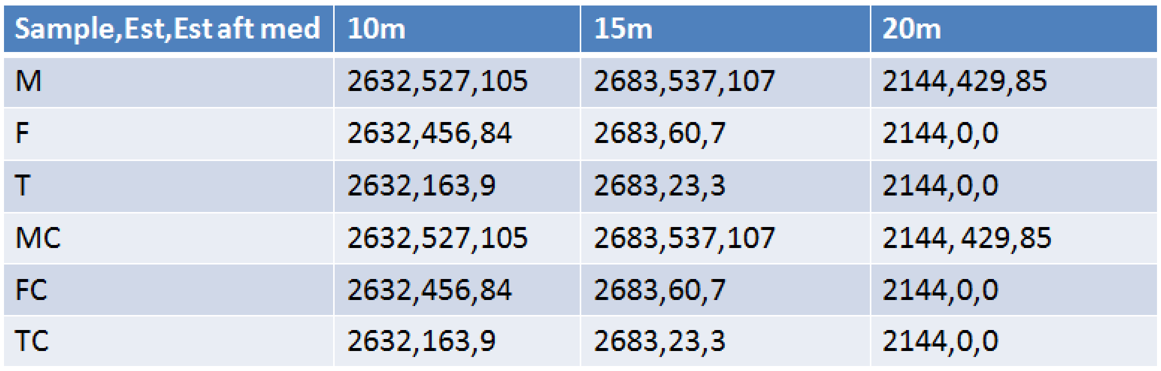
Figure 6.20: The estimation deviations in the basement



	10m	15m	20m
M	17.68	15.03	59.62
F	15.71	12.22	NA
T	31.14	72.36	NA
MC	17.38	14.57	60.26
FC	14.05	10.42	NA
TC	8.66	10.30	NA

	10m	15m	20m
MM	17.53	14.60	58.93
FM	16.67	10.10	NA
TM	17.42	18.91	NA
MCM	16.47	13.47	55.91
FCM	14.91	8.26	NA
TCM	9.19	7.07	NA

Figure 6.21: RSS to distance estimation error at long distances



Sample,Est,Est aft med	10m	15m	20m
M	2632,527,105	2683,537,107	2144,429,85
F	2632,456,84	2683,60,7	2144,0,0
T	2632,163,9	2683,23,3	2144,0,0
MC	2632,527,105	2683,537,107	2144, 429,85
FC	2632,456,84	2683,60,7	2144,0,0
TC	2632,163,9	2683,23,3	2144,0,0

Figure 6.22: Valid estimation number of RSS to distance mapping at long distances

6.2.2–we should restrict the estimation to short distances. Even for long distances, “TC” and “TCM” outperform the other estimators.

Figure 6.22 shows the number of outputs at each stage during the data processing. For each cell in the table, there are three numbers. The first number shows the number of the input RSS values; the second number shows the number of the outputs from each estimator;

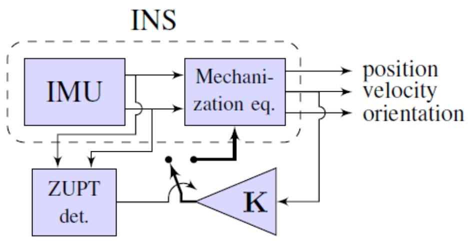
and the third number is the number after the post median filter processing. For estimators "F" and "T", sometimes they cannot find the RSS values above the threshold in a moving window. That is why they may output much less number of estimations at long distances. Furthermore, during the median filter processing, if there is not enough samples (less than 3 in our setting) in the current window, we output NA. Therefore, the number of outputs is further reduced at this stage. We see that "TC" outputs least, and the followed is "FC". All these design is to follow one principle: **no estimation is better than an inaccurate estimation**.

6.3 Foot-mounted Inertial Navigation System

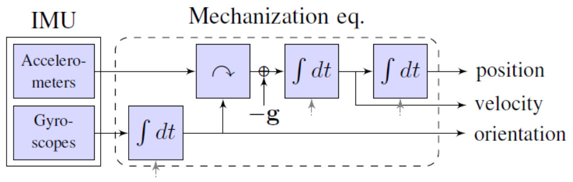
As presented in Section 3.1.1, there are two types of location related measurements–the strong measurement and the weak measurement. A node is localizable if it can be reached by a strong path (an undirected path which only consists of strong measurements) starting from an anchor node in a measurement graph. The above RSS to distance measurement is a weak measurement. In order to localize the mobile agent, strong measurements are needed.

Since one of the goals is to be fully self-contained, the IMU based solution is the best fit. An IMU usually includes a 3-axis accelerometer and a 3-axis gyroscope (some PC boards may also integrate a compass chip and an altimeter chip). Based on real-time acceleration and angular rate measurements, a strong measurement–the displacement between two time points–can be obtained theoretically by coordinate transformation and integration. The best advantage of the IMU is that it is a fully self-contained solution, and it could be a very tiny and cheap device if it is MEMS based. However, the downside for the tiny wearable IMU devices is that the acceleration and angular rate measurements are usually very noisy, which would result in hundreds of meters error in one minute if only the naive integration method is used.

Therefore, in order to make the tiny IMU device usable, some efficient filter is needed to



(a) ZUPT system diagram



(b) Inertial navigation diagram

Figure 6.23: ZUPT inertial navigation system diagram [243]

correct the noise. For the pedestrian tracking problem, different filters are developed, which require different mounting positions on human body (see Section 2.3.1). The most effective one is to mount the IMU on the foot, and use the zero-velocity-update (ZUPT) methodology to track a pedestrian (can also track running, crawling, etc.). The idea of ZUPT is very simple, the foot velocity should be zero in its stance phase during the walking. If the estimation of the velocity is not zero, the error needs to be fed back into the module for correction. Under the naive implementation, the error growth rate is of cubic of time. However, by using ZUPT, the error growth rate is reduced to be linear in the number of steps.

6.3.1 Zero Velocity Update Methodology

There are a number of works about ZUPT foot mounted inertial navigation system. We adopt the implementation of the OpenShoe project [243]. As shown in Figure 6.23(a), there are three components in the system–the IMU hardware, the ZUPT detector and the mechanization module. The input of the system is from the IMU, which includes the acceleration and angular rate measurements in the sensor board coordinate frame. The outputs are position, velocity and orientation in the navigation coordinate frame. The acceleration and angular rate measurements are used by the mechanization component and the ZUPT detector. For the mechanization component, the inertial measurements are used to update the velocity, position and orientation by coordinate transformation and integration under Newton's Law. For the ZUPT detector, the inertial measurements are used to detect the stance event during the walking (or other actions). If a stance event is detected, the current velocity is the error (since the correct velocity should be 0), and thus fed into the mechanization component to correct the accumulated error of this step.

The mechanization component is illustrated in Figure 6.23(b). The angular rate from the gyroscope is first integrated to get the 3D angle change during the small time frame δt_k. This angle change is then used to update the coordinate rotation matrix and the current orientation. The acceleration under the sensor board coordinate frame is first transferred to the navigation coordinate frame by using the coordinate rotation matrix. Then we get the acceleration of the foot under the navigation coordinate frame by subtracting the gravity. After that, the velocity change is calculated by integrating the acceleration under the navigation coordinate frame, and the displacement is further calculated by integrating this velocity.

6.3.1.1 The Mechanization Model

The mechanization module is represented as follows:

$$\begin{bmatrix} \mathbf{x}_{k|k-1} \\ \mathbf{v}_{k|k-1} \\ \mathbf{q}_{k|k-1} \end{bmatrix} = \begin{bmatrix} \mathbf{x}_{k-1|k-1} + \mathbf{v}_{k-1|k-1}\delta t_k \\ \mathbf{v}_{k-1|k-1} + (\mathbf{q}_{k-1|k-1}\mathbf{f}_k\mathbf{q}_{k-1|k-1}^{-1} - \mathbf{g})\delta t_k \\ \mathbf{\Omega}(\omega_k \delta t_k)\mathbf{q}_{k-1|k-1} \end{bmatrix} \tag{6.49}$$

where k is the time index, $k|k-1$ and $k|k$ represent the prediction step and update step of the Kalman filter respectively, δt_k is the time difference between the measurement instants, $\mathbf{x}_k$ is the position, $\mathbf{v}_k$ is the velocity, and $\mathbf{q}_k$ is the quaternion describing the orientation of the system relative to the navigation coordinate frame, $\mathbf{f}_k$ is the accelerometer reading, g is the gravity, ω_k is the gyroscope reading, and $\mathbf{\Omega}(\cdot)$ is the quaternion update matrix.

In aided navigation, an error state vector is used. For ZUPT, the error state is $\delta \mathbf{s}_k = [\delta \mathbf{x}_k, \delta \mathbf{v}_k, \delta \varphi_k]$, where $\delta \mathbf{x}_k$ is the position error state, $\delta \mathbf{v}_k$ is the velocity error state, and $\delta \varphi_k$ is the orientation error state in terms of roll, pitch and yaw. Some work also includes the error of angular rate and the acceleration bias [195], however, it has been proved that it does not gain much in practice [244]. Using this error state, the linearized transition model is:

$$\delta \mathbf{s}_{k|k-1} = \mathbf{\Phi}_k \delta \mathbf{s}_{k-1|k-1} + \mathbf{w}_{k-1} \tag{6.50}$$

The transition matrix $\mathbf{\Phi}_k$ is

$$\mathbf{\Phi}_k = \begin{bmatrix} \mathbf{I} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} & \mathbf{S}(\mathbf{f}_k^n) \\ \mathbf{0} & \mathbf{0} & \mathbf{I} \end{bmatrix} \cdot \delta t_k \tag{6.51}$$

where $\mathbf{S}(\mathbf{f}_k^n)$ is the skew matrix of the acceleration under the navigation coordinate frame.

$$\mathbf{S}(\mathbf{f}_k^n) = \begin{bmatrix} 0 & -\mathbf{f}_k^n(3) & \mathbf{f}_k^n(2) \\ \mathbf{f}_k^n(3) & 0 & -\mathbf{f}_k^n(1) \\ -\mathbf{f}_k^n(2) & \mathbf{f}_k^n(1) & 0 \end{bmatrix} \tag{6.52}$$

$\mathbf{w}_{k-1}$ is the process noise with the covariance $\mathbf{Q}_{k-1} = E(\mathbf{w}_{k-1}\mathbf{w}_{k-1}^T)$. The error covariance matrix $\mathbf{P}_{k|k-1}$ is calculated as

$$\mathbf{P}_{k|k-1} = \mathbf{\Phi}_{k-1}\mathbf{P}_{k-1|k-1}\mathbf{\Phi}_{k-1}^T + \mathbf{Q}_{k-1} \tag{6.53}$$

The measurement model is

$$\mathbf{z}_k = \mathbf{H}_k \delta \mathbf{s}_{k|k} + \mathbf{n}_k \tag{6.54}$$

$\mathbf{z}_k$ is the measurement. In ZUPT case, $\mathbf{z}_k = \mathbf{v}_{k|k-1}$. $\mathbf{H}_k$ is the measurement matrix:

$$\mathbf{H}_k = [\mathbf{0}, \mathbf{I}, \mathbf{0}] \tag{6.55}$$

$\mathbf{n}_k$ is the measurement noise with covariance $\mathbf{R}_k = E(\mathbf{n}_k \mathbf{n}_k^T)$.

When there is a stance phase detected, the following equation is used to update the error state vector:

$$\delta \mathbf{s}_{k|k} = \delta \mathbf{s}_{k|k-1} + \mathbf{K}_k(\mathbf{z}_k - \mathbf{H}_k \delta \mathbf{s}_{k|k-1}) = \mathbf{K}_k \mathbf{z}_k \tag{6.56}$$

The above equation uses $\delta \mathbf{s}_{k|k-1} = 0$, because each time after the update, these errors are fed back to the state of position, velocity and orientation for correction, and thus reset to zeros. $\mathbf{K}_k$ is the Kalman gain, and is calculated as:

$$\mathbf{K}_k = \mathbf{P}_{k|k-1} \mathbf{H}^T (\mathbf{H} \mathbf{P}_{k|k-1} \mathbf{H}^T + \mathbf{R}_k)^{-1} \tag{6.57}$$

The error covariance matrix $\mathbf{P}_{k|k}$ is calculated as:

$$\mathbf{P}_{k|k} = (\mathbf{I} - \mathbf{K}_k \mathbf{H}) \mathbf{P}_{k|k-1} \tag{6.58}$$

6.3.1.2 The Zero Velocity Detector

In order to correct the estimation error, the stance event needs to be detected. It is arguably that the zero velocity detector might be more important than the IMU hardware performance [244]. The work [194] gives the evaluations for several different zero velocity detectors. The Acceleration Magnitude Detector (MAG) is based on the acceleration energy, i.e. when the energy of the 3D acceleration is comparable to the gravity, it assumes it is a stance event. The Angular Rate Energy Detector (ARE) is based on the energy of the 3D angular rate. If it is closed to 0, then it assumes it is a stance event. The Acceleration Moving Variance Detector (MV) is based on the variance of the acceleration. When it is less than a threshold, then a stance event is assumed. The last detector is called The S-tance Hypothesis Optimal Detector (SHOE), which uses both acceleration and angular rate measurements, and is proved to be the optimal. Our implementation adopts this detector, whose formula is written as follows [194]:

$$T(\{\mathbf{f}_k^c\}_{k=n}^{n+W-1}, \{\omega_k^c\}_{k=n}^{n+W-1}) = \frac{1}{W} \sum_{k=n}^{n+W-1} \left(\frac{1}{\sigma_f^2} \left\| \mathbf{f}_k^c - g \frac{\bar{\mathbf{f}}_k^c}{\|\bar{\mathbf{f}}_k^c\|} \right\|^2 + \frac{1}{\sigma_\omega^2} \|\omega_k^c\|^2 \right) \tag{6.59}$$

where $\mathbf{f}_k^c$ is the corrected (subtracting the bias) acceleration measurement at time k, ω_k^c is the corrected angular measurement at time k, $\{\cdot\}_{k=n}^{n+W-1}$ represents a time series of a value with window size W, $\bar{\mathbf{f}}_k^c$ is the average of $\mathbf{f}_k^c$ in the window, $\| \cdot \|$ is the norm operator, and σ_f and σ_ω are scalars representing the deviation of the noise of the acceleration and the angular rate.

6.3.2 IMU Calibration and ZUPT Detector Threshold Setting

Since a tiny and cheap IMU device is used, carefully modeling its properties in a probabilistic way is important to the final estimation results. Additionally, the IMU is also used as a ZUPT detector, and thus the setting of the threshold of the ZUPT detector is also important.

6.3.2.1 IMU Characterization

An IMU usually consists of a 3-axis accelerometer and a 3-axis gyroscope. There are different types of errors for its outputs.

- Constant bias: for a gyroscope, it is the average output when it stays static; for an accelerometer, it is the average difference between its output and the local gravity. It causes the angular error to grow linearly with time for the gyroscope bias, and the position error to grow quadratically with time for the accelerometer bias.
- White noise: the name is self explanatory. The white noise is usually modeled as a normal distribution. It introduces a zero-mean random walk error into the angular change for the gyroscope (the angular error deviation is proportional to the square root of the time, i.e. $\sigma_\theta(t) \propto \sqrt{t}$), and a zero-mean second order random walk error into the displacement for the accelerometer (the displacement error deviation has the

following relationship with the time: $\sigma_s(t) \propto t^{\frac{3}{2}}$)

- Bias stability: this error creates a random walk in the bias. Therefore, it introduces a second random walk in the angular error for the gyroscope, and a third random walk in the displacement error for the accelerometer.

- Scale factors: this error introduces a multiplicative error to the outputs, i.e. $m = (1 + \epsilon_s)z$, where m is the output, ϵ_s is the scale factor error, and z is the real value. This error cannot be measured when the real output is zero. This error introduces the same scale error to the final angular or displacement results.

- Alignment error: it means that the 3 axis of the sensor may not be strictly aligned perpendicularly. It can be modeled as $m = \mathbf{A} \cdot z$, where m is the 3-axis output vector, $\mathbf{A}$ is the error alignment matrix, and z is the real 3-axis vector.

- Linearities error: this error means that the output may not be linear with the real value. It is actually the nonlinear type of the scale factor. It usually happens when the sensor output is out of the normal range.

The last three error types, i.e. the scale factors, the alignment error and the linearities error, are called calibration errors. These errors are usually pre-corrected by manufactures. Therefore, the first three errors need to be characterized. For the constant bias, as explained above, for a gyroscope, it can be estimated by averaging its outputs when it is placed statically; for an accelerometer, it can be estimated by aligning the sensor in 3 different orientations, i.e. x-axis up, y-axis up and z-axis up, and averaging the outputs of the axes which are aligned horizontally.

For the white noise and the bias stability, a mathematical tool called Allan Variance (or Allan deviation) is able to capture them. The computation of Allan Variance is as follows. First take a long time sequence of samples, and divide them into bins of length t. t should not be too large so that there are at least 9 bins. Then calculate the average of all the bins $(\bar{a}(t)_1, \bar{a}(t)_2, \cdots, \bar{a}(t)_n)$, where n is the the number of the bins. After that the Allan

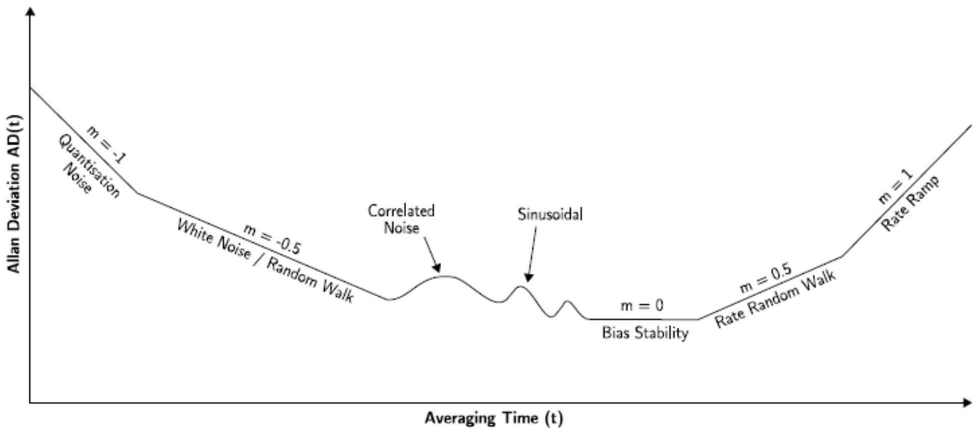


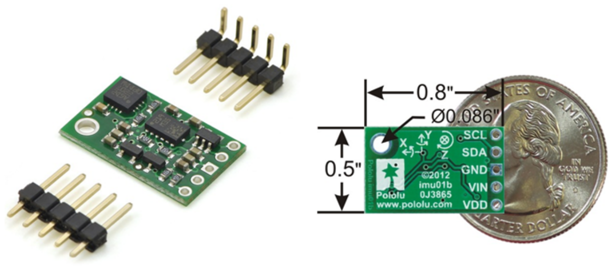
Figure 6.24: An example of log-log plot of Allan Deviation analysis results [14].

Variance is calculated as:

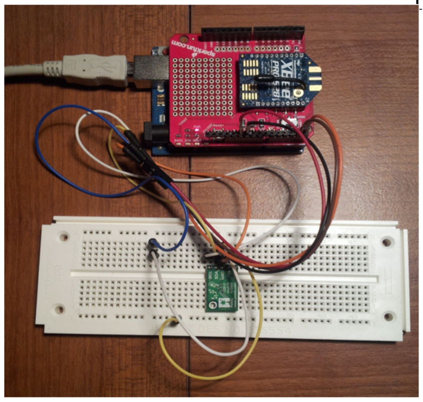
$$Al(t) = \frac{1}{2(n-1)} \sum_{i=2}^{n} (\bar{a}(t)_i - \bar{a}(t)_{i-1})^2 \tag{6.60}$$

The Allan Deviation is $Ad(t) = \sqrt{Al(t)}$. By using different t, we get different Allan Deviation values. Then all these points are plotted in the log-log scale. The different types of noise then can be characterized visually. Figure 6.24 shows an example of a log-log plot of Allan Deviation of some results. The white noise can be identified by fitting a straight line with the slope -0.5, and read the value at $t = 1$. The bias stability appears as the flat region around the minimum of the curve, and thus we can simply get the minimum value as the bias stability.

We use Allan Deviation to characterize the IMU MinIMU-9 v2 [245], which includes the gyroscope chip L3GD20 [246] and the accelerometer (with a magnetometer) chip LSM303DLHC [247]. The IMU hardware is shown in Figure 6.25(a). The data collection setup is shown in Figure 6.25(b), where an Arduino UNO board, a serial cable, a breadboard and the MinIMU-9 v2 are used. The data are collected for 7.6 hours at 50Hz sampling rate–totally 1,371,289 samples of each axis for the gyroscope/accelerometer. The Allan Deviation log-log scale plots are shown in Figure 6.26. With Allan Deviation reading and the bias we measured

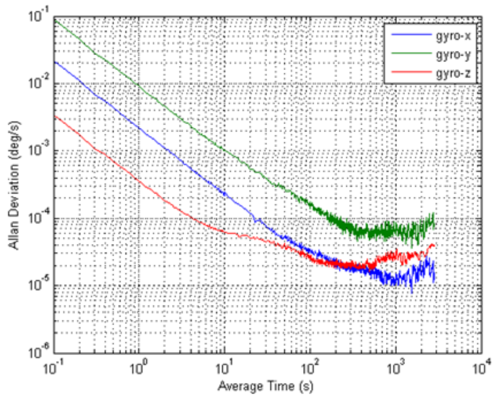


(a) MinImu-9 V2

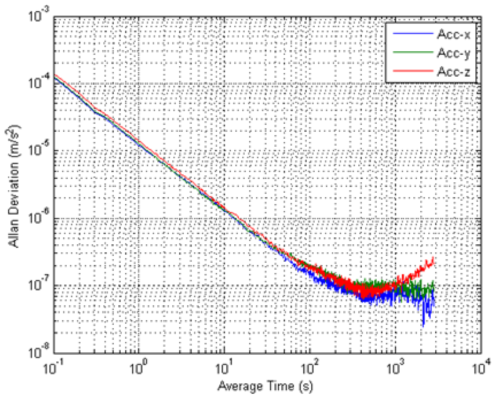


(b) Data collection setup

Figure 6.25: Data collection for MinImu-9 V2



(a) Allan Deviation log-log scale plot for the gyroscope



(b) Allan Deviation log-log scale plot for the accelerometer

Figure 6.26: Allan Deviation log-log scale plot for the gyroscope and the accelerometer

following the aforementioned way, Table 6.1 shows the bias, the white noise and the bias stability of one L3GD20 gyroscope; and Table 6.2 is for the LSM303DLHC accelerometer chip.

