Clock Synchronization and Localization for Wireless Sensor Network

Cheng-Yu Han

université PARIS-SACLAY

NNT : 2018SACLS453

UNIVERSITÉ PARIS SUD
Comprendre le monde, construire l'avenir

Clock Synchronization and Localization for Wireless Sensor Network

Thèse de doctorat de l'Université Paris-Saclay
préparée a l'Universite Paris-Sud

École doctorale n°580 Sciences et technologies de l'information et de la communication (STIC)
Spécialité de doctorat: Robotique

Thèse présentée et soutenue à Gif-sur-Yvette, le 12 Novembre 2018, par

Cheng-Yu Han

Composition du Jury :

Jaulin LUC Professeur, Université de Bretagne Occidentale, Lab-STICC	Président
Hichem SNOUSSI Professeur, Université de technologie de Troyes, LM2S	Rapporteur
Marcus SHAWKY Professeur, Université de Technologie de Compiègne, Heudiasyc	Rapporteur
Farah CHEHADE Maître de Conférences, Université de technologie de Troyes, LM2S	Examinateur
Alain LAMBERT Maître de Conférences, Université Paris-Sud, LRI	Directeur de thèse
Thomas NOWAK Maître de Conférences, Université Paris-Sud, LRI	Co-Directeur de thèse
Michel KIEFFER Professeur, Centrale Supélec, L2S	Co-Directeur de thèse

Thèse de doctorat

Contents

Acknowledgement

Most of all, I would like to thank Prof. Alain Lambert, Prof. Thomas Nowak, and Prof. Michel Kieffer who supported my work and helped me get results of better quality. I also like to thank the reviewers and the juries. With their help, I have better knowledge of the domain of wireless sensor network and related algorithms. They are the best models for my future career. I would like to thank my girl friend Dr. Min-I Lee. I could not finish this long journey without hers accompany. Studying in Paris-Sud University with her is the best decision in my life. I would like to thank my friends for the support and being my travel companions. Last but not the least, I would like to thank my family: my parents, my brother, aunts and uncles for supporting me spiritually throughout writing this thesis and my my life in general.

Chapter 1

Introduction

Wireless sensor networks (WSN) consist of many individual low-cost and low-power sensor nodes, which achieve goals such as environmental monitoring by running micro electrical chips and detection sensors. Due to wide applications in environmental monitoring, health care, and the IoT embed WSN, WSN has caught researcher's attention and been developed rapidly [Han et al., 2015].

In many applications, sensors in WSNs observe some physical phenomenon and gather the observations to an evaluate estimation of the original physical phenomenon. This process is known as data fusion [Huang et al., 2003]. In order to perform this task, the ability of sensors to estimate their physical location in some well chosen frame and the knowledge of time for WSN to work under a universal synchronized clock is essential. Synchronization among sensors is usually more important than absolute synchronization with some reference high-accuracy clock. For instance, real-time controller systems usually assume that events occur at predetermined time instants, and that require sensors sampled at the assumed rate together, otherwise the calculations might be wrong. This necessity is reflected by the development of a significant number of algorithms for localization and time synchronization for sensor networks in the recent past.

For example, many sensors can only detect the proximity of an observed object. Higher-level information, such as speed, size, or shape of an object can then only be

obtained by correlating data from multiple sensor nodes whose locations are known. The shape or the velocity of a mobile object, for example, can be estimated by the ratio of the spatial and temporal distances between two consecutive object sightings by different sensor nodes. For example, a sensor node can detect measurements of a widespread object, and the size and shape can be approximated by the information of the sensor nodes.

In WSN, many applications are based on the ability of sensors nodes to estimate the physical location and to synchronize the time with each other. Thus, a significant number of algorithms for localization and clock synchronization for WSN were studied in the recent years. However, due to the complexity and dynamicity of the WSN, robust algorithms that can tolerate inaccurate measurements during the communication and computation of WSN still have a tremendous demand.

For the synchronization problem, sensor nodes synchronize the clock value to perform data fusion algorithms or speed computation [Sundararaman et al., 2005, Solis et al., 2006, Su and Akyildiz, 2005]. The local physical clock is driven so that the clock values are synchronized among sensors. A straightforward solution is to use a source node to provide a coordinate universal time. This solution is not suitable for WSN because the location of the sensor nodes is uncertain, so a source node may not be able to send the message to all the nodes in the WSN and the propagation delay is uncertain. Moreover, this solution is not robust, because a failure of the source node can affect all the sensors in the network.