Using the above measured errors, we simulate and evaluate the impacts of these different errors. Based on Equation 6.49, we simulate the position estimation for a still device when applying different types of random errors, such as the bias, the white noise and the bias stability of the accelerometer and the gyroscope. Figure 6.27(a) shows the position estimation error for a static device for one minute with all these errors. The blue line is

Table 6.1: The characterization of one L3GD20 gyroscope

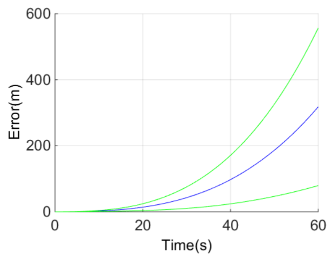
Error type	Data sheet value	Measured value
Bias	$\pm 75°/s$	$[-0.4641, 0.2455, -0.8077](°/s)$
White noise	$0.03°/\sqrt{s}$	$[-0.0466, 0.0958, -0.0190](°/\sqrt{s})$
Bias stability	NA	$[-0.0028, 0.006, -0.004](°/s)$

Table 6.2: The characterization of one LSM303DLHC accelerometer

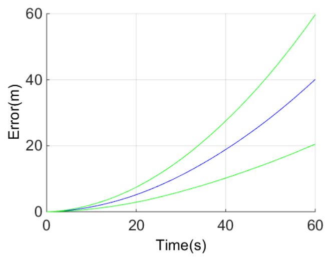
Error type	Data sheet value	Measured value
Bias	$\pm 0.06g$	$[0.0183, -0.0886, 0.036](g)$
White noise	NA	$[0.0035, 0.0036, 0.0038](g/\sqrt{s})$
Bias stability	NA	$[0.000157, 0.000237, 0.000252](g)$

the average drift for 100 runs, and the green lines represents the one deviation error bound. We see that for our device, in one minute, the naive inertial navigation implementation introduces around 320m drift on average. Figure 6.27(b) shows the estimation error with only accelerometer errors. We see that the error is much less than in Figure 6.27(a). This is because the gyroscope error creates the drift to the final position estimation cubically in time, while the accelerometer error only creates the error quadratically in time. Figures 6.27(c) and 6.27(d) show the drift for white noise and bias stability, respectively. We see that the drift caused by the white noise is bigger than the bias stability. In practice, we model the white noise with a random variable with the proper deviation σ_w, and do not model the bias stability since the ZUPT is sufficient to correct it.

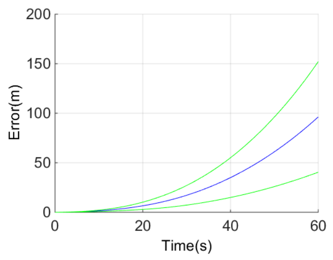
The above calibration is for a static device only. In practice, there are other factors impacting these parameters. For example, the dynamic error only appears when the inertial sensors are moving or rotating; the bias and alignment error would change if the sensor is tied to the human body tightly and gets deformed (in our case). Therefore, besides the above static calibration, there are several other ways to do the calibration. One way is to use a simple home-made calibration tool to decide the dynamic errors as described in the work [248]. Another method is that before a pedestrian starts walking, let the pedestrian stand still for tens of seconds to collect data so that the on site bias can be calculated. However, based on our experimental data, these pre-calibration methods are not accurate



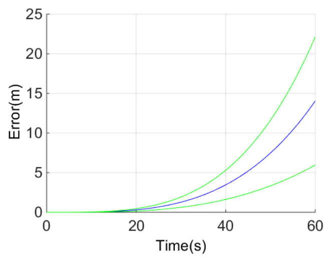
(a) With all errors of the accelerometer and the gyroscope



(b) With all errors of the accelerometer only



(c) With only white noises of both the accelerometer and the gyroscope



(d) With only the bias stability of both the accelerometer and the gyroscope

Figure 6.27: The simulated impact of different types of sensor errors to the position estimation. 50Hz, one minute and total 100 runs.

enough. Therefore, we choose to use a post-tuning method for these error parameters, i.e. we treat the calibration as a parameter estimation problem after we collect all the raw data.

Since the bias error of the gyroscope dominates the final error of the position estimation, and in practice we find that the above static calibration method is sufficient for estimating the accelerometer bias error, the post-tuning method is only interested in the gyroscope bias. Modeling the whole foot-mounted inertial navigation system as a black box, then the system with the gyroscope bias parameter can be written as

$$\hat{\mathbf{x}} = f(\mathbf{z}; \theta_\omega) \tag{6.61}$$

where $\hat{\mathbf{x}}$ is the time series of the position estimation, $f(\cdot)$ represents the inertial navigation black box function, $\mathbf{z}$ is the input vector which includes the time series of raw acceleration and raw gyroscope readings, and θ_ω is the unknown vector of the gyroscope bias to be tuned. Then the bias estimation can be written as an optimization problem:

$$\min_{\theta_\omega \in \mathbb{R}^3} \|\mathbf{x} - f(\mathbf{z}; \theta_\omega)\|^2 \tag{6.62}$$

where $\mathbf{x}$ is the true position series.

Since the ZUPT navigation function is too complex to obtain an analytic form, the iterative nonlinear optimization solver is chosen. One most straightforward solution is brute force. Assume the 3-axis bias is within the cubic range of $[-100, -100, -100]$ and $[100, 100, 100]$. We can first set a big step, e.g. 10, to search the whole space coarsely. Assuming it finally hits the point $[-10, 10, 20]$ as the minimum, then the step is reduced to 1, and the search space is shrunk to the cubic range of $[-30, -10, 0]$ and $[10, 30, 40]$. By using this method, the search space keeps reducing, and the steps become finer. Finally, some local minimum is found.

Other classic iterative nonlinear optimization solutions can be used too, such as genetic algorithms [249], the Nelder-Mead simplex algorithm [250] and the simulated annealing algorithm [251]. We choose the simulated annealing algorithm since it is based on the probabilistic model and can reach the global minimum theoretically. Matlab implements the simulated annealing algorithm with the function "simulannealbnd()". For more details of this algorithm, please refer to the work [251].

6.3.2.2 ZUPT Detector Threshold Setting

Another parameter is the threshold of the ZUPT detector, which is arguably more important than the hardware performance [244]. This parameter setting is not difficult: plot the outputs of the detector over time, and set the threshold just above the low part of the plot. The above optimization solver can be used to further refine the threshold. Figure 6.28 shows a part of the output of the ZUPT detector. The periodical peaks are foot strides, and the valleys are stance phases. In this figure, the threshold is set to 76,000.

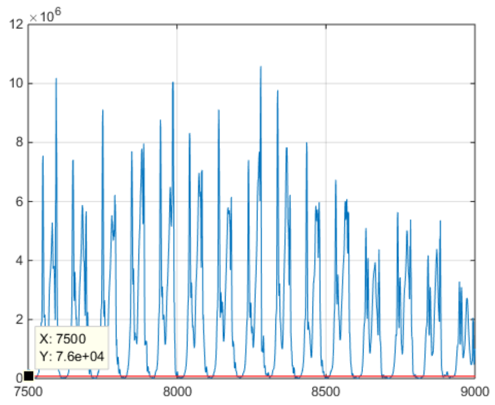


Figure 6.28: ZUPT threshold setting

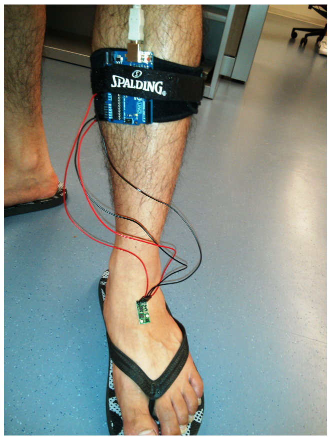
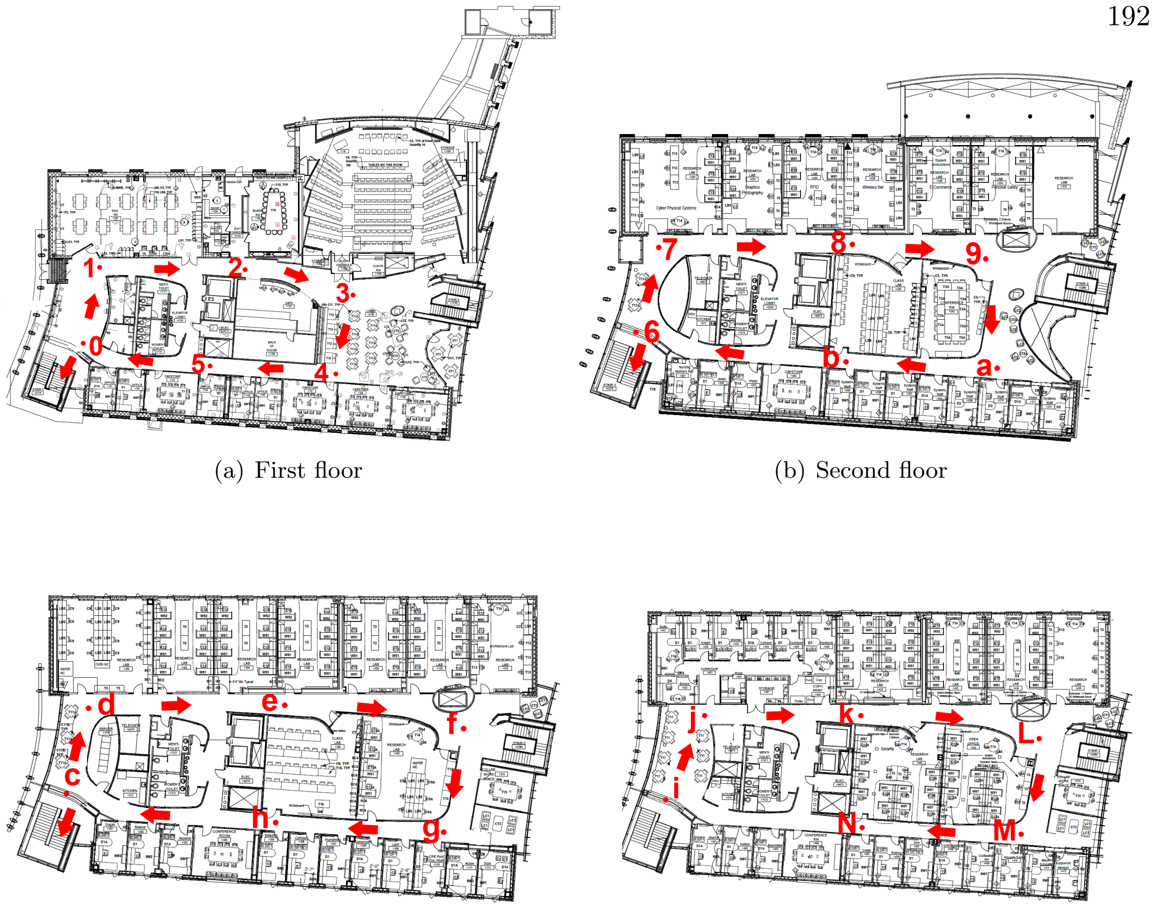


Figure 6.29: Foot-mounted inertial navigation module set up

6.3.3 Evaluation

We evaluate the localization performance of the foot-mounted inertial navigation system in the Department of Computer Science of University of Virginia (The Rice Hall), which has 4 floors. The IMU we use is MinIMU-9 v2 [245], which integrates the gyroscope chip L3GD20 [246], and the accelerometer and magnetometer combination chip LSM303DLHC [247]. The IMU is connected to an Arduino UNO board [252], and the data are output to a laptop through the serial port. The hardware setup is shown in Figure 6.29.

Figure 6.30 shows the pre-defined route in the building. The marks from numbers 0 to 9 and the alphabet "a" to "n" are waypoints, whose 3D coordinates are precisely measured in



(a) First floor

(b) Second floor

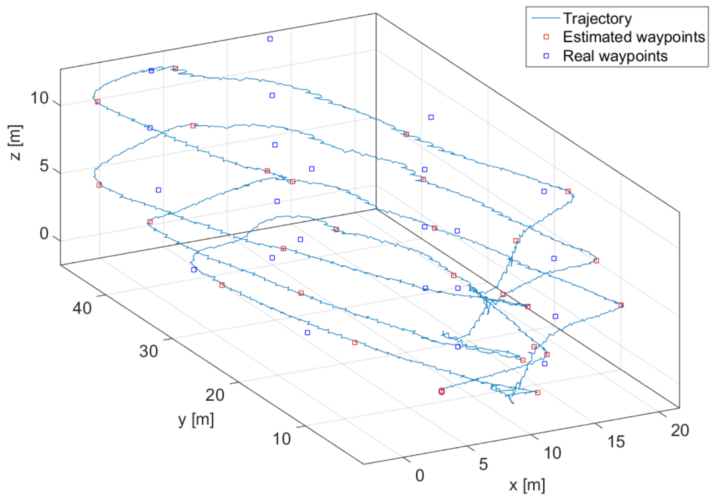
(c) Third floor

(d) Fourth floor

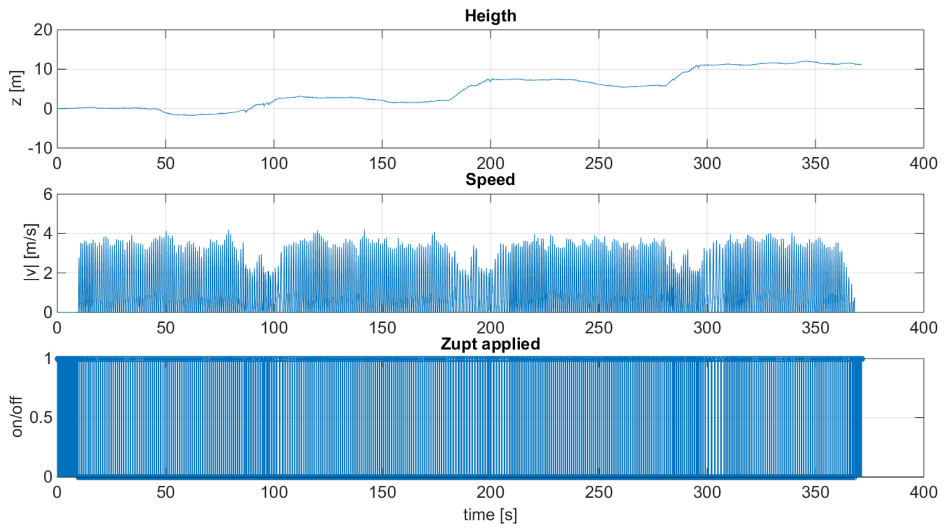
Figure 6.30: The floor plan and the pre-defined route.

advance. The route begins with Waypoint 0 of the first floor, makes a loop, turns back to 0, and then goes upstairs. The second floor starts from 6, turns back to 6 after a loop, and then goes to the third floor. Similarly, the third floor is from c, d, e, f, g, h, and back to c. The fourth floor starts from i, and ends with N. When the pedestrian reaches one mark, he/she presses the corresponding button on the laptop, so it generates a special record inserted into the inertial data. Using this method, we know at which time point the pedestrian reaches which waypoint. Our code is based on the open source project–OpenShoe [243].

Figure 6.31 shows the estimation of the foot-mounted inertial navigation system with aforementioned configuration. Figure 6.31(a) shows the 3D trajectory estimation of the



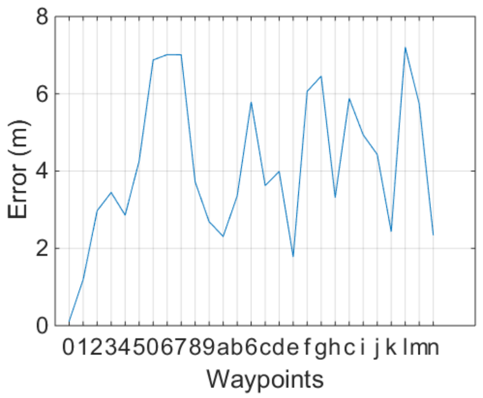
(a) 3D trajectory estimation



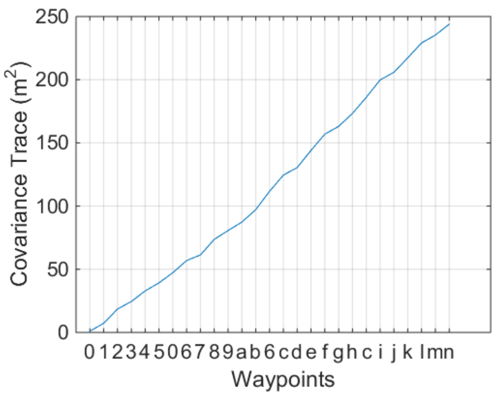
(b) The height, speed and ZUPT event estimation

Figure 6.31: The foot-mounted inertial navigation system estimation

system. We see that as the pedestrian walks from the first floor to the fourth floor, the position error increases. Figure 6.31(b) shows the estimation of the height, the speed and the time when ZUPT is applied. The height changes of different floors are clearly seen from the top plot of Figure 6.31(b). The estimated pedestrian velocity is about 4m/s. The steps are also clearly identified in the “ZUPT applied” figure (the bottom one in Figure 6.31(b)).



(a) The estimation error at waypoints



(b) The trace of the covariance matrix at waypoints

Figure 6.32: The estimation error and covariance of the foot-mounted inertial navigation system

Figure 6.32 shows the estimation error and the trace of the covariance matrix at the waypoints. We see that the error in Figure 6.32(a) increases quickly at beginning, but keeps fluctuation in the remaining time. This is probably because the testing route is circular, the accumulated drift is compensated when the direction reverses. The trace of the covariance matrix in Figure 6.32(b) shows the trend of estimation uncertainty. It is linear with the distance as claimed at the beginning of this section. The estimated traveling distance is around 400m, and the maximum error is 7.205m. Therefore, the error to traveling distance ratio is 1.8%. In the following sections, we will see that when we combine the foot-mounted inertial navigation system, the RSS to distance mapping module, and the EDCL algorithm, the error is further bounded.

6.4 The System Integration

The above sections independently present the RSS to distance estimator (Section 6.2), the foot-mounted inertial navigation system (Section 6.3), and the EDCL filter (Chapter 5). This section presents the integration of these three modules. For proof of concept purpose, we do not embed the module code into each individual hardware. Instead, the RSS data from the radio and the inertial data from the IMU are collected into each agent's laptop,

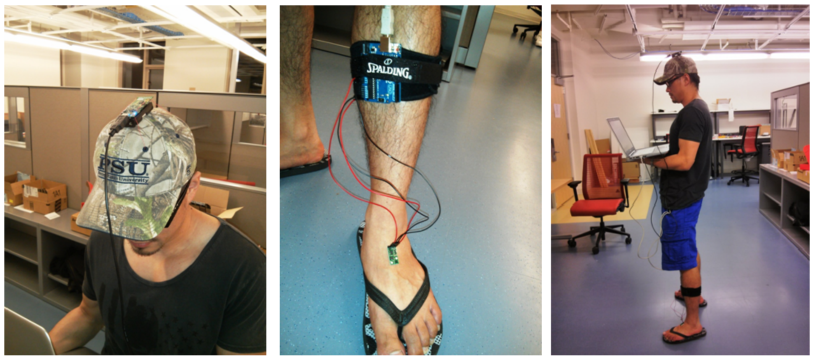
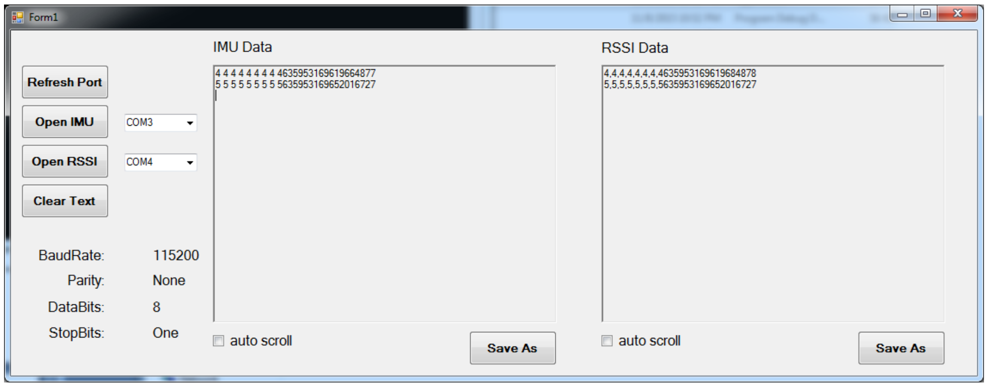


Figure 6.33: The indoor pedestrian localization system setup



COM3

COM4

BaudRate: 115200
Parity: None
DataBits: 8
StopBits: One

```
4 4 4 4 4 4 4 4 4635953169619664877
5 5 5 5 5 5 5 5 5635953169652016727
```

```
4,4,4,4,4,4,4,4635953169619684878
5,5,5,5,5,5,5,5635953169652016727
```

Figure 6.34: The GUI of the data collector

and then post processed in a central place. In future, we plan to make each module work independently so that each module only outputs its processed positional measurements. Then through the wireless communication, these measurements are gathered into a portable device which runs the EDCL module to estimate the final position in real time.

Figure 6.33 shows the system setup for one agent. The left picture is the RSS to distance estimator module, which is implemented using a TelosB mote. The mote is put on the top of the pedestrian's head to avoid the body shadowing effect. All the packets it receives are forwarded to the laptop through a serial port. The middle picture is the foot-mounted inertial navigation module. The minImu9 V2 board is attached to the foot, and connects to an Arduino UNO board on the shank. The Arduino board forwards the data from the IMU to the laptop through the serial port. The right picture shows that a pedestrian holds a laptop, which saves all the data from the TelosB and the IMU. Figure 6.34 shows the GUI of the data collector on laptop. The left panel is to display the data from the IMU, and the right panel displays the RSS data. Each serial port can be turned on/off independently. When a predefined button is pressed on the laptop, a special line is inserted into both panels. The line includes the button name and the local laptop timestamp. This feature is used for recording the waypoints and also for time synchronization. All the data shown on the panels can be saved to a text file locally.

6.4.1 Packet Format and Time Synchronization

There are three modules in one pedestrian's localization system, each of which has its own local clock. Multiple pedestrians' participation makes the time discrepancy issue even more complicated. To correctly relate a measurement to the right time, a time synchronization mechanism is needed. Therefore, a careful design of the data format of each module is required.

For the RSS to distance estimator, based on the data format in Section 6.2.3.3, the sending packet format in the integrated system is designed as follows:

8	1	7	64	64	64
sender_id	phase_mode	slot_id	seq	last_seq	last_send_time

When the receiver receives the packet, it appends the local timestamp and the RSS value, and then forwards it to the laptop. The packet format that the serial port receives is as below:

8	1	7	64	64	64	64	8
sender_id	phase_mode	slot_id	seq	last_seq	last_send_time	receive_time	rssi

The IMU output format is as below:

8	8	8	8	8	8	64	64
gyro_x	gyro_y	gyro_z	acc_x	acc_y	acc_z	local_time	seq

Figure 6.46 shows the time synchronization process. Assuming that Agent 2's IMU local time t_i^2 needs to be synced to Agent 1's IMU local clock t_i^1. t_i^2 is first synced to Agent 2's

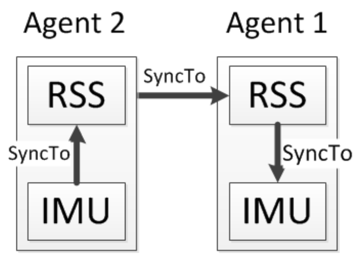


Figure 6.35: Synchronize Agent 2's IMU local time to Agent 1's IMU clock

local RSS module time t_r^2, then to Agent 1's local RSS module time t_r^1, and finally to Agent 1's IMU local time t_i^1.

For time synchronization between two clocks, we need to know some anchor time. The anchor time is defined as a time when two clock's timestamps refer to the same time. The data structure of anchor times could be a mapping $\mathbf{m}^{12} = \{t_1^1 \to t_1^2, t_2^1 \to t_2^2, \cdots, t_n^1 \to t_n^2\}$, where t_i^1 is a list of timestamps from Clock 1, and t_j^2 is a list of timestamps from Clock 2, $i, j = 1, 2, \cdots, n$. Each element of the mapping list represents that the timestamp t_i^1 in Clock 1 and the timestamp t_i^2 in Clock 2 refer to the same time. The time synchronization problem can be modeled as:

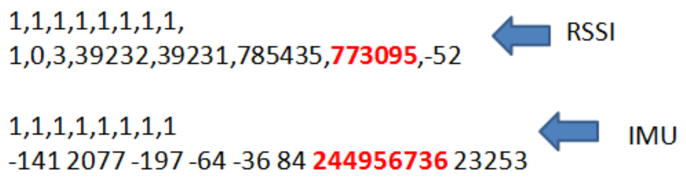
$$\mathbf{t}^2 = f(\mathbf{m}^{12}, \mathbf{t}^1) \tag{6.63}$$

where $\mathbf{t}^2$ is the list of output times in Clock 2, $\mathbf{m}^{12}$ is the anchor time mapping list from Clock 1 to Clock 2, $\mathbf{t}^1$ is the list of input times in Clock 1, and $f(\cdot)$ is the conversion function. $f(\cdot)$ is essentially an interpolation function with considering the time drifts.

The algorithm of $f(\cdot)$ is shown in Algorithm Box 1. The input $\mathbf{m}^{12}$ is a n by 2 matrix, representing the anchor time mapping list from Clock 1 to Clock 2. $\mathbf{t}^1$ is a vector of size m, representing the timestamp list in Clock 1 to be converted to the time list in Clock 2. Line 2 is to sort the time mapping list in the ascend way in Clock 1's time. Lines 2 and 3 are to calculate the time drift per unit time in Clock 1. Note that the matrix index annotation is represented in the Matlab way, e.g. $\mathbf{m}_s^{12}(:, 2)$ represents Column 2 of matrix $\mathbf{m}_s^{12}$. Line 5 to Line 15 is to convert the input Clock 1's time to Clock 2's local time one by one. Line 6 is

Algorithm 1 Time synchronization algorithm

```
 1: procedure TIMESYNCHRONIZE($\mathbf{m}^{12}, \mathbf{t}^1$)
 2:     $\mathbf{m}_s^{12} \leftarrow sort(\mathbf{m}^{12})$;
 3:     $\delta\mathbf{t}^{21} \leftarrow \mathbf{m}_s^{12}(:,2) - \mathbf{m}_s^{12}(:,1)$;
 4:     $t_d \leftarrow \frac{\delta\mathbf{t}^{21}(end) - \delta\mathbf{t}^{21}(1)}{\mathbf{m}_s^{12}(end,1) - \mathbf{m}_s^{12}(1,1)}$;
 5:     for $t \in \mathbf{t}^1$ do
 6:         $I \leftarrow find(\mathbf{m}_s^{12}(:,1) \geq t, 1)$;
 7:         if $I = \emptyset$ then
 8:             $t^2 = \mathbf{m}_s^{12}(end,2) + (t - \mathbf{m}_s^{12}(end,1)) \times (1 + t_d)$;
 9:         else if $I = 1$ then
10:             $t^2 = \mathbf{m}_s^{12}(I,2) - (\mathbf{m}_s^{12}(I,1) - t) \times (1 + t_d)$;
11:         else
12:             $t^2 = \mathbf{m}_s^{12}(I-1,2) + \frac{(\mathbf{m}_s^{12}(I,2) - \mathbf{m}_s^{12}(I-1,2)) \times (t - \mathbf{m}_s^{12}(I-1,1))}{\mathbf{m}_s^{12}(I,1) - \mathbf{m}_s^{12}(I-1,1)}$;
13:         end if
14:         $\mathbf{t}^2 = [\mathbf{t}^2, t^2]$;
15:     end for
16:     return $\mathbf{t}^2$;
17: end procedure
```



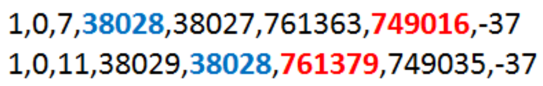
```
1,1,1,1,1,1,1,1,
1,0,3,39232,39231,785435,773095,-52

1,1,1,1,1,1,1,1
-141 2077 -197 -64 -36 84 244956736 23253
```

Figure 6.36: The RSS data and the IMU data after pressing the key of number 1.

to find which slot of the anchor time mapping the time t is in. After that, depending on if time t is bigger than the largest time in $\mathbf{m}_s^{12}(:,2)$ (Lines 7 and 8), smaller than the smallest time in $\mathbf{m}_s^{12}(:,2)$ (Lines 9 and 10), or in some between (Lines 11 and 12), t^2 is calculated differently. The basic idea is that if timestamp t^1 in Clock 1 is between the anchor times t_i^1 and t_j^1 in Clock 1, it should be in the same proportional position between the anchor times t_i^2 and t_j^2 in Clock 2 with the time drift correction. At the end of each loop, the resultant t^2 is appended to the return list $\mathbf{t}^2$ (Line 14). Finally $\mathbf{t}^2$ is returned (Line 16).

For time synchronization between Agent 2's IMU module and RSS module, the anchor time mapping list is built by the key pressed by the pedestrian at each waypoint. Figure 6.36 shows the IMU data and the RSS data after pressing the key of number 1. We can assume that the data of the IMU and the RSS right after the special line "1,1,1,1,1,1,1,1" refer to the



```
1,0,7,38028,38027,761363,749016,-37
1,0,11,38029,38028,761379,749035,-37
```

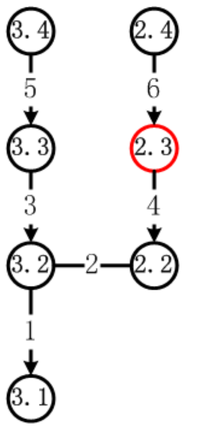
Figure 6.37: Two lines of successive RSS records from sender Id 1.

same time, which means for this example the anchor time mapping is 244956736 $\rightarrow$ 773095 (Agent 2's IMU clock to Agent 2's RSS clock). Since in our experiment, the data rate of IMU is 100Hz, and the packet sending rate is 50Hz, the maximum error for each element of this anchor time mapping is $\max(r1, r2) = 20ms$. If the pedestrian passes n waypoints, the size of the anchor time mapping list is $n \times 2$. Given the mapping list, we can freely convert IMU's time to RSS module's time using Algorithm 1, or vice verse.

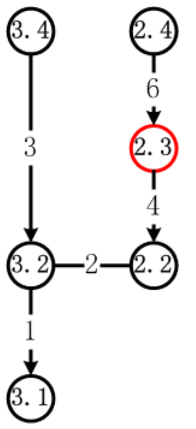
Figure 6.37 shows Agent 2's two successive RSS records from sender Id 1. The first record shows that the message sequence number is 38028, and the receiving timestamp (receiving SFD) is 749016 in Agent 2's local clock. The second record shows that the last message sequence number is 38028, and the last sending timestamp (sending SFD) is 761379 in Agent 1's local lock. Hence, we get one element of the anchor time mapping list from Agent 1's RSS module clock to Agent 2's RSS module clock: 761379 $\rightarrow$ 749016. By using this method, an anchor time mapping list between Agent 1's RSS module and Agent 2's RSS module can be built. After that we can freely convert Agent 1's RSS time to Agent 2's RSS time using Algorithm 1, or vice verse. As described in Section 6.2.3.1, the error for this time synchronization mechanism is less than $1\mu s$.

6.4.2 Measurement Generation

After collecting the raw data from the IMU and the RSS modules, position related measurements need to be generated for the EDCL algorithm. For the RSS raw data, all senders' messages are mixed together. By separating each sender's message, the RSS time series of each sender and the anchor time mapping list between each sender and the receiver can be obtained. After that, by using the corrected truncated maximum likelihood distance estimator in Section 6.2.2.3, the range measurements between the receiver and each sender



(a) Before compounding



(b) After compounding

Figure 6.38: Compounding the displacement measurements

can be generated. For the foot-mounted inertial navigation module, by inputting the raw gyroscope and accelerometer data into the ZUPT algorithm in Section 6.3, we get the position estimate and its covariance over time. Then the displacement measurements can be generated by calculating the difference between two successive position estimates.

For RSS to distance measurement, because the corrected truncated maximum likelihood distance estimator is used, many low RSS values are dropped, and thus the number of generated measurements are not that high. However, the number of the inertial navigation generated measurements is high, since the sampling frequency is 100Hz and one data sample corresponds to one measurement. If the measurements are generated in this naive way, there will be $k + 100mt$ measurements, where k is the total RSS generated inter-agent measurements which is in the order of $O(m^2t)$, m is the number of the agents, and t is the system running time.

Since most of the measurements are from the inertial navigation module, we can reduce the number by compounding two successive displacement measurements. Given two successive displacement measurements $\mathbf{z}_{ij} = (i, j, \mathbf{x}_{ij}, \mathbf{C}_{ij})$ and $\mathbf{z}_{jk} = (j, k, \mathbf{x}_{jk}, \mathbf{C}_{jk})$, the compounded one is $\mathbf{z}_{ik} = (i, k, \mathbf{x}_{ij} + \mathbf{x}_{jk}, \mathbf{C}_{ij} + \mathbf{C}_{jk})$, where $\mathbf{z}_{ij}$ represents the displacement measurement data structure from Position i to Position j, i is the node Id in a measurement graph, $\mathbf{x}_{ij}$ is the real displacement measurement from Node i to Node j, and $\mathbf{C}_{ij}$ is the corresponding covariance matrix. Note that this operation should not compound away a node which involves in an inter-agent measurement. Additionally, the waypoint node should not be compounded away since it is needed to compare with the ground truth.

Figure 6.38 shows the example of compounding the displacement measurements. Figure 6.38(a) shows the measurements before the compounding. There are totally two agents (2 and 3), and 6 measurements. Measurement 2 is the RSS range measurement, and all the remaining are displacement measurements. The red circled node 2.3 is a waypoint. Following the above rule, only Measurements 3 and 5 can be compounded. Figure 6.38(b) is the result after compounding.

By using the compounding operation, all displacement measurements without involving a node having an inter-agent measurement can be compounded. Therefore, the total number of measurements are determined by the number of inter-agent measurements and the number of waypoints. Each inter-agent measurement introduces three measurements (2 compounded displacement measurements), each waypoint introduces an extra displacement measurement, and there is one ending displacement measurement for each agent. Assume there are m agents, k inter-agent measurements and w recorded waypoints. The total number of the measurements is $n = 3k + w + m$.

When generating the displacement measurements, one thing needs to be done carefully. A displacement measurement is generated by $\hat{\mathbf{x}}_{ij} = \hat{\mathbf{x}}_i - \hat{\mathbf{x}}_j$, and the corresponding covariance matrix is $C_{ij} = C_j - C_i$, where $i < j$. However, by introducing the Kalman filter, the covariance matrix may not monotonically increase, which results in C_{ij} may not be a positive definite matrix. Figure 6.39 shows the traces of the covariance matrix output from ZUPT in some time period. There are peaks and valleys in the figure. One peak and its successive valley represent a step, where the pedestrian begins with a stride and finally ends up with a stance phase which is corrected by ZUPT algorithm. Assume there are three inter-agent measurements at time points A, B and C, respectively. When the displacement measurement is generated for time duration between A and B, a non-positive definite matrix

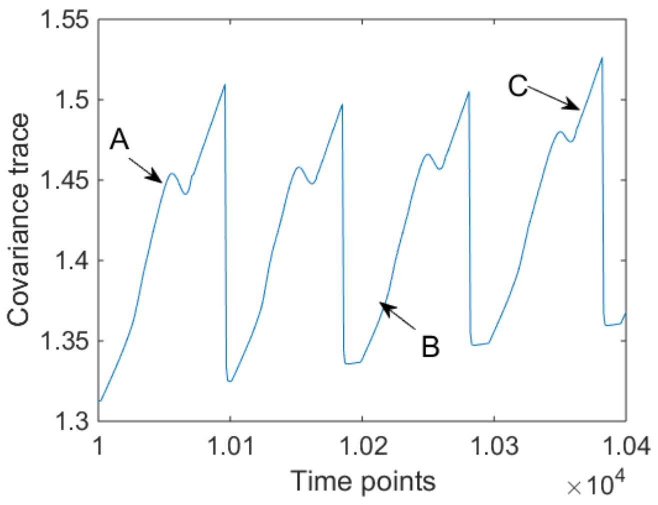


Figure 6.39: The trace of the covariance matrix of ZUPT estimation may decrease in some time period

C_{AB} will be obtained, which would crash the whole estimation process. Therefore, in our implementation, we choose to drop the inter-agent measurement at Time B, and use the next available inter-agent measurement which can generate a positive definite covariance matrix. For this example, the inter-agent measurement at Time C is taken, and thus a positive definite covariance matrix C_{AC} is obtained.

6.5 Evaluation

To evaluate the integrated system, four people participate in the experiment. Each person is equipped with a TelosB mote on the top of the head, a foot-mounted inertial navigation module on foot, and a laptop holden on hands. The experiment is conducted inside a four floor building (the department of computer science of University of Virginia–Rice Hall). Waypoints are preset, and their location are precisely measured (shown in Figure 6.30).

Three different routes are pre-defined. The first route is that four people walk together. As shown in Figure 6.30), they first start from Waypoint 0 on the first floor, walk a loop back to Waypoint 0, and then go upstairs. The routes on Floors 2, 3 and 4 are similar. They finally stop at Waypoint N on the fourth floor. We name this route ”together”. The precise route defined by the passing waypoints is as Table 6.3 shows.

Table 6.3: Route “together”

P1,P2,P3,P4	0, 1, 2, 3, 4, 5, 0, 6, 7, 8, 9, a, b, 6, c, d, e, f, g, h, c, i, j, k, L, M, N

The second route is that 2 people start from Waypoint 0 on the first floor, and the other two start from Waypoint N on the fourth floor. They walk in opposite directions, and meet at Waypoint d on the third floor. If one group reach Waypoint d first, they wait until the other group meets them. After meeting, Group 1 continues to walk to Waypoint N on the fourth floor, while Group 2 continues to go oppositely to Waypoint 0 on the first floor. Therefore, the route of Group 1 is the same as the above, and the route of Group 2 reverses. We name this route ”encountering”. The precise route is shown in Table 6.4

Table 6.4: Route “encountering”

P1,P2	0, 1, 2, 3, 4, 5, 0, 6, 7, 8, 9, a, b, 6, c, d	e, f, g, h, c, i, j, k, L, M, N
P3,P4	N, M, L, k, j, i, c, h, g, f, e, d	c, 6, b, a, 9, 8, 7, 6, 0, 5, 4, 3, 2, 1, 0

The third route is more complicated. Group 1 with two people (P1 and P2) starts from Waypoint 0 on the first floor, and Group 2 (P3 and P4) with the other two starts from Waypoint N on the fourth floor. Group 1 first follows the same route as the above until reaching Waypoint 7 on the second floor. Group 2 first follows the following route: L, k, j, i, N, M, L, f, e, d, c, h, g, f, 9, a, b. At this time point, Group 1 should wait for Group 2 at Waypoint 7, and Group 2 is now at Waypoint b. Then two groups keep still, and P4 leaves Group 2 and moves forward to join Group 1. After that, P4 and Group 1 walk together to the next waypoint 8, and at the same time Group 2 (now only P3) moves forward to the next waypoint 6. After that, two Groups keep still, P4 leaves Group 1 and re-joins Group 2. After joining, two groups keep moving to their next waypoints, respectively, and P4 starts commuting between two groups again. The walking pattern for this route is that every time after two groups moving forward for one waypoint, P4 leaves the current group and joins the other group. Then two groups move forward to the next waypoint and P4 commutes again. When P4 is changing group, the two groups keep still. The two groups always keep the distance of two waypoints. The final destination for the 4 pedestrians is at Waypoint N on the fourth floor. We name this route "commuting". The precise route is defined in Table 6.5.

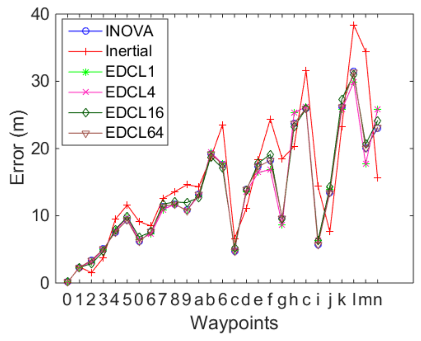
Table 6.5: Route “commuting”

P1,P2	0, 1, 2, 3, 4, 5, 0, 6, 7													7	7	8	8	8	9	9
P3	L, k, j, i, N, M, L, f, e, d, c, h, g, f, 9, a, b													b	b	6	6	6	7	7
P4	L, k, j, i, N, M, L, f, e, d, c, h, g, f, 9, a, b													6	7	8	7	6	7	8
P1,P2	9	a	a	a	b	b	b	6	6	6	c	c	c	d	d	d	e	e	f	f
P3	7	8	8	8	9	9	9	a	a	a	b	b	b	6	6	6	c	c	c	d	d
P4	9	a	9	8	9	a	b	6	b	a	b	6	c	d	c	6	c	d	e	f	e
P1,P2	f	g	g	g	h	h	h	c	c	c	i	i	i	j	j	j	k	k	k	L	L
P3	d	e	e	e	f	f	f	g	g	g	h	h	h	c	c	c	i	i	i	j	j
P4	d	e	f	g	h	g	f	g	h	c	i	c	h	c	i	j	k	j	i	j	k
P1,P2	L	M	M	M	N																
P3	j	k	k	k	L	M,N															
P4	L	M	L	k	L	M,N															

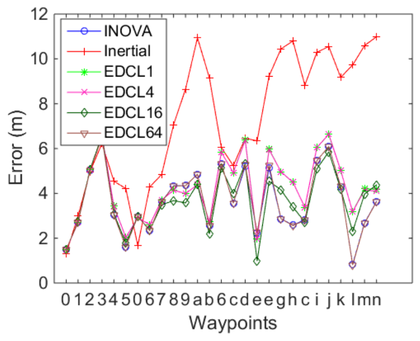
During each route, when a pedestrian reaches a waypoint, he presses the corresponding key, which inserts a special data record in both inertial data and RSS data. At the end of each route, all pedestrians save the data in the laptops. All these data are then sent to a central place for post processing.

6.5.1 Localization Accuracy and Uncertainty

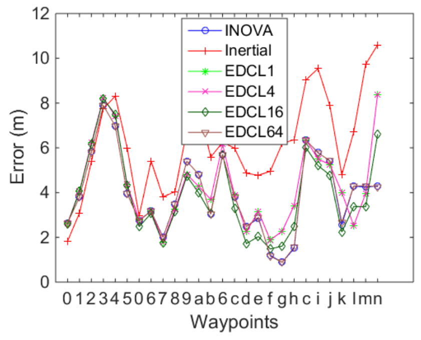
This section compares the localization accuracy and uncertainty for the inertial-only algorithm, the centralized INOVA algorithm and the distributed EDCL algorithm. The inertial-only algorithm is only using the foot-mounted inertial navigation system (ZUPT) without fusing any other measurements. The centralized INOVA algorithm is based on the algorithm described in Section 3.2.3, which assumes that all the inertial and RSS measurements of the four pedestrians are collected at a central place for processing. The EDCL algorithm is a fully decentralized algorithm, which is described in Chapter 5. The marginalization factor Q in EDCL controls the trade-off between the localization accuracy and the resource usage, so it is interesting to evaluate EDCL with different Q settings.



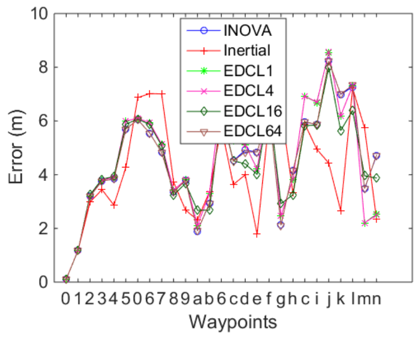
(a) Pedestrian 1



(b) Pedestrian 2



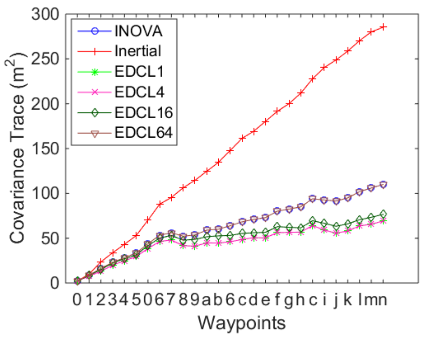
(c) Pedestrian 3



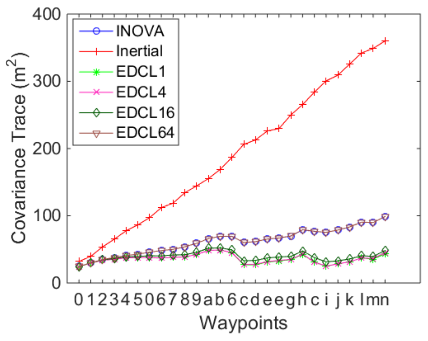
(d) Pedestrian 4

Figure 6.40: The localization norm error comparison at waypoints of Route "together" for the inertial-only, INOVA, EDCL ($Q = 1, 4, 16, 64$) algorithms.