Considering the dynamicity, a distributed algorithm is proposed, which extend the previous work from Függer et al. [2015b] and reduce the clock skew. In the new pulse-coupled synchronization, time-wheel algorithm is introduced for sensors to exchange the round number and averaging the pulses of the same round. Moreover drift compensation is applied to reduce the clock skew. The convergence analysis shows that the drift compensation reduces the clock skew and improves the convergence rate. When considering a highly dynamic WSN, the algorithm converges the software clock faster then in previous works. The proposed algorithm is successfully applied to a vehicular network.

For the localization problem, sensors can try locating each other possibly with the help of some anchor sensors in order to obtain a position estimation. A simple method is to put a GPS receiver on each sensor [Yick et al., 2008]. With GPS receivers, sensors can receive information from satellites and estimate the geographical position themselves. However, GPS receivers drain the battery very quickly, so usually, only a small portion of sensors in WSN have GPS receivers. For some applications, GPS signal may not be received by the sensors (e.g., indoor, underwater scenarios).

For the second part of the thesis, we estimate the location of each sensor using LSCR, which produces a non-asymptotic confidence region. The benefit of applying LSCR to localization problem is that the noise assumption on the noise of RSS signal is mild, so the confidence region can be estimated accurately. We analyze different correlation functions of LSCR and found that only some correlation functions can produce a well-shaped confidence region. We also implement alternative localization algorithms, involving, e.g., a maximum likelihood estimator (MLE), semi-definite programming (SDP), bounded error estimator (BE), robust bounded error estimator (RBE). We compared the alternative methods and found that LSCR provides the best localization results: its average error is smaller than other alternative methods.

In this thesis, we divide our work into two parts for viewers to understand the backgrounds and methodologies for both clock synchronization and localization. In the first part, the clock synchronization is discussed, and in the second part, the wireless sensor localization is discussed.

For clock synchronization part, Chapter 3 introduces a formal model of the wireless sensor network and specify the clock synchronization problem. In Chapter 4, the methodology used in this work is shown with the analysis of the clock skew and convergence rate with mathematical derivation, which is followed by some experimental results in Chapter 5, and finally, the conclusions of the survey and the direction of future work are given in Chapter 6.

For the localization part, in Chapter 8, tools from Interval analysis is described, which is used by some of the localization methods. With the knowledge of Interval analysis, Chapter 9 introduces RSS localization model and specify several localiza-

tion approaches. In Chapter 10, the methodology used in this work is shown and compared with other localization approaches. The different correlation functions of LSCR are also simulated and analyzed. Finally, the conclusions of the survey is given in Chapter 12.

Part I

Clock synchronization

Chapter 2

Introduction

This chapter presents the importance of the clock synchronization problem. Previous works and proposed techniques are explained briefly. The chapter ends with a description of the structure and contents of clock synchronization.

2.1 Motivation

Clock synchronization is a critical issue in sensor networks [Ganeriwal et al., 2003, Munir et al., 2007, Syed et al., 2006, Sun et al., 2005]. In particular, it is a cornerstone of data fusion, where algorithms work in a better way when data are precisely synchronized rather than only time stamped. Distributed data fusion of non-synchronized data necessitates taking into account the shift between the time of the measurement. Such shifts increase the data fusion imprecision results [Sundararaman et al., 2005, Solis et al., 2006, Su and Akyildiz, 2005]. A quite large number of clock synchronization protocols, mostly for wired networks, have been developed over the past few decades. However, most of the protocols are unsuitable for wireless sensor networks due to the dynamicity, low bandwidth, and often low-powered computing nodes [Elson and Römer, 2003, Ganesan et al., 2004].

On the other hand, the hardware of the sensors are usually very restricted (lim-

ited storage, slow computation) since the size of the sensors is small [Bertrand, 2011, Sundararaman et al., 2005]. Energy consumption of wireless sensor networks can be high when service constraints are high [Akyildiz et al., 2002, Sundararaman et al., 2005]. The energy is usually provided by the battery with limited power for sensors, so synchronization must be achieved while preserving energy [Ye et al., 2002]. For those sensors, the limit on bandwidth directly influences message exchanges among sensors [Al-Karaki and Kamal, 2004]. During the message delivery, the flux of intensity of electromagnetic radiation is reduced through the medium. Therefore, the message may not be delivered from the sender to the receiver if their distance is long [Swain and Hansdah, 2015]. In addition, unlike wired networks, wireless sensor networks are dynamic. Sensors may move and communication links may change. In view of this, traditional clock synchronization methods for wired networks [Mills, 1996, Elson et al., 2002] are often not applicable for wireless sensor networks.