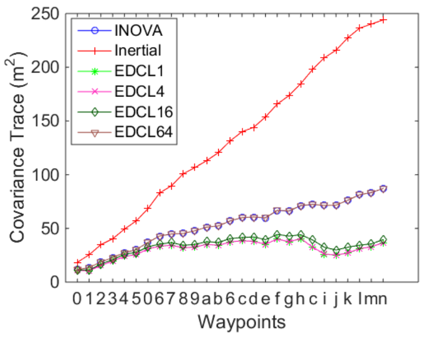
Figure 6.40 shows the localization norm error of four pedestrians at waypoints of route "together". The norm error is defined as $e = \sqrt{(\hat{x}_1 - x_1)^2 + (\hat{x}_2 - x_2)^2 + (\hat{x}_3 - x_3)^2}$, where e is the norm error, $\hat{x}_1$, $\hat{x}_2$ and $\hat{x}_3$ are the estimated position in the three axes, and x_1, x_2 and x_3 are the real position in the three axes. From the figure, we see that the inertial-only algorithm performs worst, and the INOVA algorithm performs best. For the EDCL algorithm, the localization accuracy is different for different Q values. The bigger the Q is, the more accurate the estimation is. Additionally, when $Q = 64$, it has the same localization accuracy as the centralized INOVA algorithm. This means when $Q = 64$, the estimation is optimal. For the inertial-only algorithm, the norm error for Pedestrian 1 is high, which fluctuates around 30m at the end part of the route. All the other Pedestrians have the norm error end at the end of 10m. The high error for Pedestrian 1 may be caused by not tieing



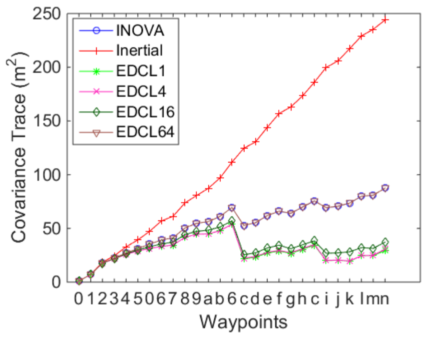
(a) Pedestrian 1



(b) Pedestrian 2



(c) Pedestrian 3

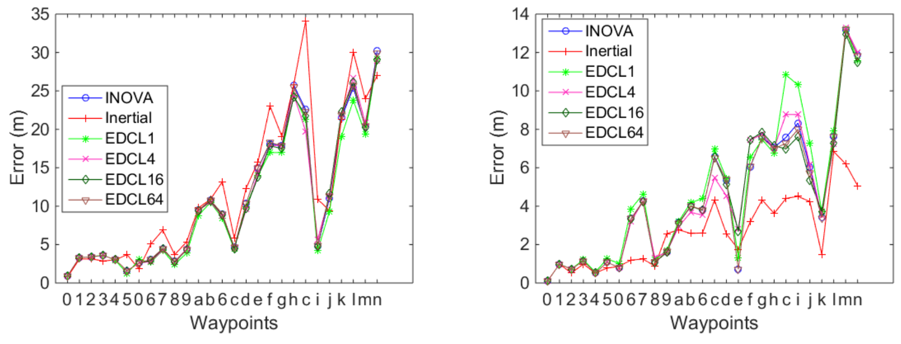


(d) Pedestrian 4

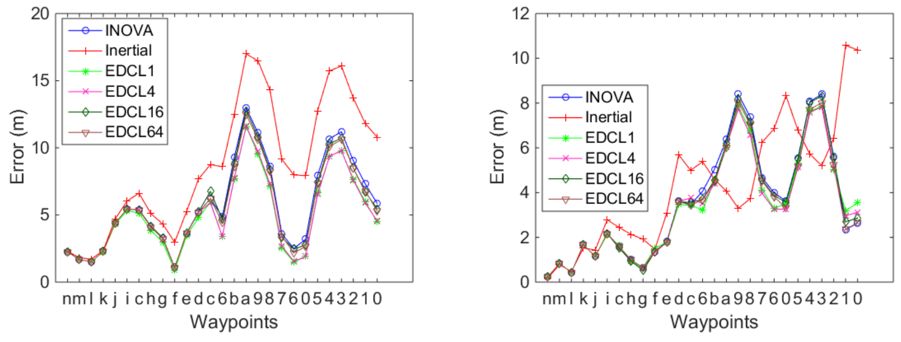
Figure 6.41: The localization covariance matrix traces comparison at waypoints of Route "together" for the inertial-only, INOVA, EDCL ($Q = 1, 4, 16, 64$) algorithms.

the sensor firmly onto the shoes. For INOVA, except the ending norm error for Pedestrian 1 is a between 20m and 30m, all the other Pedestrians have the errors around 6m. For the EDCL algorithm, although their performances differ, they are close to INOVA. By summing up all the norm errors for all Pedestrians and comparing this total error with the one the inertial-only algorithm, we get the percentage of the performance gain for each algorithm. EDCL1 reduces the norm error by 19.7%, EDCL4 reduces by 19.72%, EDCL16 reduces by 22%, and EDCL64 and INOVA both reduce by 22.23%.

Figure 6.41 shows the estimates covariance matrix traces for each Pedestrian over time. The trace of the estimation covariance matrix is an important metric for an estimator's performance, which usually shows the same trend as the practical estimation errors. In addition, by comparing an estimator's covariance matrix with an optimal estimator's, we



(a) Pedestrian 1 (b) Pedestrian 2



(c) Pedestrian 3 (d) Pedestrian 4

Figure 6.42: The localization norm error comparison at waypoints of Route “encountering” for the inertial-only, INOVA, EDCL ($Q = 1, 4, 16, 64$) algorithms.

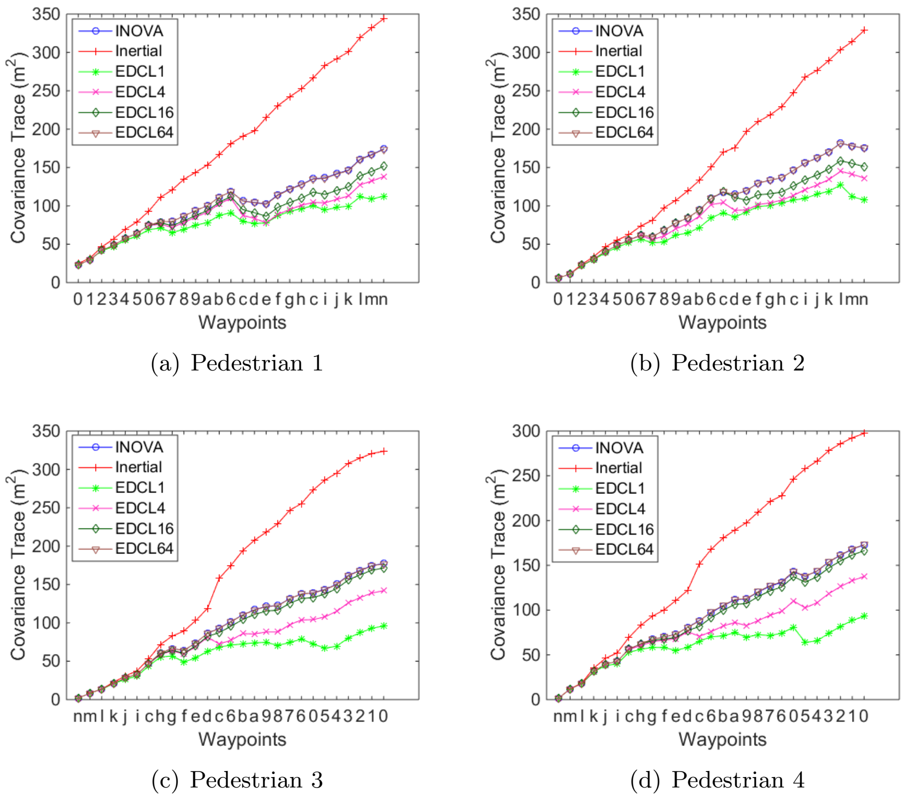
know if the estimator suffers from an over-confidence problem, and how much it has.

From the figure, we see that the trace of the covariance matrix of the inertial-only algorithm increases linearly, while the other estimators have sub-linear trends. Additionally, INOVA and EDCL64 have overlapped curves, which means EDCL64 is optimal in this route. For the other values of Q, their covariance matrix traces are lower than INOVA’s, which means they are non-optimal and suffer from the over confidence problem. The smaller the Q is, the lower the covariance matrix traces it has, which indicates the more approximation it has.

Figure 6.42 shows the norm error for Route “encountering”. The similar observation is made as the previous route. But this time, the overall error is bigger than the route “together”. It may be because two groups are separated and only meet once, so the inter-agent measurements are fewer. For Agents 2 and 4, the collaborative algorithm performs close or even worse than the inertial-only algorithm. The reason is unclear yet. It may be because some bad estimate is associated with a small covariance matrix. By using the same mean, all algorithms are compared with the inertial-only algorithm. EDCL1 reduces the error by 19.7%, EDCL4 reduces by 18.92%, EDCL16 reduces by 16.32%, EDCL 64 reduces by 16.66%, and INOVA reduces by 14.83%. This time EDCL1 seems to be better than others. It is possible that the inertial estimates usually only deviate to one side (see [253]), when two groups meet from the opposite direction, the errors are canceled out. For EDCL1, it is much easier to fully cancel these error out since it has less constraints than EDCL64 does. For the covariance matrix trace in Figure 6.43, it has the similar results as Route "together". Inertial-only algorithm grows linearly, and the collaborative algorithms are more bounded. Similarly, EDCL64 is close to INOVA, and EDCL1 suffers from the over confidence problem in a most severe way.

Figure 6.44 shows the waypoint norm error for Route "commuting". This route is more complex than the previous two routes. The purpose of this route is to show the benefits of Q with a high value. In Route "together", since all pedestrians walk together, their position estimation error is similar. Hence, EDCL1 which is essentially an averaging method should not perform very differently from EDCL64. In Route "encountering", two groups only meet once, and they are from the opposite directions. Therefore, EDCL1 maybe even better to cancel out the error in the opposite directions. However, in Route "commuting", Pedestrian 3 keeps commuting between two groups, which results in the position information is shared between groups in a complex way. In this process, if the correlations between pedestrians are not handled well, the error is not be reduced much.

From Figure 6.44, the first observation is that all collaborative algorithms are much better than the inertial-only algorithm (except the ending part of Pedestrian 2). The second observation is that the gap between EDCL1 and EDCL 64 are bigger than the previous routes. However, in this route, since two groups are apart form each other with only two waypoints, they may still have some information sharing. If the two group's information



(a) Pedestrian 1

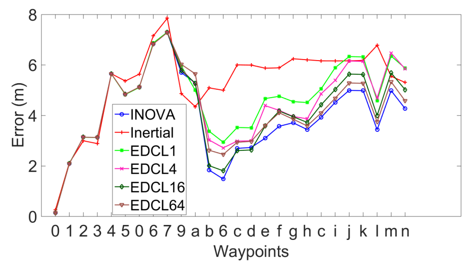
(b) Pedestrian 2

(c) Pedestrian 3

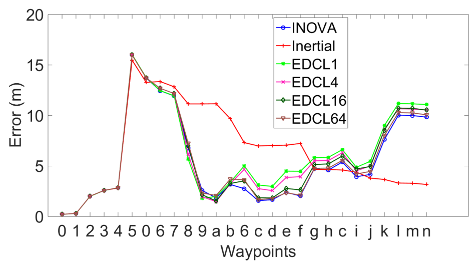
(d) Pedestrian 4

Figure 6.43: The localization covariance matrix traces comparison at waypoints of Route "encountering" for the inertial-only, INOVA, EDCL ($Q = 1, 4, 16, 64$) algorithms.

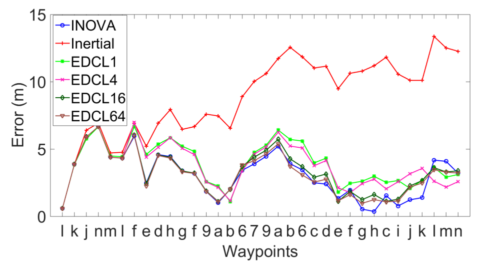
sharing only relies on Pedestrian 4, the gap between EDCL1 and EDCL64 may be even more significant. The third observation is that Pedestrians 3 and 4 reduce the inertial errors more than Pedestrians 1 and 2 do. For Pedestrian 4, it is easy to understand. Since Pedestrian 4 is commuting, he walks the longest way. On the other side, Pedestrian 4 fuses the most position information from two groups, the collaborative effect is the most significant for him. For Pedestrian 3, the reason for the good performance is that he walks with Pedestrian 4 at the beginning, so he accumulated the complete information relation graph with Pedestrian 4. After that, when he fuses the information relations from Pedestrian 4, he has less graph structure conflicts than Pedestrians 1 and 2. For this route, comparing with inertial-only algorithm, EDCL1 reduces the error by 39.45%, EDCL4 reduces by 41.59%, EDCL16 reduces by 47.81%, EDCL64 reduces by 49.44%, and INOVA reduces by 50.41%. Figure



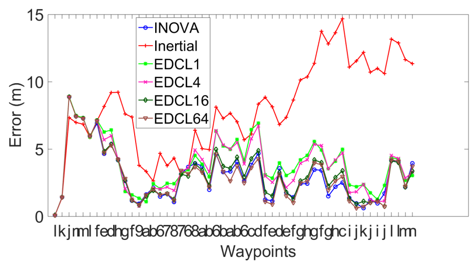
(a) Pedestrian 1



(b) Pedestrian 2

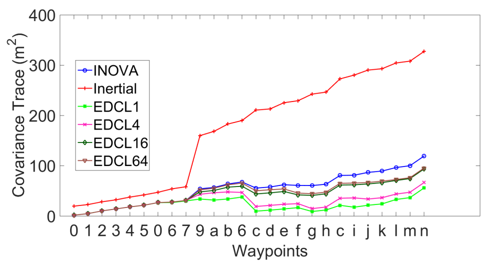


(c) Pedestrian 3

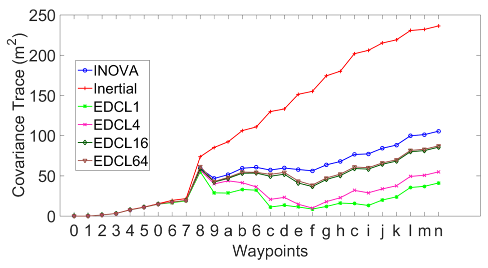


(d) Pedestrian 4

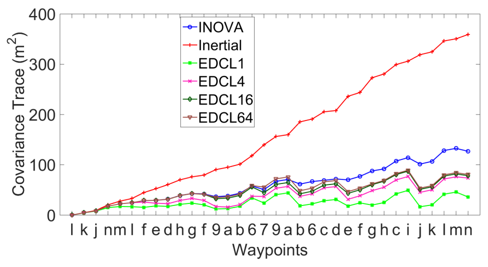
Figure 6.44: The localization norm error comparison at waypoints of Route "commuting" for the inertial-only, INOVA, EDCL ($Q = 1, 4, 16, 64$) algorithms.



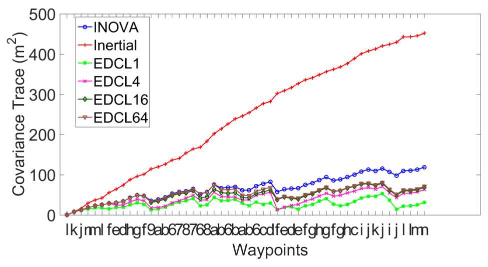
(a) Agent 1



(b) Agent 2



(c) Agent 3



(d) Agent 4

Figure 6.45: The localization covariance matrix traces comparison at waypoints of Route "commuting" for the inertial-only, INOVA, EDCL ($Q = 1, 4, 16, 64$) algorithms.

6.45 shows the covariance matrix trace for each waypoint. The similar conclusion is made

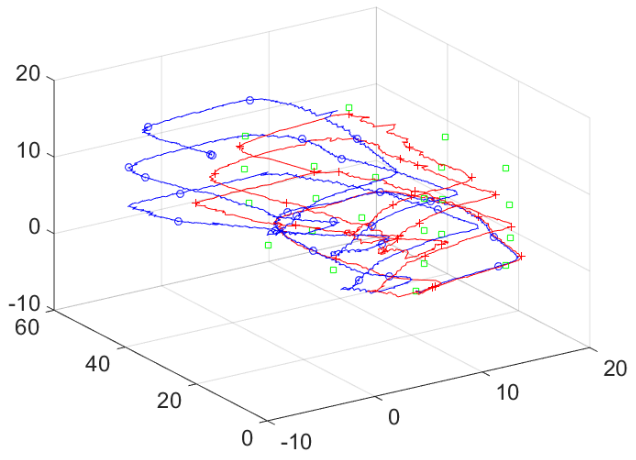


Figure 6.46: Pedestrian 2's trajectory estimation in Route "together". The blue curve is the inertial-only estimation. The green curve is the estimation of EDCL64. The green square marker is the ground truth of the waypoints.

as for the above two routes.

Figure 6.46 is one example of how collaborative localization improves the pedestrian's trajectory estimation. The figure is Pedestrian 2's trajectory in Route "together". The blue curve is the estimation of the inertial-only algorithm. The green curve is the estimation of EDCL64. The green square marker is the ground truth of the waypoints. We see that the estimation of the inertial-only algorithm deviates the ground truth more than the EDCL64 .

6.5.2 EDCL Resource Usage Evaluation

As described in Chapter 5, EDCL uses the marginalization factor Q to balance the trade-off between the localization accuracy and resource usage. The bigger the Q is, the more accurate (more consistent) the result is, and on the other hand, the more resources it uses. This chapter studies the memory usage, the communication cost and the information relation graph structural conflicts of EDCL with different Q setting for the above experiment.

Figure 6.47 shows the maximum memory usage for three pedestrians in Route "together". Actually, at most time the information relation graph of the pedestrian is full loaded.

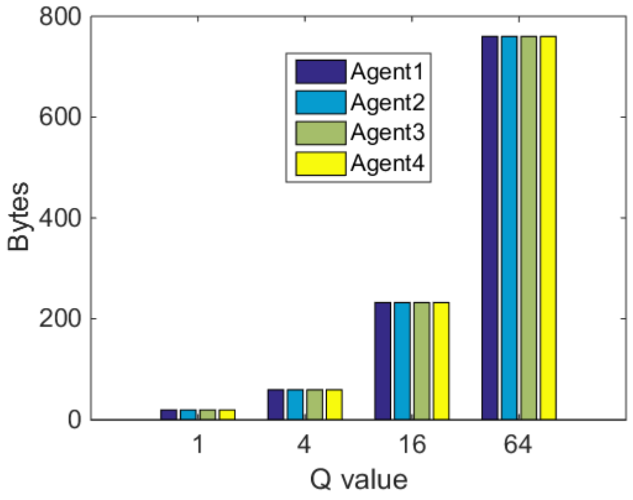
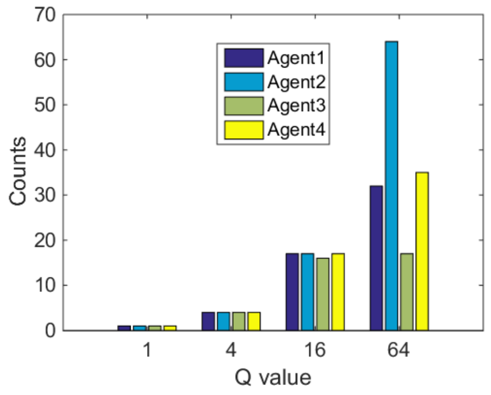


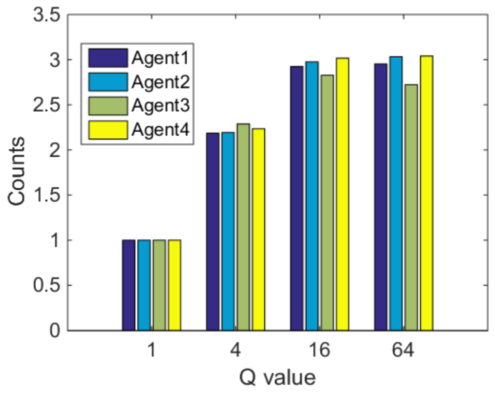
Figure 6.47: Maximum memory usage in Route “together”

Therefore, this maximum memory usage number is almost the same as the average memory usage. The figure shows the with a bigger Q, more memory is needed. When $Q = 1$, the memory usage is only 19 byte, while when $Q = 64$, the memory usage is 760 bytes, which is still quite small. Chapter 5 explains that each agent has the following data in its memory: the agent Id, the agent current position estimate, the corresponding estimation covariance matrix, and an information relation graph which consists of a number of information relations and the belief of the base nodes. For $Q = 64$ and the number of the agents is 4 (so there should be 4 base nodes), we only allows $m = 60$ information relations preserved in the memory. An information relation is a four-tuple $(i, j, \mathbf{y}, \mathbf{Y})$, where i and j are the node Id for source and destination, and $\mathbf{y}$ and $\mathbf{Y}$ are the information vector and information matrix, respectively. Therefore, one information relation is totally $s_i = 1 + 1 + 3 + 6 = 11$ bytes. For the belief of the base nodes, it has a position vector and the corresponding covariance matrix. Therefore, the size is $s_b = 3m + 3m(3m + 1)/2 = 3m(3m + 3)/2$ bytes. The total memory usage for 4 agents are $s = 1 + 3 + 6 + 60S_i + s_b = 1 + 3 + 6 + 660 + 90 = 760$ bytes, which matches the number in the figure. The memory usage is the same in Routes “encountering” and “commuting”.

Figure 6.48 shows the maximum/average number of information relations sent for each



(a) Maximum number of information relations sending in Route "together"

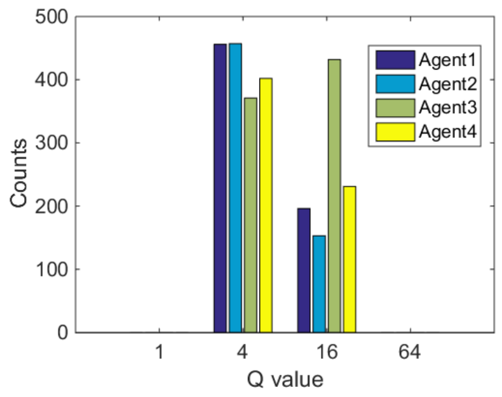


(b) Average number of information relations sending in Route "together"

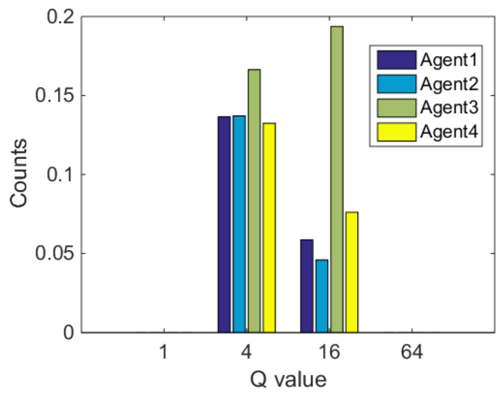
Figure 6.48: The number of the information relations sending in Route "together"

agent in Route "together". In Figure 6.48(a), we see that as Q increases, the maximum number of information relations sent also increases. This metric represents the burst communication bandwidth consumption for EDCL with different Q's. When $Q = 1$, the maximum number sent is only 1 since there is only one information relation is allowed to store in memory. When $Q = 64$, the maximum number raises to 64 in Agent 2. This burst is caused by the initial time one agent not having all of the other one's graph information. Figure 6.48(b) shows the average number of information relations sent. We still see the increment trend with Q, but from $Q = 4$ to $Q = 64$, they do not differ much. This is because all pedestrians are walking together, they constantly exchanges information, and thus the number of information relations need to be sent is low.

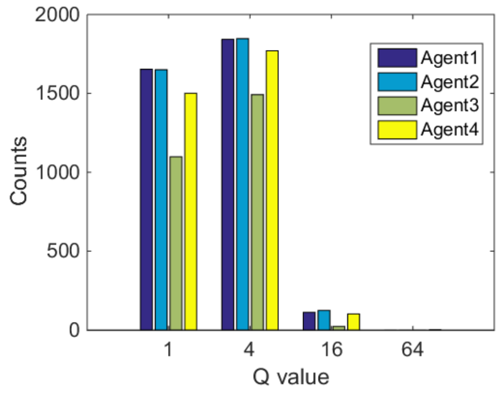
Figure 6.49 shows the structural conflicts of four pedestrians in Route "together". In Section 5.4.2, we define the explicit structural conflict and the implicit structural conflict. Recall that the explicit structural conflict means that when Agent A wants to merge Agent B's information chain about C, the base of Agent B's Chain C is older than that of Agent A's. For the implicit structural conflict, it means Agent B has a fully new Chain C which is not existing in Agent A. But actually A has the Chain C before, but marginalizes it away because of the Q restriction. Both of these two structural conflicts causes the non-consistent



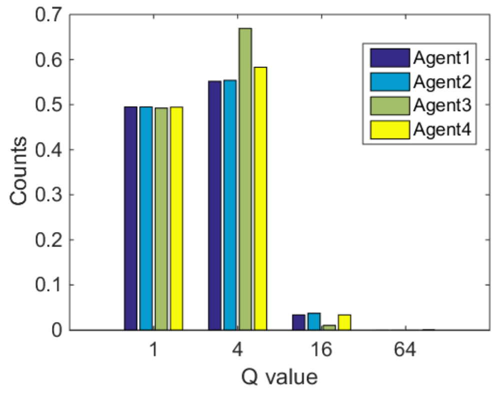
(a) Explicit conflict counts in Route "together"



(b) Average explicit conflict counts in Route "together"



(c) Missing relation chain conflict counts in Route "together"

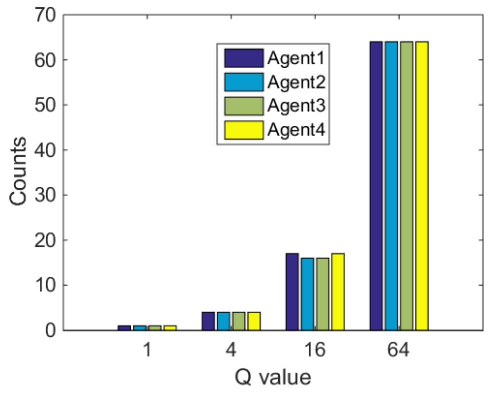


(d) Average Missing relation chain conflict counts in Route "together"

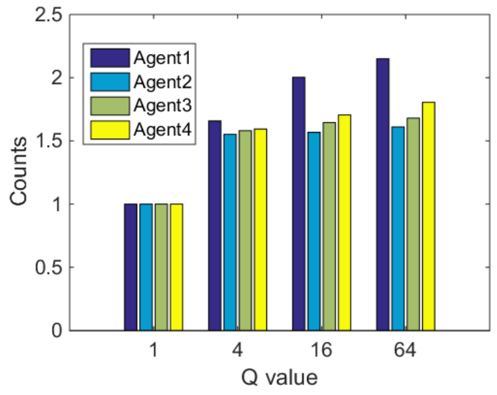
Figure 6.49: EDCL conflict statistics in Route "together"

issue for EDCL. The more it happens, the more severe approximation the estimator suffers from.

For explicit conflicts in Figures 6.49(a) and 6.49(b), when $Q = 1$, the number is 0. This is when $Q = 1$, each agent only keeps the base node of itself, and thus it is impossible for the agent to have conflict on some other agent's information relation chain. For $Q = 4, 16, 64$, we see the bigger the Q is, the less the explicit conflict it has. There is no explicit conflict when $Q = 64$. For implicit conflicts in Figures 6.49(c) and 6.49(d), the same trend holds. The $Q = 1$ setting now has many implicit conflicts, $Q = 16$ has very few, and $Q = 64$ does not have any. That is why in Section 6.5.1, EDCL64 performs the same well as INOVA,



(a) Maximum number of information relations sending in Route "encountering"



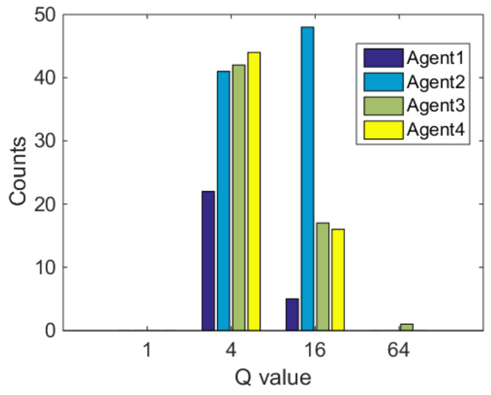
(b) Average number of information relations sending in Route "encountering"

Figure 6.50: The number of the information relations sending in Route "encountering"

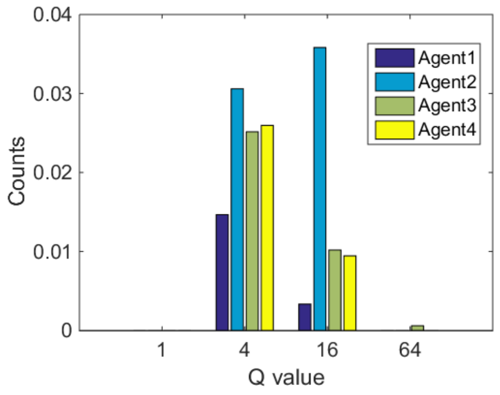
because it has no conflict, and thus it is optimal. From the average conflict figures, we see that for $Q = 1$ and $Q = 4$, they have conflicts for almost every two communications, which impacts their localization accuracy.

Figure 6.50 shows the maximum and average number of information relations sending in Route "encountering". The maximum number is the same as in Route "together", which reaches the maximum number of the information relations allowed in memory. For the average sending, it is a little less than Route "together". This is because that four people are separated into two groups, at most of the time, only two people in the same group are exchanging the information.

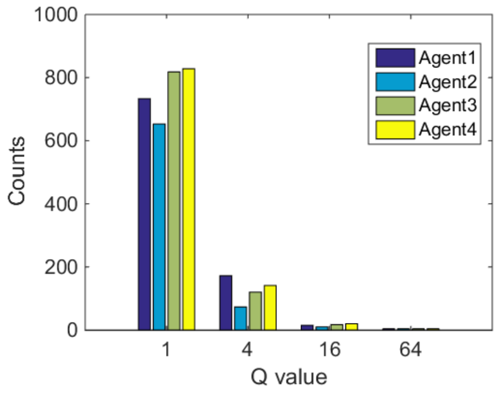
Figure 6.51 shows the information relation graph structural conflicts for Route "encountering". The conflicts reduce as Q increases. Similar, the overall conflict number is lower than that in Route "together" because of less information exchanging. We note that for the explicit conflicts, the value of Agent 2 is much higher than others for $Q = 16$. But its implicit conflicts are the lowest. This is because when exchanging the information, Agent 2 still holds all the other agent's information relations in its memory, and thus most of the time, it has the explicit conflict but not the implicit conflict. In contract, the other agents has lower explicit conflicts because they have more implicit conflicts to obtain the



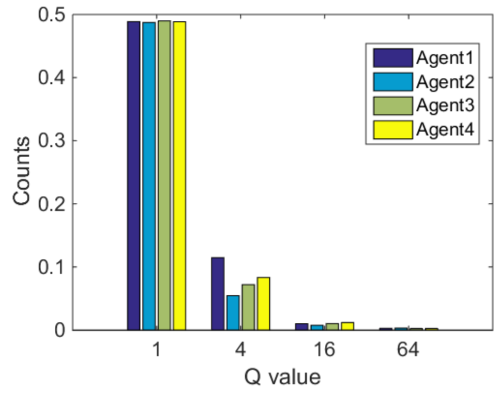
(a) Explicit conflict counts in Route "encountering"



(b) Average explicit conflict counts in Route "encountering"



(c) Missing relation chain conflict counts in Route "encountering"



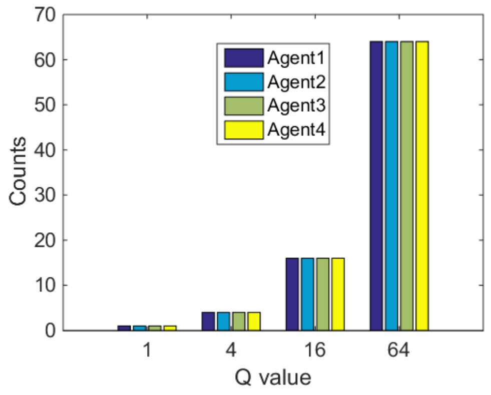
(d) Average Missing relation chain conflict counts in Route "encountering"

Figure 6.51: EDCL conflict statistics in Route "encountering"

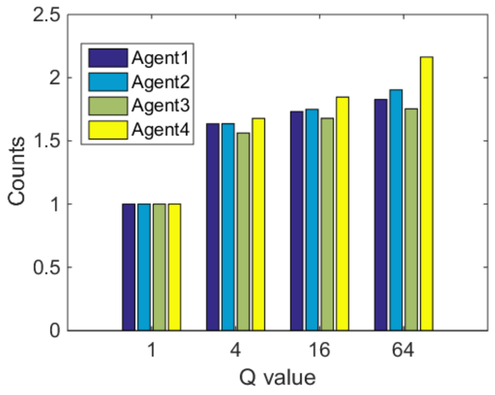
new information.

Figure 6.52 shows the maximum and average number of information relations sending in Route "commuting". The average number is between the numbers of Routes "together" and "encountering". This is because the walking pattern makes the number of the communications less than Route "together" but more than "encountering".

Figure 6.53 shows the conflict statistics in Route "commuting". For the low or medium Q setting, the explicit conflicts are comparable with Route "encountering", but less than Route "together". For implicit conflicts, they are more than Route "encountering" and less than "together". But for $Q = 64$, both the explicit conflicts and the implicit conflicts are

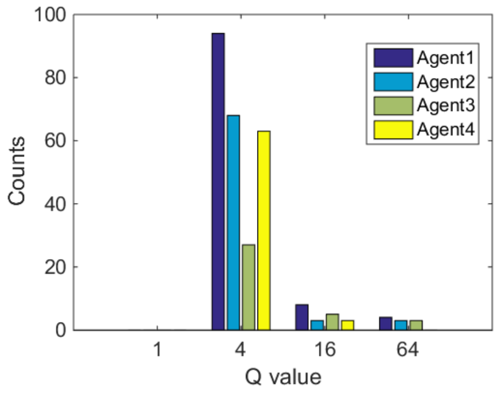


(a) Maximum number of information relations sending in Route "commuting"

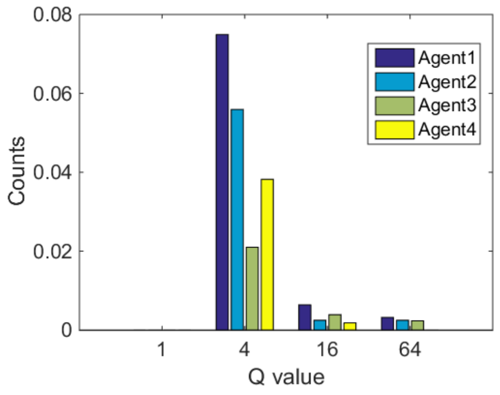


(b) Average number of information relations sending in Route "commuting"

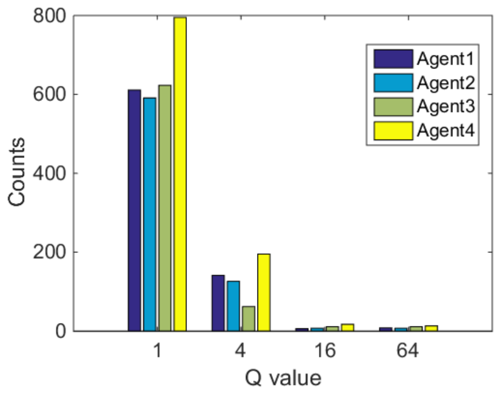
Figure 6.52: The number of the information relations sending in Route "commuting"



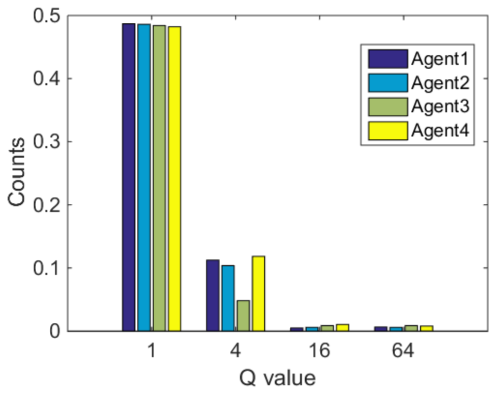
(a) Explicit conflict counts in Route "commuting"



(b) Average explicit conflict counts in Route "commuting"



(c) Missing relation chain conflict counts in Route "commuting"



(d) Average Missing relation chain conflict counts in Route "commuting"

Figure 6.53: EDCL conflict statistics in Route "commuting"

more than those in Routes “together” and “encountering”. In this walking pattern, the big Q shows its advantages than other low Q’s.

6.5.3 Conclusion

This section studies the localization performance for different algorithm, and the resource consumption for EDCL with different Q settings. The conclusion is that all the collaborative localization algorithms outperform the inertial-only algorithm significantly (reducing the error by 14.83% to 50.41%). Although EDCL is essentially not an optimal estimator, its accuracy and the estimation covariance matrix is very close to the optimal. Other important advantages of EDCL are: it is fully decentralized, and the resource consumption is low and controllable, both of which are very important in practice. For the resource usage, in our experiment case with $Q = 64$, the maximum memory usage is only 760 bytes, and the average communication cost is only 3 information relations which is $3 \times 11 = 33$ bytes.

Note that because of the negative covariance matrix problem, not all the inter-agent measurements are used. In Route “together”, there are totally 3989 inter-agent measurements generated by the RSS module, but 1032 are dropped. In Route “encountering”, there are totally 1940 inter-agent measurements, and 288 are dropped. In Route “commuting”, there are 1328 inter-agent measurements, and 12 are dropped. The root cause of the negative covariance matrix problem is that the ZUPT pseudo measurements are fused with the inertial-only data, which makes the generated measurements not independent with each other. If we decouple these two measurements, and make the ZUPT pseudo measurements as another type of information relation in the information relation graph, then the RSS measurements do not need to be dropped. However, the inertial navigation uses the different states as the ZUPT ($[\mathbf{x}, \mathbf{v}, \varphi]$ versus $[\delta\mathbf{x}, \delta\mathbf{v}, \delta\varphi]$). If we want to decouple them, a new space-state model needs to be built. This will be left for future work.

6.6 Discussion

This chapter presents an implementation for a multi-agent collaborative localization system, which is fully decentralized and self-contained. It consists of an inter-agent range estimator, an inertial navigation module and a measurement filter module. The inter-agent range is to generate the inter-agent measurement so that multiple agents's position estimation can be related; the inertial navigation module works as a dead reckoning module which provides the displacement measurement over time, and the measurement filter plays a role of fusing all measurements to produce the final position estimation. In practice, each module can be implemented in different ways according to different applications. In this chapter, for proof of concept and cost saving purpose, we use the radio RSS to distance mapping for the inter-agent range estimator, a foot-mounted inertial navigation system with ZUPT algorithm as the inertial navigation module, and EDCL as the measurement filter.

For RSS to distance mapping, we choose to use TelosB mote. In order to overcome the multipath effect, selective frequency effect and the body shadowing effect, several techniques are used. We place the mote on the top of a person's head which effectively avoids the body shadowing effect. By using moving median window technique, we reduce the noise of RSS signals. A novel frequency hopping TDMA protocol is proposed to overcome the multipath and selective frequency effects. Furthermore, a corrected truncated maximum likelihood distance estimator is developed to restrict the measurements in short distances. The combination of these methods makes the noisy RSS to distance mapping usable, whose mean error is around 1m.

For the foot-mounted inertial navigation system, we adopt the ZUPT method, which uses the zero velocity pseudo measurements in the stance phase of a step to correct the accumulated errors. In practice, the sensor characterization is the most important for this module. We use Allen Variance, and the simulated annealing algorithm to find the biases and other parameters. With this method, the final experimental result shows that the error is around 2% of the travel distance.

In order to evaluate the whole integrated system, we predefined three different walking pattern routes in a four floor building. The waypoints are marked on the way, and each person pushes the corresponding key on the laptop in order to record the time when he reaches the waypoint. Since multiple person and multiple modules are involved, the time synchronization between devices are important. By carefully design the packet format, we successfully associate the timestamps in different clock to anchor times. Another practical issue is that the number of measurement generated from the foot-mounted inertial navigation module is much more than the RSS to distance mapping module. By using the compounding operation, the measurement number is reduced from the order of time to the order of the inter-agent measurement number.

By post processing these data using different algorithms, we find all the collaborative localization algorithms outperform inertial-only method. With a proper Q setting, the accuracy of EDCL is close to the centralized optimal method INOVA. By studying the resource usage of EDCL, we find that EDCL is efficient in terms of the memory usage, the communication cost and the computational complexity. The full decentralized feature and the controllability of resource consumption makes EDCL useful in practice.

In the future, we plan to embed the code into each module so that they output the estimates independently. Additionally, a different choice of implementation is planned for some specific domain, e.g. the fire fighting. An improved filtering model will be studied in order to decouple the the ZUPT pseudo measurement and the inertial-only measurement. By this decoupling, no inter-agent measurements will be lost for avoiding the negative displacement covariance matrix issue.

Chapter 7

Conclusions

7.1 Summary of Dissertation

Wireless sensor networks (WSN) have already been used in many applications, such as environmental monitoring, health care, home automation, emergency support, vehicle networks, industry 4.0 and military tasks. Knowing nodes' positions is always a critical issue for these applications. Without the position information, the sensory data become meaningless, and a number of location based routing protocols would not work.

In the WSN field, the existing range-based or range-free localization solutions have one or more of the following drawbacks: 1) only including homogeneous positioning information; 2) only suitable for static WSNs, but not mobile WSNs or WSNs with high dynamic topology changes; 3) requiring pre-set infrastructures; 4) non-deterministic uncertainty. Another category of mobile node localization methods are from the robotics field. The representatives are simultaneous localization and mapping (SLAM) and the collaborative localization (CL). However, SLAM solution is not suitable for resource constrained WSN because of its iterative dynamic model, high computational complexity and not taking advantage the collaboration between nodes. The existing CL solutions suffer from either inefficiency or the over confidence problem.

The contributions of this dissertation have two parts: the theory part and the system part. For the theory part, this dissertation proposes a unified range-based localization mathematical model. With some variations, it can be applied to a large scale static WSN, to a large scale WSN with dynamic topology changes, or to a mobile WSN. The model addresses the following issues: 1) unified representation of various measurement types; 2) suitable for both mobile and static networks; 3) quantitative representation for uncertainty; 4) both centralized and decentralized architecture support; 5) providing not only position estimation, but also the estimation covariance matrix; 6) efficiency and scalability. Furthermore, by using this model, we also deduce how to use the quantitative uncertainty for confidence region inference and guiding the anchor selection process. For the system part, we target a most challenging problem – infrastructure free indoor multiple mobile agent localization. By using inertial sensors combining with RSS to distance mapping, a fully self-contained, decentralized and collaborative localization system of relatively high accuracy is successfully built, and the effectiveness and the efficiency of the above theory model are demonstrated.

For the theory part, we first model a position related measurement as a function of positions with random noise. The function could be a unary or binary operation (or with even more variables). Then all these measurements compose a measurement graph, where each vector represents a position state, and each edge represents a measurement with both the value and the corresponding covariance matrix. The measurement graph can be analogous to a generalized electrical network, where Ohmn's Law and the node voltage analysis technology are used. Then the covariance matrix of a measurement is analogous to a generalized resistance, the variance of a node's position estimation is analogous to the generalized voltage potential of this node's identity current graph, and the covariance between Nodes A and B is analogous to the generalized voltage potential of Node B in Node A's identity current graph.

By using the node voltage analysis technique from the electrical engineering field, removing or adding a node, or adding a new edge only involves a constant number of equations. By using this idea, the INOVA algorithm incrementally updates its Kirchhoff's matrix and vector to localize the nodes in the network. Hence, INOVA is suitable for localizing a large scale static WSN or a large scale WSN with high dynamic topology changes. The simulation results show that by using INOVA the computational complexity is reduced by 70 times for a 4000 measurements network comparing with the BLUE algorithm.