2.2 Background

Most clock synchronization algorithms for wireless sensor networks are packet-based methods, where sensors exchange digital data to agree on a common time base. Centralized approaches [Mills, 1991, Cristian, 1989, Kopetz and Ochsenreiter, 1987] distribute the clock of a master node to other nodes and are not robust to any failure of the master node. In the light of this, an average consensus-based protocol is proposed in Schenato and Fiorentin [2011], where sensors exchange clock readings and update its clock drift and offset by averaging the value of its neighbors. A maximum consensus-based protocol based on Schenato and Fiorentin [2011] is proposed in He et al. [2014] to improve the convergence rate. A second-order consensus method is proposed in Carli and Zampieri [2014] to reduce the clock skew further.

For all the above packet synchronization [Schenato and Fiorentin, 2011, He et al., 2014, Carli and Zampieri, 2014], nodes exchange data information (global time, frequency, or system parameters) between its neighboring nodes and agree on some parameters of the model to have a common notion of time. In large sensor networks,

message delivery might be unreliable due to both dynamicity and sparsity of these networks [Wu et al., 2011]. Most wireless sensor networks usually have stationary sensor nodes. The network topology is usually fixed, so sensors can transmit packets without failure. However, for a sensor network, all the sensors move rapidly, which makes it difficult to apply a common clock synchronization protocols such as Schenato and Gamba [2007], Elson et al. [2002]. Also, during intensive transmission, sensors suffer from the packet collision and delay [Pešović et al., 2009]. When a sensor tries to send packets, it detects and waits until the medium is not occupied by the other sensors, so the information transmission might be postponed. Moreover, if sensors are far away, the sender cannot detect whether the medium is occupied near the receiver, so the packet collision might happen when two senders transmit packets to the receiver simultaneously.

An alternative to the packet-based synchronization approaches, pulse-coupled synchronization, is proposed in Li and Rus [2006], where nodes only exchange pulses instead of packets. The convergence properties of Li and Rus [2006] is studied in Simeone and Spagnolini [2007] through linear algebra with different communication graphs. A different topology (scenario) of the network is investigated, and a practical implementation is discussed. Also, the dynamic and bi-directional properties of pulse-coupled synchronization are analyzed in Függer et al. [2015b]. The algorithm solves clock synchronization as long as the topology of the dynamic network contains a directed spanning tree. Idle listening is introduced in Wang et al. [2012] to reduce total energy consumption.

Pulse-coupled synchronization [Li and Rus, 2006, Simeone and Spagnolini, 2007, Függer et al., 2015b, Wang et al., 2012] can be viewed as the first-order phase locked loops [Simeone et al., 2008], which suffer from carrier frequency offset between sensors resulting in a constant phase lag. Moreover, pulse frequency synchronization is studied [Varanese et al., 2011].

2.3 Objectives

The primary objective of the part I is to improve the limitations of the current state-of-the-art techniques.

The specific objectives of this thesis are as follows.

- To implement a clock synchronization protocol is able to work well with any number of nodes in a dynamic network.
- To implement a decentralized clock synchronization protocol, which tolerates communication loss of the sensors in the network.
- To develop a new communication method to improve precision (clock skew) of the clock synchronization.
- To relax the assumption of pulse-coupled clock synchronization with more robust information exchange protocol.
- To implement an energy efficient clock synchronization protocol that only performs simple calculation instead of performing energy-consuming optimization.

This work extended and improved the algorithm of the previous time synchronization protocols [Függer et al., 2015b] based on theory and experiment results in clock skew and some other aspects.

2.4 Outline

Chapter 3 introduces a formal model of the wireless sensor network and specify the clock synchronization problem. In chapter 4, the methodology used in this work is shown with the analysis of the clock skew and convergence rate with mathematical derivation, which is followed by some experimental results in Chapter 5, and finally, the conclusions of the survey and the direction of future work are given in Chapter 6.