Using the same model, OSE-COV extends the OSE algorithm to localize a static WSN in a distributed way. The idea of OSE-COV is that each node treats its n hop neighbor node's voltage potential as 0, and keeps calculating and broadcasting its voltage potential. Together with OSE, after a number of iterations, each node's position estimation and covariance matrix estimation are asymptotically close to the ones calculated from INOVA.

When applying this unified model to a mobile network, each position at a specific time point is treated as a vertex in the measurement graph. For a centralized architecture, the historical vertices can be marginalized away without losing the estimation accuracy. However, in a decentralized architecture, the historical vertices which have correlations with other vertices should not be removed in order to have a consistent fusion with those agents in the future. This brings up the dilemma: to achieve an optimal estimation, it theoretically requires the infinite memory and computational capability. To balance the optimality and the resource consumption, the EDCL algorithm uses the marginalization factor Q to constrain the number of the measurements allowed to be stored in memory. When Q is big, EDCL is more likely to be optimal, but consumes more resources; on the other hand, when Q is small, EDCL is more resource efficient, but suffers more from the inconsistent fusion. The experimental results show that EDCL achieves a close to optimal localization accuracy with very limited resource consumption.

Since the covariance matrix plays an important role in our model, the property of the covariance matrix is studied. We prove that an estimation covariance matrix in a large scale WSN under our model follows the multivariate normal distribution by using the central limit theorem. Then we can use the covariance matrix to infer the confidence region of a estimation. Furthermore, by using the covariance matrix, an optimal strategy for anchor selection is deduced, i.e. always selecting the anchor which reduces the trace of the covariance matrix of the whole network the most. This selection strategy can be achieved in a decentralized way aligning with the OSE-COV algorithm.

For the system part, we build a prototype indoor pedestrian localization system, which is fully self-contained, decentralized and collaborative. The system consists of a RSS to distance estimator, a foot-mounted inertial navigation module and the EDCL filter. For RSS to distance mapping, we address the body shadowing effect, the selective frequency effect and the multipath effect by putting the device on top of a person's head, using a moving window median filter, proposing a frequency hopping TDMA protocol and developing a corrected truncated maximum likelihood distance estimator. The final experimental results show that the estimation error is around 1m within a distance below 7m. For the foot-mounted inertial navigation module, the ZUPT algorithm is used, which utilizes the truth–the velocity is zero during a stance phase in a step–to correct the accumulated inertial errors. The hardware characterization is achieved by the Allen Variance technique and the simulated annealing algorithm. The final results shows that the localization error is 1.8% of the total traveling distance.

To integrate these modules with multiple pedestrians, time synchronization is needed. By carefully designing the packet format, time synchronization between the inertial navigation module and the RSS module, and the RSS modules between different persons is achieved. Furthermore, a compounding operation is used to merge the inertial displacement measurements. After the compounding operation, the number of measurements is dramatically reduced from time dependent to the order of the number of the inter-agent measurements. The final experiments are conducted inside a four floor building with three different routes. The results show that all the collaborative localization algorithms outperform the inertial-only algorithm significantly. The localization accuracy of EDCL is close to the centralized optimal INOVA algorithm. When $Q = 64$, EDCL only consumes 760 bytes of memory at maximum, and average of 33 bytes for each communicating message.

7.2 Discussion and Future Work

There are several interesting research topics in both the theory and the system. For the theory part, following items are valuable for future work:

- Abstracting more measurement types: in this dissertation, a measurement is represented as a function of positions plus a random noise. The function can be unary, binary or even with more variables. We only study the most popular measurements, such as GPS, displacement, distance and angle. In the future, more types of measurements should be studied, such as velocities and wave phases. The function could involve more variables, e.g. the doppler effect based technique involves four nodes [3], and the triangulation algorithm involves at least three nodes.

- Supporting more positioning constraints: currently the model assumes the measurement is in an equation form, but in practice many constraints are not in equation forms. If this type of constraints can be supported, the model will be more generic. In optimization, the non-equation constraint has a well formed solution. We may borrow some ideas from there.

- Studying the practical use of the covariance matrix: the covariance matrix plays an important role in estimation models. However, few works systemically discuss how to set a proper covariance matrix in practice. By developing a systemical way for setting and evaluating a covariance matrix, it can not only improve the estimation accuracy, but also make the confidence region concept useful in practice.

- Studying the Q setting in EDCL: in EDCL, the marginalization factor Q is used to balance the approximation degree and the resource consumption. How to set the value of Q in practice? What is the impact if different agents have different Q's? Furthermore, Q can even be dynamic and thus the impact of the dynamic Q is also valuable to study.

For the system part, the research topics include:

- Embedding the module codes into devices: in the dissertation, modules in the indoor pedestrian localization system are connected by wires, and their raw data are collected into a central place and post processed. In order to make the system more usable, we have to embed the module codes into each device, and let each device only output its processed estimation. Furthermore, the cable connections should be replaced by wireless connections for user convenience. A more user friendly enclosure and interface should be designed too.

- Using different hardware for different applications: the prototype system in this dissertation is for proof of concept purposes. In practice, different applications may need different hardware devices. For example, firefighter localization may need a more accurate inter-agent module, a laser-based range finder or the ultrasonic and electromagnetic wave based TODA technology may be a proper choice. For location-base socialization, a cell phone should be the main device, and using the magnetic field of the earth may bring exceptional results.

- Establishing a more general state model: as mentioned in Section 6.5.3, many inter-agent measurements are dropped because of the negative covariance matrix issue. The root cause of this issue is that the ZUPT pseudo measurement is fused with the inertial only measurements, which makes the measurements not independent with each other. In order to use all inte-agent measurements, we should separate a ZUPT measurement from the inertial only measurements as a fully independent measurement. This may require changing the state model from the error state $[\delta\mathbf{x}, \delta\mathbf{v}, \delta\varphi]$ to the normal state $[\mathbf{x}, \mathbf{v}, \varphi]$.

- Integrating with the breadcrumb system: the indoor pedestrian localization system in this dissertation only makes use of the collaboration between the moving agents. However, if a breadcrumb system [33] is integrated, it can not only help to improve the accuracy of the localization, but also be used as a communication relayer network.

- Calibrating inertial sensors on site: in the experiment, we find that the pre-calculated sensor parameters are not accurate. Even the parameters estimated on site are not guaranteed to be correct. Therefore, a good enclosure (to prevent the deformation of the sensor boards) and a more robust calibration method are needed.

- Initializing the yaw: since the yaw cannot be estimated merely by the inertial sensors, the magnetic sensors may be used for reference. If a map is provided, the initial yaw can also be obtained by aligning the walking direction with the path in map.

- Generalizing the RSS parameter setting: in the dissertation, the RSS parameter (path loss factor) is calculated based on the pre-measured data. In reality, some applications may require finding the right parameters on site. Therefore, if the module can learn the parameters itself through some reference or feedbacks, that would be helpful.

Chapter 8

Bibliography

[1] H. T. Kung, C.-K. Lin, T.-H. Lin, and D. Vlah, “Localization with snap-inducing shaped residuals (sisr): coping with errors in measurement,” in *Proceedings of the 15th annual international conference on Mobile computing and networking*, MobiCom ’09, (New York, NY, USA), pp. 333–344, ACM, 2009.

[2] D. Niculescu and B. Nath, “Ad hoc positioning system (aps) using aoa,” in *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, vol. 3, pp. 1734–1743 vol.3, 2003.

[3] M. Maróti, P. Völgyesi, S. Dóra, B. Kusý, A. Nádas, A. Lédeczi, G. Balogh, and K. Molnár, “Radio interferometric geolocation,” in *Proceedings of the 3rd international conference on Embedded networked sensor systems*, SenSys ’05, (New York, NY, USA), pp. 1–12, ACM, 2005.

[4] T. He, C. Huang, B. M. Blum, J. A. Stankovic, and T. Abdelzaher, “Range-free localization schemes for large scale sensor networks,” in *Proceedings of the 9th annual international conference on Mobile computing and networking (MobiCom ’03)*, (New York, NY, USA), pp. 81–95, ACM, September 2003.

[5] L. K. Saul and S. T. Roweis, “Think globally, fit locally: Unsupervised learning of low dimensional manifolds,” *J. Mach. Learn. Res.*, vol. 4, pp. 119–155, Dec. 2003.

[6] J. B. Tenenbaum, V. de Silva, and J. C. Langford3, “A global geometric framework for nonlinear dimensionality reduction,” *Science*, vol. 290, no. 5500, pp. 2319–2323, 2000.

[7] L. Doherty, K. pister, and L. El Ghaoui, “Convex position estimation in wireless sensor networks,” in *INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 3, pp. 1655–1663 vol.3, 2001.

[8] M. Li and Y. Liu, “Rendered path: Range-free localization in anisotropic sensor networks with holes,” *Networking, IEEE/ACM Transactions on*, vol. 18, pp. 320–332, Feb 2010.

[9] K. Römer, “The lighthouse location system for smart dust,” in *MobiSys ’03: Proceedings of the 1st international conference on Mobile systems, applications and services*, (New York, NY, USA), pp. 15–30, ACM, 2003.

[10] Z. Zhong, D. Wang, and T. He, “Sensor node localization using uncontrolled events,” in *Distributed Computing Systems, 2008. ICDCS ’08. The 28th International Conference on*, pp. 438–445, June 2008.

[11] H. Durrant-Whyte and T. Bailey, “Simultaneous localisation and mapping (slam): Part i the essential algorithms,” *IEEE Robotics And Automation Magazine*, 2006.

[12] S. Julier and J. Uhlmann, “A non-divergent estimation algorithm in the presence of unknown correlations,” in *American Control Conference, 1997. Proceedings of the 1997*, vol. 4, pp. 2369–2373 vol.4, Jun 1997.

[13] “Telosb datasheet. http://www4.ncsu.edu/ kkolla/csc714/datasheet.pdf.”

[14] "Ieee standard specification format guide and test procedure for single-axis interferometric fiber optic gyros," *IEEE Std 952-1997*, pp. i–, 1998.

[15] G. Tolle, J. Polastre, R. Szewczyk, D. Culler, N. Turner, K. Tu, S. Burgess, T. Dawson, P. Buonadonna, D. Gay, and W. Hong, "A macroscope in the redwoods," in *Proceedings of the 3rd international conference on Embedded networked sensor systems*, SenSys '05, (New York, NY, USA), pp. 51–63, ACM, 2005.

[16] G. Werner-Allen, K. Lorincz, M. Ruiz, O. Marcillo, J. Johnson, J. Lees, and M. Welsh, "Deploying a wireless sensor network on an active volcano," *Internet Computing, IEEE*, vol. 10, pp. 18 – 25, march-april 2006.

[17] L. Selavo, A. Wood, Q. Cao, T. Sookoor, H. Liu, A. Srinivasan, Y. Wu, W. Kang, J. Stankovic, D. Young, and J. Porter, "Luster: wireless sensor network for environmental research," in *Proceedings of the 5th international conference on Embedded networked sensor systems*, SenSys '07, (New York, NY, USA), pp. 103–116, ACM, 2007.

[18] L. Mo, Y. He, Y. Liu, J. Zhao, S.-J. Tang, X.-Y. Li, and G. Dai, "Canopy closure estimates with greenorbs: sustainable sensing in the forest," in *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*, SenSys '09, (New York, NY, USA), pp. 99–112, ACM, 2009.

[19] D. Bhadauria, V. Isler, A. Studenski, and P. Tokekar, "A robotic sensor network for monitoring carp in minnesota lakes," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pp. 3837 –3842, may 2010.

[20] S. Iyengar, F. T. Bonda, R. Gravina, A. Guerrieri, G. Fortino, and A. Sangiovanni-Vincentelli, "A framework for creating healthcare monitoring applications using wireless body sensor networks," in *Proceedings of the ICST 3rd international conference on Body area networks*, BodyNets '08, (ICST, Brussels, Belgium, Belgium), pp. 8:1–8:2, ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2008.

[21] Q. Li, J. Stankovic, M. Hanson, A. Barth, J. Lach, and G. Zhou, "Accurate, fast fall detection using gyroscopes and accelerometer-derived posture information," in *Wearable and Implantable Body Sensor Networks, 2009. BSN 2009. Sixth International Workshop on*, pp. 138 –143, june 2009.

[22] J. Ko, C. Lu, M. Srivastava, J. Stankovic, A. Terzis, and M. Welsh, "Wireless sensor networks for healthcare," *Proceedings of the IEEE*, vol. 98, pp. 1947 –1960, nov. 2010.

[23] E. Hoque, R. F. Dickerson, and J. A. Stankovic, "Monitoring body positions and movements during sleep using wisps," in *Wireless Health 2010*, WH '10, (New York, NY, USA), pp. 44–53, ACM, 2010.

[24] R. F. Dickerson, E. I. Gorlin, and J. A. Stankovic, "Empath: a continuous remote emotional health monitoring system for depressive illness," in *Proceedings of the 2nd Conference on Wireless Health*, WH '11, (New York, NY, USA), pp. 5:1–5:10, ACM, 2011.

[25] J. Lu, T. Sookoor, V. Srinivasan, G. Gao, B. Holben, J. Stankovic, E. Field, and K. Whitehouse, "The smart thermostat: using occupancy sensors to save energy in homes," in *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*, SenSys '10, (New York, NY, USA), pp. 211–224, ACM, 2010.

[26] J. Lu and K. Whitehouse, "Smart blueprints: automatically generated maps of homes and the devices within them," in *Proceedings of the 10th international conference on Pervasive Computing*, Pervasive'12, (Berlin, Heidelberg), pp. 125–142, Springer-Verlag, 2012.

[27] V. Smith, T. Sookoor, and K. Whitehouse, "Modeling building thermal response to hvac zoning," *SIGBED Rev.*, vol. 9, pp. 39–45, July 2012.

[28] J. Lu and K. Whitehouse, "Suncast: fine-grained prediction of natural sunlight levels for improved daylight harvesting," in *Proceedings of the 11th international conference*

on Information Processing in Sensor Networks, IPSN '12, (New York, NY, USA), pp. 245–256, ACM, 2012.

[29] K. Whitehouse, J. Ranjan, J. Lu, T. Sookoor, M. Saadat, C. Burke, G. Staengl, A. Canfora, and H. Haj-Hariri, "Towards occupancy-driven heating and cooling," *Design Test of Computers, IEEE*, vol. 29, pp. 17 –25, aug. 2012.

[30] M. Klann, T. Riedel, H. Gellersen, C. Fischer, M. Oppenheim, P. Lukowicz, G. Pirkl, K. Kunze, M. Beuster, M. Beigl, O. Visser, and M. Gerling, "Lifenet: an ad-hoc sensor network and wearable system to provide firefighters with navigation support," in *Ubicomp 2007*, 2007.

[31] M. R. Souryal, J. Geissbuehler, L. E. Miller, and N. Moayeri, "Real-time deployment of multihop relays for range extension," in *Proceedings of the 5th international conference on Mobile systems, applications and services*, MobiSys '07, (New York, NY, USA), pp. 85–98, ACM, 2007.

[32] V. Amendolare, D. Cyganski, R. Duckworth, S. Makarov, J. Coyne, H. Daempfling, and B. Woodacre, "Wpi precision personnel locator system: Inertial navigation supplementation," in *Position, Location and Navigation Symposium*, pp. 350 –357, May 2008.

[33] H. Liu, J. Li, Z. Xie, S. Lin, K. Whitehouse, J. A. Stankovic, and D. Siu, "Automatic and robust breadcrumb system deployment for indoor firefighter applications," in *Proceedings of the 8th international conference on Mobile systems, applications, and services*, MobiSys '10, (New York, NY, USA), pp. 21–34, ACM, 2010.

[34] A. Purohit, Z. Sun, F. Mokaya, and P. Zhang, "Sensorfly: Controlled-mobile sensing platform for indoor emergency response applications," in *Information Processing in Sensor Networks (IPSN), 2011 10th International Conference on*, pp. 223 –234, april 2011.

[35] J. Jeong, S. Guo, T. He, and D. Du, "Apl: Autonomous passive localization for wireless sensors deployed in road networks," in *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, pp. 583 –591, april 2008.

[36] X. Li, W. Shu, M. Li, H.-Y. Huang, P.-E. Luo, and M.-Y. Wu, "Performance evaluation of vehicle-based mobile sensor networks for traffic monitoring," *Vehicular Technology, IEEE Transactions on*, vol. 58, pp. 1647 –1653, may 2009.

[37] J. Knuth and P. Barooah, "Distributed collaborative localization of multiple vehicles from relative pose measurements," in *Communication, Control, and Computing, 2009. Allerton 2009. 47th Annual Allerton Conference on*, pp. 314 –321, 30 2009-oct. 2 2009.

[38] K. S. J. Pister, "Military applications of sensor networks," in *of Institute for Defense Analyses Paper P-3531, Defense Science Study Group*, 2000.

[39] A. Lédeczi, A. Nádas, P. Völgyesi, G. Balogh, B. Kusy, J. Sallai, G. Pap, S. Dóra, K. Molnár, M. Maróti, and G. Simon, "Countersniper system for urban warfare," *ACM Trans. Sen. Netw.*, vol. 1, pp. 153–177, Nov. 2005.

[40] T. He, S. Krishnamurthy, L. Luo, T. Yan, L. Gu, R. Stoleru, G. Zhou, Q. Cao, P. Vicaire, J. A. Stankovic, T. F. Abdelzaher, J. Hui, and B. Krogh, "Vigilnet: An integrated sensor network system for energy-efficient surveillance," *ACM Trans. Sen. Netw.*, vol. 2, pp. 1–38, Feb. 2006.

[41] P. Volgyesi, G. Balogh, A. Nadas, C. B. Nash, and A. Ledeczi, "Shooter localization and weapon classification with soldier-wearable networked sensors," in *Proceedings of the 5th international conference on Mobile systems, applications and services*, MobiSys '07, (New York, NY, USA), pp. 113–126, ACM, 2007.

[42] Y.-B. Ko and N. H. Vaidya, "Location-aided routing (lar) in mobile ad hoc networks," in *Proceedings of the 4th annual ACM/IEEE international conference on Mobile*

computing and networking (MobiCom '98), (New York, NY, USA), pp. 66–75, ACM, Octobor 1998.

[43] B. Karp and H. T. Kung, "Gpsr: greedy perimeter stateless routing for wireless networks," in *Proceedings of the 6th annual international conference on Mobile computing and networking*, MobiCom '00, (New York, NY, USA), pp. 243–254, ACM, 2000.

[44] Y. Xu, J. Heidemann, and D. Estrin, "Geography-informed energy conservation for ad hoc routing," in *Proceedings of the 7th annual international conference on Mobile computing and networking (MobiCom '01)*, (New York, NY, USA), pp. 70–84, ACM, July 2001.

[45] Y.-J. Kim, R. Govindan, B. Karp, and S. Shenker, "Geographic routing made practical," in *Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation - Volume 2*, NSDI'05, (Berkeley, CA, USA), pp. 217–230, USENIX Association, 2005.

[46] P. A. Vicaire, Z. Xie, E. Hoque, and J. A. Stankovic, "Physicalnet: A generic framework for managing and programming across pervasive computing networks," in *RTAS*, (Stockholm, Sweden), 2010.

[47] P. A. Vicaire, E. Hoque, Z. Xie, and J. A. Stankovic, "Bundle: a group based programming abstraction for cyber physical systems," in *Proceedings of the 1st ACM/IEEE International Conference on Cyber-Physical Systems*, ICCPS '10, (New York, NY, USA), pp. 32–41, ACM, 2010.

[48] V. Srinivasan, J. Stankovic, and K. Whitehouse, "Using height sensors for biometric identification in multi-resident homes," in *Proceedings of The 8th International Conference on Pervasive Computing (Pervasive 2010)*, (Helsinki, Finland), May 2010.

[49] T. W. Hnat, E. Griffiths, R. Dawson, and K. Whitehouse, "Doorjamb: unobtrusive room-level tracking of people in homes using doorway sensors," in *Proceedings of the*

10th ACM Conference on Embedded Network Sensor Systems, SenSys '12, (New York, NY, USA), pp. 309–322, ACM, 2012.

[50] D. Daly, T. Melia, and G. Baldwin, "Concrete embedded rfid for way-point positioning," in *Indoor Positioning and Indoor Navigation (IPIN), 2010 International Conference on*, pp. 1 –10, sept. 2010.

[51] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan, "The cricket location-support system," in *Proceedings of the 6th annual international conference on Mobile computing and networking (MobiCom '00)*, (New York, NY, USA), pp. 32–43, ACM, August 2000.

[52] A. Savvides, C.-C. Han, and M. B. Strivastava, "Dynamic fine-grained localization in ad-hoc networks of sensors," in *Proceedings of the 7th annual international conference on Mobile computing and networking (MobiCom '01)*, (New York, NY, USA), pp. 166–179, ACM, July 2001.

[53] X. Cheng, A. Thaeler, G. Xue, and D. Chen, "Tps: A time-based positioning scheme for outdoor wireless sensor networks," in *INFOCOM 2004: Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, pp. 2685–2696, March 2004.

[54] M. Maróti, P. Völgyesi, S. Dóra, B. Kusý, A. Nádas, A. Lédeczi, G. Balogh, and K. Molnár, "Radio interferometric geolocation," in *SenSys '05: Proceedings of the 3rd international conference on Embedded networked sensor systems*, (New York, NY, USA), pp. 1–12, ACM, 2005.

[55] S. Lanzisera, D. T. Lin, and K. S. J. Pister, "Rf time of flight ranging for wireless sensor network localization," in *Workshop on Intelligent Solutions in Embedded Systems (WISES '06)*, pp. 1–12, June 2006.

[56] J. Liu, Y. Zhang, and F. Zhao, "Robust distributed node localization with error management," in *MobiHoc '06: Proceedings of the 7th ACM international symposium on Mobile ad hoc networking and computing*, (New York, NY, USA), pp. 250–261, ACM, 2006.

[57] X. Cheng, H. Shu, Q. Liang, and D. H.-C. Du, "Silent positioning in underwater acoustic sensor networks," *IEEE Transactions on Vehicular Technology*, vol. 57, p-p. 1756–1766, May 2008.

[58] D. Niculescu and B. Nath, "Dv based positioning in ad hoc networks," *Journal of Telecommunication System*, 2003.

[59] L. I. On, R. Nagpal, H. Shrobe, and J. Bachrach, "Organizing a global coordinate system from," in *IPSN'03*, pp. 333–348, 2003.

[60] R. Stoleru, T. He, J. A. Stankovic, and D. Luebke, "A high-accuracy, low-cost localization system for wireless sensor networks," in *SenSys '05: Proceedings of the 3rd international conference on Embedded networked sensor systems*, (New York, NY, USA), pp. 13–26, ACM, 2005.

[61] R. Stoleru, P. Vicaire, T. He, and J. A. Stankovic, "Stardust: A flexible architecture for passive localization in wireless sensor networks," in *ACM Conference on Embedded Networked Sensor Systems (SenSys*, pp. 57–70, ACM Press, 2006.

[62] C. Wang and L. Xiao, "Locating sensors in concave areas," in *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings*, pp. 1 –12, apr. 2006.

[63] Z. Zhong and T. He, "Msp: multi-sequence positioning of wireless sensor nodes," in *SenSys '07: Proceedings of the 5th international conference on Embedded networked sensor systems*, (New York, NY, USA), pp. 15–28, ACM, 2007.

[64] M. Li and Y. Liu, "Rendered path: range-free localization in anisotropic sensor networks with holes," in *MobiCom '07: Proceedings of the 13th annual ACM international conference on Mobile computing and networking*, (New York, NY, USA), pp. 51–62, ACM, 2007.

[65] S. Lederer, Y. Wang, and J. Gao, "Connectivity-based localization of large-scale sensor networks with complex shape," *ACM Trans. Sen. Netw.*, vol. 5, pp. 31:1–31:32, Nov. 2009.

[66] Z. Zhong and T. He, "Achieving range-free localization beyond connectivity," in *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems (SenSys '09)*, (New York, NY, USA), pp. 281–294, ACM, November 2009.

[67] Y. Kwon, K. Mechitov, S. Sundresh, W. Kim, and G. Agha, "Resilient localization for sensor networks in outdoor environments," *ACM Trans. Sen. Netw.*, vol. 7, pp. 3:1–3:30, Aug. 2010.

[68] M. Belkin and P. Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation," *Neural Comput.*, vol. 15, pp. 1373–1396, June 2003.

[69] P. Barooah and J. Hespanha, "Estimation on graphs from relative measurements," *Control Systems Magazine, IEEE*, vol. 27, pp. 57 –74, August 2007.

[70] H. Nurminen, A. Ristimaki, S. Ali-Loytty, and R. Piche, "Particle filter and smoother for indoor localization," in *Indoor Positioning and Indoor Navigation (IPIN), 2013 International Conference on*, pp. 1–10, Oct 2013.

[71] M. Khan and J. Syrjarinne, "Investigating effective methods for integration of building's map with low cost inertial sensors and wifi-based positioning," in *Indoor Positioning and Indoor Navigation (IPIN), 2013 International Conference on*, pp. 1–8, Oct 2013.

[72] O. Woodman and R. Harle, "Pedestrian localisation for indoor environments," in *Proceedings of the 10th International Conference on Ubiquitous Computing*, UbiComp '08, (New York, NY, USA), pp. 114–123, ACM, 2008.

[73] S. Kaiser, M. Khider, and P. Robertson, "A maps-based angular pdf for navigation systems in indoor and outdoor environments," in *Indoor Positioning and Indoor Navigation (IPIN), 2011 International Conference on*, pp. 1–7, Sept 2011.

[74] H. Durrant-Whyte and T. Bailey, "Simultaneous localisation and mapping (slam): Part ii - state of the art," *IEEE Robotics And Automation Magazine*, 2006.

[75] R. Kurazume and S. Hirose, "Study on cooperative positioning system: optimum moving strategies for cps-iii," in *IEEE International Conference on Robotics and Automation*, vol. 4, pp. 2896 –2903 vol.4, may 1998.

[76] D. Fox, W. Burgard, H. Kruppa, and S. Thrun, "A probabilistic approach to collaborative multi-robot localization," *Auton. Robots*, vol. 8, pp. 325–344, June 2000.

[77] A. Bahr, M. Walter, and J. Leonard, "Consistent cooperative localization," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3415 –3422, May 2009.

[78] T. Bailey, M. Bryson, H. Mu, J. Vial, L. McCalman, and H. Durrant-Whyte, "Decentralised cooperative localisation for heterogeneous teams of mobile robots," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2859 –2865, May 2011.

[79] G. E. Moore, "Cramming more components onto integrated circuits," *Electronics Magazine*, Nov. 11, 2006.

[80] P. Barooah and J. Hespanha, "Estimation from relative measurements: Electrical analogy and large graphs," *Signal Processing, IEEE Transactions on*, vol. 56, no. 6, pp. 2181–2193, 2008.

[81] R. B. Ash and C. A. Doléans-Dade, *Probability and Measure Theory (2nd ed.)*. San Diego: Harcourt/Academic Press, 2000.

[82] F. Gustafsson, F. Gunnarsson, and D. Lindgren, "Sensor models and localization algorithms for sensor networks based on received signal strength," *EURASIP Journal on Wireless Communications and Networking*, vol. 16, Jan. 2012.

[83] A. Savvides, C.-C. Han, and M. B. Strivastava, "Dynamic fine-grained localization in ad-hoc networks of sensors," in *Proceedings of the 7th annual international conference on Mobile computing and networking*, MobiCom '01, (New York, NY, USA), pp. 166–179, ACM, 2001.

[84] K. Whitehouse, C. Karlof, A. Woo, F. Jiang, and D. Culler, "The effects of ranging noise on multihop localization: an empirical study," in *Proceedings of the 4th international symposium on Information processing in sensor networks*, IPSN '05, (Piscataway, NJ, USA), IEEE Press, 2005.

[85] K. Whitehouse, C. Karlof, and D. Culler, "A practical evaluation of radio signal strength for ranging-based localization," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 11, pp. 41–52, Jan. 2007.

[86] N. Patwari, H. A. O. III, and J. A. Costa, "Learning sensor location from signal strength and connectivity," *Advances in Information Security*, vol. 30, 2007.

[87] E. Miluzzo, X. Zheng, K. Fodor, and A. T. Campbell, "Radio characterization of 802.15.4 and its impact on the design of mobile sensor networks," in *Proceedings of the 5th European conference on Wireless sensor networks*, EWSN'08, (Berlin, Heidelberg), pp. 171–188, Springer-Verlag, 2008.

[88] R. M. Vaghefi, M. R. Gholami, R. M. Buehrer, and E. G. Strom, "Cooperative received signal strength-based sensor localization with unknown transmit powers," *Signal Processing, IEEE Transactions on*, vol. 61, pp. 1389 –1403, march15, 2013.

[89] J. Zhao, W. Xi, Y. He, Y. Liu, X.-Y. Li, L. Mo, and Z. Yang, "Localization of wireless sensor networks in the wild: Pursuit of ranging quality," *Networking, IEEE/ACM Transactions on*, vol. 21, pp. 311 –323, feb. 2013.

[90] H. Friis, "A note on a simple transmission formula," *Proceedings of the IRE*, vol. 34, pp. 254 – 256, may 1946.

[91] T. S. Rappaport, *Wireless communications, principles and practice.* Prentice Hall, 1996.

[92] L. Jian, Z. Yang, and Y. Liu, "Beyond triangle inequality: Sifting noisy and outlier distance measurements for localization," in *INFOCOM, 2010 Proceedings IEEE*, pp. 1 –9, march 2010.

[93] Y. Shang, W. Ruml, Y. Zhang, and M. P. J. Fromherz, "Localization from mere connectivity," in *Proceedings of the 4th ACM international symposium on Mobile ad hoc networking & computing*, MobiHoc '03, (New York, NY, USA), pp. 201–212, ACM, 2003.

[94] J. Costa, N. Patwari, and A. Hero, "Achieving high-accuracy distributed localization in sensor networks," in *Acoustics, Speech, and Signal Processing, 2005. Proceedings. (ICASSP '05). IEEE International Conference on*, vol. 3, pp. iii/641–iii/644 Vol. 3, March.

[95] N. Patwari and A. Hero, "Adaptive neighborhoods for manifold learning-based sensor localization," in *Signal Processing Advances in Wireless Communications, 2005 IEEE 6th Workshop on*, pp. 1098–1102, June.

[96] F. R. Hampel, E. M. Ronchetti, R. P. J., and W. A. Stahel, *Robust Statistics: The Approach Based on Influence Functions.* Wiley New York, 2003.

[97] Y. Shang, W. Rumi, Y. Zhang, and M. Fromherz, “Localization from connectivity in sensor networks,” *Parallel and Distributed Systems, IEEE Transactions on*, vol. 15, no. 11, pp. 961–974, Nov.

[98] A. Harter, A. Hopper, P. Steggles, A. Ward, and P. Webster, “The anatomy of a context-aware application,” in *Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, MobiCom ’99, (New York, NY, USA), pp. 59–68, ACM, 1999.

[99] L. Girod and D. Estrin, “Robust range estimation using acoustic and multimodal sensing,” in *Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on*, vol. 3, pp. 1312–1320 vol.3, 2001.

[100] C. Peng, G. Shen, Y. Zhang, Y. Li, and K. Tan, “Beepbeep: a high accuracy acoustic ranging system using cots mobile devices,” in *Proceedings of the 5th international conference on Embedded networked sensor systems*, SenSys ’07, (New York, NY, USA), pp. 1–14, ACM, 2007.

[101] X. Cheng, H. Shu, Q. Liang, and D.-C. Du, “Silent positioning in underwater acoustic sensor networks,” *Vehicular Technology, IEEE Transactions on*, vol. 57, no. 3, pp. 1756–1766, 2008.

[102] W. Cheng, A. Thaeler, X. Cheng, F. Liu, X. Lu, and Z. Lu, “Time-synchronization free localization in large scale underwater acoustic sensor networks,” in *Distributed Computing Systems Workshops, 2009. ICDCS Workshops ’09. 29th IEEE International Conference on*, pp. 80–87, 2009.

[103] B. Liu, H. Chen, Z. Zhong, and H. Poor, “Asymmetrical round trip based synchronization-free localization in large-scale underwater sensor networks,” *Wireless Communications, IEEE Transactions on*, vol. 9, no. 11, pp. 3532–3542, 2010.

[104] M. Win and R. Scholtz, "Impulse radio: how it works," *Communications Letters, IEEE*, vol. 2, no. 2, pp. 36–38, 1998.

[105] D. McCrady, L. Doyle, H. Forstrom, T. Dempsey, and M. Martorana, "Mobile ranging using low-accuracy clocks," *Microwave Theory and Techniques, IEEE Transactions on*, vol. 48, no. 6, pp. 951–958, 2000.

[106] R. Fontana and S. Gunderson, "Ultra-wideband precision asset location system," in *Ultra Wideband Systems and Technologies, 2002. Digest of Papers. 2002 IEEE Conference on*, pp. 147–150, 2002.

[107] J.-Y. Lee and R. A. Scholtz, "Ranging in a dense multipath environment using an uwb radio link," *IEEE J.Sel. A. Commun.*, vol. 20, pp. 1677–1683, Sept. 2006.

[108] S. Gezici, Z. Tian, G. Giannakis, H. Kobayashi, A. Molisch, H. Poor, and Z. Sahinoglu, "Localization via ultra-wideband radios: a look at positioning aspects for future sensor networks," *Signal Processing Magazine, IEEE*, vol. 22, no. 4, pp. 70–84, 2005.

[109] M. Youssef, A. Youssef, C. Rieger, U. Shankar, and A. Agrawala, "Pinpoint: An asynchronous time-based location determination system," in *Proceedings of the 4th international conference on Mobile systems, applications and services*, MobiSys '06, (New York, NY, USA), pp. 165–176, ACM, 2006.

[110] T. Karalar and J. Rabaey, "An rf tof based ranging implementation for sensor networks," in *Communications, 2006. ICC '06. IEEE International Conference on*, vol. 7, pp. 3347–3352, 2006.

[111] S. Lanzisera, D. Lin, and K. Pister, "Rf time of flight ranging for wireless sensor network localization," in *Intelligent Solutions in Embedded Systems, 2006 International Workshop on*, pp. 1–12, 2006.

[112] D. Jourdan, *Wireless sensor network planning with application to UWB localization in GPS-denied environments.* PhD thesis, Massachusetts Institute of Technolog, 2006.

[113] P. Kułakowski, J. Vales-Alonso, E. E.-L. ópez, W. Ludwin, and J. García-Haro, "Angle-of-arrival localization based on antenna arrays for wireless sensor networks," *Computers & Electrical Engineering*, vol. 36, no. 6, pp. 1181 – 1186, 2010.

[114] N. B. Priyantha, A. K. Miu, H. Balakrishnan, and S. Teller, "The cricket compass for context-aware mobile applications," in *Proceedings of the 7th annual international conference on Mobile computing and networking*, MobiCom '01, (New York, NY, USA), pp. 1–14, ACM, 2001.

[115] S. Elvira, A. de Castro, and J. Garrido, "Alo: An ultrasound system for localization and orientation based on angles," *Microelectronics Journal*, 2013.

[116] H.-l. Chang, J.-b. Tian, T.-T. Lai, H.-H. Chu, and P. Huang, "Spinning beacons for precise indoor localization," in *Proceedings of the 6th ACM conference on Embedded network sensor systems*, SenSys '08, (New York, NY, USA), pp. 127–140, ACM, 2008.

[117] A. Basu, J. Gao, J. S. B. Mitchell, and G. Sabhnani, "Distributed localization using noisy distance and angle information," in *Proceedings of the 7th ACM international symposium on Mobile ad hoc networking and computing*, MobiHoc '06, (New York, NY, USA), pp. 262–273, ACM, 2006.

[118] A. Bishop, P. Pathirana, B. Fidan, B. D. O. Anderson, and G. Mao, "Passive angle measurement based localizationconsistency via geometric constraints," in *Information, Decision and Control, 2007. IDC '07*, pp. 199–204, 2007.

[119] B. Kusy, A. Ledeczi, M. Maroti, and L. Meertens, "Node density independent localization," in *Proceedings of the 5th international conference on Information processing in sensor networks*, IPSN '06, (New York, NY, USA), pp. 441–448, ACM, 2006.

[120] B. Kusy, G. Balogh, J. Sallai, A. Lédeczi, and M. Maróti, "Intrack: high precision tracking of mobile sensor nodes," in *Proceedings of the 4th European conference on*

Wireless sensor networks, EWSN'07, (Berlin, Heidelberg), pp. 51–66, Springer-Verlag, 2007.

[121] B. Kusy, J. Sallai, G. Balogh, A. Ledeczi, V. Protopopescu, J. Tolliver, F. DeNap, and M. Parang, "Radio interferometric tracking of mobile wireless nodes," in *Proceedings of the 5th international conference on Mobile systems, applications and services*, MobiSys '07, (New York, NY, USA), pp. 139–151, ACM, 2007.

[122] N. Bulusu, J. Heidemann, and D. Estrin, "Gps-less low-cost outdoor localization for very small devices," *Personal Communications, IEEE*, vol. 7, no. 5, pp. 28–34, 2000.

[123] L. Ni, Y. Liu, Y. C. Lau, and A. Patil, "Landmarc: indoor location sensing using active rfid," in *Pervasive Computing and Communications, 2003. (PerCom 2003). Proceedings of the First IEEE International Conference on*, pp. 407–415, 2003.

[124] J. Blumenthal, R. Grossmann, F. Golatowski, and D. Timmermann, "Weighted centroid localization in wlan-based sensor networks," in *Proceedings of the 2007 IEEE International Symposiumon Intelligent Signal Processing (WISP2007)*, 2007.

[125] S. Schuhmann, K. Herrmann, K. Rothermel, J. Blumenthal, and D. Timmermann, "Improved weighted centroid localization in smart ubiquitous environments," in *Proceedings of the 5th international conference on Ubiquitous Intelligence and Computing*, UIC '08, (Berlin, Heidelberg), pp. 20–34, Springer-Verlag, 2008.

[126] M. Rudafshani and S. Datta, "Localization in wireless sensor networks," in *Information Processing in Sensor Networks, 2007. IPSN 2007. 6th International Symposium on*, pp. 51–60, 2007.

[127] D. Niculescu and B. Nath, "Ad hoc positioning system (aps)," in *Global Telecommunications Conference, 2001*, pp. 2926–2931 vol.5, 2001.

[128] K. Langendoen and N. Reijers, "Distributed localization in wireless sensor networks: A quantitative comparison," *Computer Networks: The International Journal of Computer and Telecommunications Networking*, vol. 43, no. 4, pp. 499–518, 2003.

[129] A. Savvides, H. Park, and M. B. Srivastava, "The bits and flops of the n-hop multilateration primitive for node localization problems," in *Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications*, WSNA '02, (New York, NY, USA), pp. 112–121, ACM, 2002.

[130] R. Nagpal, H. Shrobe, and J. Bachrach, "Organizing a global coordinate system from local information on an ad hoc sensor network," in *Proceedings of the 2Nd International Conference on Information Processing in Sensor Networks*, IPSN'03, (Berlin, Heidelberg), pp. 333–348, Springer-Verlag, 2003.

[131] C. Savarese, J. M. Rabaey, and K. Langendoen, "Robust positioning algorithms for distributed ad-hoc wireless sensor networks," in *Proceedings of the General Track of the Annual Conference on USENIX Annual Technical Conference*, ATEC '02, (Berkeley, CA, USA), pp. 317–327, USENIX Association, 2002.

[132] A. Karbasi and S. Oh, "Distributed sensor network localization from local connectivity: Performance analysis for the hop-terrain algorithm," in *Proceedings of the ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, SIGMETRICS '10, (New York, NY, USA), pp. 61–70, ACM, 2010.

[133] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, no. 5500, pp. 2323–2326, 2000.

[134] D. L. Donoho and C. Grimes, "Hessian eigenmaps: new locally linear embedding techniques for high-dimensional data," technical report tr2003-08, Department of Statistics, Stanford University, Mar. 2003.

[135] J. A. Costa, N. Patwari, and A. O. H. III, “Distributed multidimensional scaling with adaptive weighting for node localization in sensor networks,” *IEEE/ACM Transactions on Sensor Networks*, May 2006.

[136] L. Li and T. Kunz, “Cooperative node localization using nonlinear data projection,” *ACM Trans. Sen. Netw.*, vol. 5, pp. 1:1–1:26, Feb. 2009.

[137] X. Ji and H. Zha, “Sensor positioning in wireless ad-hoc sensor networks using multidimensional scaling,” in *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, vol. 4, pp. 2652–2661 vol.4, March 2004.

[138] Y. Shang, W. Ruml, and M. P. Fromherz, “Positioning using local maps,” *Ad Hoc Networks*, vol. 4, no. 2, pp. 240–253, 2006.

[139] S. Boyd, E. L. Ghaoui, E. Feron, and V. Balakrishnan, *Linear Matrix Inequalities in System and Control Theory*. SIAM, 1994.

[140] P. Biswas and Y. Ye, “Semidefinite programming for ad hoc wireless sensor network localization,” in *Information Processing in Sensor Networks, 2004. IPSN 2004. Third International Symposium on*, pp. 46–54, April 2004.

[141] A. M.-C. So and Y. Ye, “Theory of semidefinite programming for sensor network localization,” in *Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA ’05, (Philadelphia, PA, USA), pp. 405–414, Society for Industrial and Applied Mathematics, 2005.

[142] M. W. Carter, H. H. Jin, M. A. Saunders, and Y. Ye, “Spaseloc: An adaptive subproblem algorithm for scalable wireless sensor network localization,” *SIAM J. on Optimization*, vol. 17, pp. 1102–1128, Dec. 2006.

[143] S. Y. Wong, J. G. Lim, S. V. Rao, and W.-G. Seah, “Multihop localization with density and path length awareness in non-uniform wireless sensor networks,” in *Vehicular*

Technology Conference, 2005. VTC 2005-Spring. 2005 IEEE 61st, vol. 4, pp. 2551–2555 Vol. 4, May 2005.

[144] H. Lim and J. Hou, "Localization for anisotropic sensor networks," in *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, vol. 1, pp. 138–149 vol. 1, March 2005.

[145] Y. Wang, J. Gao, and J. S. Mitchell, "Boundary recognition in sensor networks by topological methods," in *Proceedings of the 12th Annual International Conference on Mobile Computing and Networking*, MobiCom '06, (New York, NY, USA), pp. 122–133, ACM, 2006.

[146] O. Saukh, R. Sauter, M. Gauger, P. Marron, and K. Rothermel, "On boundary recognition without location information in wireless sensor networks," in *Information Processing in Sensor Networks, 2008. IPSN '08. International Conference on*, pp. 207–218, April 2008.

[147] D. Dong, Y. Liu, and X. Liao, "Fine-grained boundary recognition in wireless ad hoc and sensor networks by topological methods," in *Proceedings of the Tenth ACM International Symposium on Mobile Ad Hoc Networking and Computing*, MobiHoc '09, (New York, NY, USA), pp. 135–144, ACM, 2009.

[148] T. He, J. A. Stankovic, R. Stoleru, Y. Gu, and Y. Wu, "Essentia: Architecting wireless sensor networks asymmetrically," in *INFOCOM 2008*, April 2008.

[149] A. Nasipuri and R. el Najjar, "Experimental evaluation of an angle based indoor localization system," in *Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks, 2006 4th International Symposium on*, pp. 1–9, April 2006.

[150] N. Jin, Z. Zhong, and T. He, "Optimizing event-driven localization," in *Mobile Ad-Hoc and Sensor Systems (MASS), 2013 IEEE 10th International Conference on*, pp. 55–63, Oct 2013.

[151] M. W. M. G. Dissanayake, P. Newman, S. Clark, H. Durrant-Whyte, and M. Csorba, “A solution to the simultaneous localization and map building (slam) problem,” *Robotics and Automation, IEEE Transactions on*, vol. 17, pp. 229–241, Jun 2001.

[152] R. Smith, M. Self, and P. Cheeseman, “Autonomous robot vehicles,” ch. Estimating Uncertain Spatial Relationships in Robotics, pp. 167–193, New York, NY, USA: Springer-Verlag New York, Inc., 1990.

[153] J. Leonard and H. Durrant-Whyte, “Simultaneous map building and localization for an autonomous mobile robot,” in *Intelligent Robots and Systems '91. 'Intelligence for Mechanical Systems, Proceedings IROS '91. IEEE/RSJ International Workshop on*, pp. 1442–1447 vol.3, Nov 1991.

[154] S. Thrun, W. Burgard, and D. Fox, “A probabilistic approach to concurrent mapping and localization for mobile robots,” *Auton. Robots*, vol. 5, pp. 253–271, July 1998.

[155] K. Murphy, “Bayesian map learning in dynamic environments,” in *In Neural Info. Proc. Systems (NIPS*, pp. 1015–1021, MIT Press, 1999.

[156] D. Crisan and A. Doucet, “A survey of convergence results on particle filtering methods for practitioners,” *Signal Processing, IEEE Transactions on*, vol. 50, pp. 736–746, Mar 2002.

[157] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, “Fastslam: A factored solution to the simultaneous localization and mapping problem,” in *Eighteenth National Conference on Artificial Intelligence*, (Menlo Park, CA, USA), pp. 593–598, American Association for Artificial Intelligence, 2002.

[158] M. Montemerlo, S. Thrun, D. Roller, and B. Wegbreit, “Fastslam 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges,” in *Proceedings of the 18th International Joint Conference on Artificial*

Intelligence, IJCAI'03, (San Francisco, CA, USA), pp. 1151–1156, Morgan Kaufmann Publishers Inc., 2003.

[159] T. S., D. Koller, Z. Ghahramani, H. Durrant-Whyte, and A. Ng, "Simultaneous mapping and localization with sparse extended information filters theory and initial results," technical report, Carnegie Mellon University, 2002.

[160] Y. Liu, D. Koller, A. Ng, Z. Ghahramani, and H. Durrant-Whyte, "Simultaneous localization and mapping with sparse extended information filters," *The International Journal of Robotics Research*, 2004.

[161] R. Eustice, H. Singh, and J. Leonard, "Exactly sparse delayed-state filters for view-based slam," *Robotics, IEEE Transactions on*, vol. 22, pp. 1100–1114, Dec 2006.

[162] R. Eustice, M. Walter, and J. Leonard, "Sparse extended information filters: Insights into sparsification," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3281–3288, 2005.

[163] R. Eustice, H. Singh, W. Hole, and W. Hole, "Visually navigating the rms titanic with slam information filters," in *in Proceedings of Robotics: Science and Systems*, pp. 57–64, 2005.

[164] J.-G. Kang, S.-Y. An, S. Kim, and S. young Oh, "A new approach to simultaneous localization and map building with learning: Neoslam (neuro-evolutionary optimizing)," in *Computational Intelligence in Robotics and Automation (CIRA), 2009 IEEE International Symposium on*, pp. 278–284, Dec 2009.

[165] S. Julier and J. Uhlmann, "A counter example to the theory of simultaneous localization and map building," in *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, vol. 4, pp. 4238–4243 vol.4, 2001.

[166] T. Bailey, J. Nieto, and E. Nebot, "Consistency of the fastslam algorithm," in *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pp. 424–429, May 2006.

[167] K. Konolige, "Large-scale map-making," in *Proceedings of the 19th National Conference on Artifical Intelligence*, AAAI'04, pp. 457–463, AAAI Press, 2004.

[168] F. Dellaert and M. Kaess, "Square root sam: Simultaneous localization and mapping via square root information smoothing," *Int. J. Rob. Res.*, vol. 25, pp. 1181–1203, Dec. 2006.

[169] J. J. Leonard and H. J. S. Feder, "A computationally efficient method for large-scale concurrent mapping and localization," in *The Ninth International Symposium (ISRR'99)*, pp. 169–176, 2000.

[170] J. Guivant and E. Nebot, "Optimization of the simultaneous localization and map-building algorithm for real-time implementation," *Robotics and Automation, IEEE Transactions on*, vol. 17, pp. 242–257, Jun 2001.

[171] M. Bosse, P. Newman, J. Leonard, M. Soika, W. Feiten, and S. Teller, "An atlas framework for scalable mapping," in *Robotics and Automation, 2003. Proceedings. ICRA '03. IEEE International Conference on*, vol. 2, pp. 1899–1906 vol.2, Sept 2003.

[172] J. Leonard and P. Newman, "Consistent, convergent, and constant-time slam," in *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, I-JCAI'03, (San Francisco, CA, USA), pp. 1143–1150, Morgan Kaufmann Publishers Inc., 2003.

[173] M. Bosse, P. Newman, J. Leonard, and S. Teller, "Slam in large-scale cyclic environments using the atlas framework," *International Journal of Robotics Research*, pp. 1113–1139, 2004.

[174] Y. Bar-Shalom and T. Fortmann, “Tracking and data association,” *New York: Academic Press*, 1988.

[175] J.-S. Gutmann and K. Konolige, “Incremental mapping of large cyclic environments,” in *Computational Intelligence in Robotics and Automation, 1999. CIRA ’99. Proceedings. 1999 IEEE International Symposium on*, pp. 318–325, 1999.

[176] J. Neira and J. Tardos, “Data association in stochastic mapping using the joint compatibility test,” *Robotics and Automation, IEEE Transactions on*, vol. 17, pp. 890–897, Dec 2001.

[177] J. Folkesson and H. Christensen, “Graphical slam - a self-correcting map,” in *Robotics and Automation, 2004. Proceedings. ICRA ’04. 2004 IEEE International Conference on*, vol. 1, pp. 383–390 Vol.1, April 2004.

[178] S. Argamon-Engelson, “Using image signatures for place recognition,” *Pattern Recogn. Lett.*, vol. 19, pp. 941–951, Aug. 1998.

[179] P. Newman, D. Cole, and K. Ho, “Outdoor slam using visual appearance and laser ranging,” in *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pp. 1180–1187, May 2006.

[180] R. Kurazume, S. Hirose, S. Nagata, and N. Sashida, “Study on cooperative positioning system (basic principle and measurement experiment),” in *Robotics and Automation, 1996. Proceedings., 1996 IEEE International Conference on*, vol. 2, pp. 1421–1426 vol.2, Apr 1996.

[181] R. Kurazume, S. Nagata, and S. Hirose, “Cooperative positioning with multiple robots,” in *Robotics and Automation, 1994. Proceedings., 1994 IEEE International Conference on*, pp. 1250–1257 vol.2, May 1994.

[182] J. M. Walls and R. M. Eustice, “An exact decentralized cooperative navigation algorithm for acoustically networked underwater vehicles with robustness to faulty com-

munication: Theory and experiment," in *Proceedings of the Robotics: Science & Systems Conference*, (Berlin, Germany), June 2013.

[183] J.-O. Nilsson, D. Zachariah, I. Skog, and P. Händel, "Cooperative localization by dual foot-mounted inertial sensors and inter-agent ranging," *EURASIP Journal on Advances in Signal Processing*, vol. 1, Aug. 2013.

[184] S. Roumeliotis and G. A. Bekey, "Distributed multirobot localization," *Robotics and Automation, IEEE Transactions on*, vol. 18, pp. 781–795, Oct 2002.

[185] H. Mu, T. Bailey, P. Thompson, and H. Durrant-Whyte, "Decentralised solutions to the cooperative multi-platform navigation problem," *Aerospace and Electronic Systems, IEEE Transactions on*, vol. 47, pp. 1433–1449, April 2011.

[186] K. Leung, T. Barfoot, and H. Liu, "Decentralized localization of sparsely-communicating robot networks: A centralized-equivalent approach," *Robotics, IEEE Transactions on*, vol. 26, pp. 62–77, Feb 2010.

[187] D. Fox, W. Burgard, H. Kruppa, and S. Thrun, "A probabilistic approach to collaborative multi-robot localization," *Auton. Robots*, vol. 8, pp. 325–344, June 2000.

[188] P. Zhang and M. Martonosi, "Locale: Collaborative localization estimation for sparse mobile sensor networks," in *International Conference on Information Processing in Sensor Networks (IPSN '08)*, pp. 195 –206, april 2008.

[189] P. Barooah, W. J. Russell, and J. Hespanha, "Approximate distributed kalman filtering for cooperative multi-agent localization," in *Intl. conference in Distributed Computing in Sensor Systems*, DCOSS'10, 2010.

[190] J. Knuth and P. Barooah, "Distributed collaborative 3d pose estimation of robots from heterogeneous relative measurements: an optimization on manifold approach," *Robotica*, pp. 1–29, April 2014.

[191] L. Yang, Y. Chen, X.-Y. Li, C. Xiao, M. Li, and Y. Liu, “Tagoram: Real-time tracking of mobile rfid tags to high precision using cots devices,” in *Proceedings of the 20th Annual International Conference on Mobile Computing and Networking*, MobiCom ’14, (New York, NY, USA), pp. 237–248, ACM, 2014.

[192] E. Foxlin, “Pedestrian tracking with shoe-mounted inertial sensors,” *Computer Graphics and Applications, IEEE*, vol. 25, pp. 38–46, Nov 2005.

[193] I. Skog, P. Handel, J. Nilsson, and J. Rantakokko, “Zero-velocity detection–an algorithm evaluation,” *IEEE Transactions on Biomedical Engineering*, vol. 57, pp. 2657 –2666, nov. 2010.

[194] I. Skog, J.-O. Nilsson, and P. Handel, “Evaluation of zero-velocity detectors for foot-mounted inertial navigation systems,” in *International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, pp. 1 –6, sept. 2010.

[195] A. Jiménez, F. Seco, J. Prieto, and J. Guevara, “Indoor pedestrian navigation using an ins/ekf framework for yaw drift reduction and a foot-mounted imu,” in *7th Workshop on Positioning Navigation and Communication (WPNC), 2010*, pp. 135 –143, march 2010.

[196] A. Jimenez, F. Seco, F. Zampella, J. Prieto, and J. Guevara, “Improved heuristic drift elimination (ihde) for pedestrian navigation in complex buildings,” in *International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, pp. 1 –8, Sept. 2011.

[197] A. Kelly, M. Laverne, M. George, D. Lord, and T. Mukhergee, “Personal navigation system based on dual shoe-mounted imus and intershoe ranging,” in *Precision Personnel Locator Workshop*, Aug. 2011.

[198] L. Fang, P. Antsaklis, L. Montestruque, M. McMickell, M. Lemmon, Y. Sun, H. Fang, I. Koutroulis, M. Haenggi, M. Xie, and X. Xie, “Design of a wireless assisted pedes-

trian dead reckoning system - the navmote experience," *Instrumentation and Measurement, IEEE Transactions on*, vol. 54, pp. 2342–2358, Dec 2005.

[199] P. Goyal, V. Ribeiro, H. Saran, and A. Kumar, "Strap-down pedestrian dead-reckoning system," in *Indoor Positioning and Indoor Navigation (IPIN), 2011 International Conference on*, pp. 1–7, Sept 2011.

[200] H. Ying, C. Silex, A. Schnitzer, S. Leonhardt, and M. Schiek, "Automatic step detection in the accelerometer signal," in *4th International Workshop on Wearable and Implantable Body Sensor Networks (BSN 2007)*, pp. 80–85, March 2007.

[201] H. WeinBerg, "An-602: Using the adxl202 in pedometer and personal navigation applications," analog devices technical report 2002, Mar. 2002.

[202] S. Yang and Q. Li, "Ambulatory walking speed estimation under different step lengths and frequencies," in *Advanced Intelligent Mechatronics (AIM), 2010 IEEE/ASME International Conference on*, pp. 658–663, July 2010.

[203] V. Radu and M. Marina, "Himloc: Indoor smartphone localization via activity aware pedestrian dead reckoning with selective crowdsourced wifi fingerprinting," in *Indoor Positioning and Indoor Navigation (IPIN), 2013 International Conference on*, pp. 1–10, Oct 2013.

[204] D. Gusenbauer, C. Isert, and J. Krösche, "Self-contained indoor positioning on off-the-shelf mobile devices," in *Indoor Positioning and Indoor Navigation (IPIN), 2010 International Conference on*, pp. 1–9, Sept 2010.

[205] H. Ye, T. Gu, X. Zhu, J. Xu, X. Tao, J. Lu, and N. Jin, "Ftrack: Infrastructure-free floor localization via mobile phone sensing," in *Pervasive Computing and Communications (PerCom), 2012 IEEE International Conference on*, pp. 2–10, March 2012.

[206] B. Paramvir and V. Padmanabhan, "Radar: An in-building rfbased user location and tracking system," in *INFOCOM 2000*, 2000.

[207] W. Chai, C. Chen, E. Edwan, J. Zhang, and O. Loffeld, “Ins/wi-fi based indoor navigation using adaptive kalman filtering and vehicle constraints,” in *Positioning Navigation and Communication (WPNC), 2012 9th Workshop on*, pp. 36–41, March 2012.

[208] J. Ledlie, J. geun Park, D. Curtis, A. Cavalcante, L. Camara, A. Costa, and R. Vieira, “Mole: A scalable, user-generated wifi positioning engine,” in *Indoor Positioning and Indoor Navigation (IPIN), 2011 International Conference on*, pp. 1–10, Sept 2011.

[209] E. Bhasker, S. Brown, and W. G. Griswold, “Employing user feedback for fast, accurate, low-maintenance geolocationing,” in *Pervasive Computing and Communications, 2004. PerCom 2004. Proceedings of the Second IEEE Annual Conference on*, pp. 111–120, March 2004.

[210] P. Bolliger, “Redpin - adaptive, zero-configuration indoor localization through user collaboration,” in *Proceedings of the First ACM International Workshop on Mobile Entity Localization and Tracking in GPS-less Environments*, MELT ’08, (New York, NY, USA), pp. 55–60, ACM, 2008.

[211] J.-g. Park, B. Charrow, D. Curtis, J. Battat, E. Minkov, J. Hicks, S. Teller, and J. Ledlie, “Growing an organic indoor location system,” in *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services*, MobiSys ’10, (New York, NY, USA), pp. 271–284, ACM, 2010.

[212] A. Barry, B. Fisher, and M. L. Chang, “A long-duration study of user-trained 802.11 localization,” in *Proceedings of the 2Nd International Conference on Mobile Entity Localization and Tracking in GPS-less Environments*, MELT’09, (Berlin, Heidelberg), pp. 197–212, Springer-Verlag, 2009.

[213] A. Günther and C. Hoene, “Measuring round trip times to determine the distance between wlan nodes,” in *Proceedings of the 4th IFIP-TC6 International Conference on Networking Technologies, Services, and Protocols; Performance of Computer*

and Communication Networks; Mobile and Wireless Communication Systems, NETWORKING'05, (Berlin, Heidelberg), pp. 768–779, Springer-Verlag, 2005.

[214] M. Ciurana, F. Barcelo-Arroyo, and F. Izquierdo, "A ranging system with ieee 802.11 data frames," in *Radio and Wireless Symposium, 2007 IEEE*, pp. 133–136, Jan 2007.

[215] M. Ciurana, D. López, and F. Barceló-Arroyo, "Softoa: Software ranging for toa-based positioning of wlan terminals," in *Proceedings of the 4th International Symposium on Location and Context Awareness*, LoCA '09, (Berlin, Heidelberg), pp. 207–221, Springer-Verlag, 2009.

[216] I. Casacuberta and A. Ramirez, "Time-of-flight positioning using the existing wireless local area network infrastructure," in *Indoor Positioning and Indoor Navigation (IPIN), 2012 International Conference on*, pp. 1–8, Nov 2012.

[217] L. Schauer, F. Dorfmeister, and M. Maier, "Potentials and limitations of wifi-positioning using time-of-flight," in *Indoor Positioning and Indoor Navigation (IPIN), 2013 International Conference on*, pp. 1–9, Oct 2013.

[218] C. Pereira, L. Guenda, and N. B. Carvalho, "A smart-phone indoor outdoor localization system," in *Indoor Positioning and Indoor Navigation (IPIN), 2011 International Conference on*, pp. 1–10, Sept 2011.

[219] J. Paek, K.-H. Kim, J. P. Singh, and R. Govindan, "Energy-efficient positioning for smartphones using cell-id sequence matching," in *Proceedings of the 9th International Conference on Mobile Systems, Applications, and Services*, MobiSys '11, (New York, NY, USA), pp. 293–306, ACM, 2011.

[220] L. Shangguan, Z. Li, Z. Yang, M. Li, and Y. Liu, "Otrack: Order tracking for luggage in mobile rfid systems," in *INFOCOM, 2013 Proceedings IEEE*, pp. 3066–3074, April 2013.

[221] C. Hekimian-Williams, B. Grant, X. Liu, Z. Zhang, and P. Kumar, "Accurate localization of rfid tags using phase difference," in *RFID, 2010 IEEE International Conference on*, pp. 89–96, April 2010.

[222] R. Miesen, F. Kirsch, and M. Vossiek, "Holographic localization of passive uhf rfid transponders," in *RFID (RFID), 2011 IEEE International Conference on*, pp. 32–37, April 2011.

[223] A. Parr, R. Miesen, and M. Vossiek, "Inverse sar approach for localization of moving rfid tags," in *RFID (RFID), 2013 IEEE International Conference on*, pp. 104–109, April 2013.

[224] W. Zhu, J. Cao, Y. Xu, L. Yang, and J. Kong, "Fault-tolerant rfid reader localization based on passive rfid tags," in *INFOCOM, 2012 Proceedings IEEE*, pp. 2183–2191, March 2012.

[225] Y. Liu, Y. Zhao, L. Chen, J. Pei, and J. Han, "Mining frequent trajectory patterns for activity monitoring using radio frequency tag arrays," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 23, pp. 2138–2149, Nov 2012.

[226] W. Chai, J. Zhou, C. Chen, H. Nies, and O. Loffeld, "Continuous indoor localization and navigation based on low-cost ins/wi-fi integration," in *Indoor Positioning and Indoor Navigation (IPIN), 2011 International Conference on*, pp. 1–10, Sept 2011.

[227] C. Ascher, S. Werling, G. Trommer, L. Zwirello, C. Hansmann, and T. Zwick, "Radio-asissted inertial navigation system by tightly coupled sensor data fusion: Experimental results," in *Indoor Positioning and Indoor Navigation (IPIN), 2012 International Conference on*, pp. 1–7, Nov 2012.

[228] N. Nakajima, T. Hazukawa, K. Nishida, and K. Hattori, "Accurate spot positioning technology for indoor hybrid navigation," in *Indoor Positioning and Indoor Navigation (IPIN), 2011 International Conference on*, Sept 2011.

[229] M. Langer, S. Kiesel, C. Ascher, and G. Trommer, "Deeply coupled gps/ins integration in pedestrian navigation systems in weak signal conditions," in *Indoor Positioning and Indoor Navigation (IPIN), 2012 International Conference on*, pp. 1–7, Nov 2012.

[230] F. Zampella, A. Jimenez R, and F. Seco, "Robust indoor positioning fusing pdr and rf technologies: The rfid and uwb case," in *Indoor Positioning and Indoor Navigation (IPIN), 2013 International Conference on*, pp. 1–10, Oct 2013.

[231] D. Fox, "Kld-sampling: Adaptive particle filters," in *NIPS*, 2002.

[232] S. Beauregard, Widyawan, and M. Klepal, "Indoor pdr performance enhancement using minimal map information and particle filters," in *Position, Location and Navigation Symposium, 2008 IEEE/ION*, pp. 141–147, May 2008.

[233] R. K. Jeremy, R. M. Karp, J. Elson, C. H. Papadimitriou, and S. Shenker, "Global synchronization in sensornets," in *LATIN*, pp. 609–624, 2004.

[234] A. Frommery, H. Schwandtz, and D. B. Szyldx, "Asynchronous weighted additive schwarz methods," *Electronic Transactions on Numerical Analysis*, vol. 5, pp. 48–61, June 1997.

[235] R. B. Ash and C. A. Doléans-Dade, *Probability and Measure Theory (2nd ed.).* San Diego: Harcourt/Academic Press, 2000.

[236] Y. Kwon, K. Mechitov, S. Sundresh, W. Kim, and G. Agha, "Resilient localization for sensor networks in outdoor environments," in *ICDCS*, 2005.

[237] S. Roumeliotis and G. Bekey, "Collective localization: a distributed kalman filter approach to localization of groups of mobile robots," in *ICRA '00*, vol. 3, pp. 2958 –2965 vol.3, 2000.

[238] Z. Xie, M. Hong, H. Liu, J. Li, K. Zhu, and J. Stankovic, "Quantitative uncertainty-based incremental localization and anchor selection in wireless sensor networks," in *MSWiM '11*, (NY, USA), ACM, 2011.

[239] J. Yuan, Y. Zheng, X. Xie, and G. Sun, “Driving with knowledge from the physical world,” in *KDD ’11*, (NY, USA), ACM, 2011.

[240] T. S. Rappaport, *Wireless Communications: Principles and Practice*. Prentice-Hall Inc., 1996.

[241] H. Hashemi, “The indoor radio propagation channel,” *Proceedings of the IEEE*, vol. 81, pp. 943–968, Jul 1993.

[242] G. Durgin, T. S. Rappaport, and H. Xu, “Measurements and models for radio path loss and penetration loss in and around homes and trees at 5.85 ghz,” *IEEE Transactions on Communications*, vol. 46, pp. 1484–1496, Nov 1998.

[243] J. O. Nilsson, I. Skog, P. Händel, and K. V. S. Hari, “Foot-mounted ins for everybody - an open-source embedded implementation,” in *Position Location and Navigation Symposium (PLANS), 2012 IEEE/ION*, pp. 140–145, April 2012.

[244] J. O. Nilsson, I. Skog, and P. Händel, “A note on the limitations of zupts and the implications on sensor error modeling,” in *2012 International Conference on Indoor Positioning and Indoor Navigation*, November 2012.

[245] “Minimu-9 v2. http://www.pololu.com/catalog/product/1268.”

[246] “Gyroscope l3gd20. https://www.pololu.com/file/download/l3gd20.pdf?file_id=0j563.”

[247] “Accelerometer and magnetometer lsm303dlhc. https://www.pololu.com/file/download/lsm303dlhc.pdf?file_id=0j564.”

[248] M. Looney, “A simple calibration for mems gyroscopes,” *EDN*, July 2010.

[249] A. Eiben, P.-E. Raué, and Z. Ruttkay, “Genetic algorithms with multi-parent recombination,” in *PPSN III: Proceedings of the International Conference on Evolutionary Computation*, 1994.

[250] J. A. Nelder and R. Mead, “A simplex method for function minimization,” *Computer Journal 7*, p. 308C313, 1965.

[251] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, “Optimization by simulated annealing,” *SCIENCE*, vol. 220, no. 4598, pp. 671–680, 1983.

[252] “Arduino uno. https://www.arduino.cc/en/main/arduinoboardarduno.”

[253] S. Wan and E. Foxlin, “Improved pedestrian navigation based on drift-reduced mems imu chip,” in *Proceedings of the 2010 International Technical Meeting of The Institute of Navigation*, 2010.