Chapter 3

Formal Model and Problem Formulation

3.1 Introduction

This chapter describes the formal model of the wireless sensor network and specifies the problem of clock synchronization. First, the local physical clock will be introduced. Second, the formal model of the wireless sensor network is defined. In the end, an error function is presented to evaluate the quality of the clock synchronization algorithm. Also, the existing state of the art techniques which deal with this problem are described.

3.2 Formal Model and Problem Specification

The goal of clock synchronization is to synchronize the time of all sensors because physical clocks of each sensor drift even if they have the same frequency, and these frequencies might even change over time. Figure 3.1 shows the physical clocks of sensor i and j. The frequency of the physical clocks for i and j are different due to clock drift or hardware setup, so the time period of pulses are different as well.

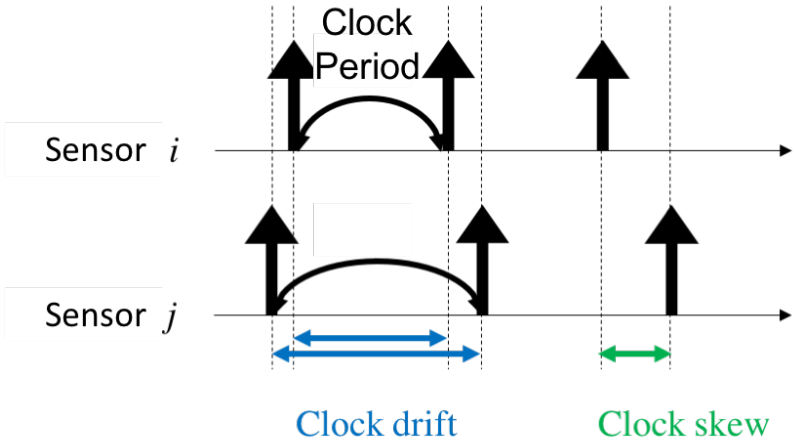


Figure 3.1: The physical clocks of the sensor i and j without any corrections. The frequency of the physical clock for both i and j can be different due to the clock drift or hardware setup, so the time period of pulses are different. The time difference of the ticks between sensors is evaluated using clock drift and clock skew. The clock drift is the difference of clock periods between sensors and the clock skew is the largest difference of time between pulses of sensors.

The quality of the clocks is evaluated using clock drift and clock skew. The clock drift is the difference in clock periods between sensors. The clock skew is the largest difference of time between pulses of sensors. The goal is to reduce the clock skew and the clock drift as much as possible.

3.2.1 Formal Model of Wireless Sensor Network

A finite set $\mathcal{N} = \{1, 2, \ldots, n\}$ of sensors communicating by message passing is considered. They are endowed with imperfect physical local clocks evolving in the time-base $\mathcal{T} = [0, \infty)$ of nonnegative reals.

The sensor network is modeled by a set of graph $G_t = (\mathcal{N}, \mathcal{V}_t), t \in \mathcal{T}$, where $\mathcal{N}$ is the set of sensors and $\mathcal{V}_t$ is a set of directed edges. The relation $(i, j) \in \mathcal{V}_t$ means that sensor i can send a message to sensor j at time t. Mobility is modeled by having a (possibly different) communication graph of different time instants.

A *local algorithm* for a sensor comprises a set of *local states* $\mathcal{S}$, a set of *events* $\mathcal{E}$, and its *state transition* function $f(s, e)$. The local state is the internal variables for each sensor. The local state is initialized at $s_0 \in \mathcal{S}$. Local state changes only when an event $e \in \mathcal{E}$ occurs.

An *execution* of an algorithm in scenarios G_t is a sequence of events $(e_q)_{q\geq 0}$, which are triggered by sensors $i_q \in \mathcal{N}$ with its present states $s_q \in \mathcal{S}$ at a specific time $t_q \in \mathcal{T}$. The value of q is a sequence number of the event in the system. Thus, the first event is e_0 and the second event is e_1 and so on. The sequence of events is sorted by t_q. At the beginning of the execution, every sensor triggers its first event with its initial state $s_0^{(q)}$ at time $t_q = 0$.

There are three types of event $e^{\mathrm{b}}, e^{\mathrm{m}}, e^{\mathrm{t}} \subset \mathcal{E}$. The event e^{b} represents the beginning of the local algorithm. The event e^{m} indicates that a sensor receives a message. The event e^{t} is triggered when the timer in a sensor reaches a certain value (sensors can schedule this event when executing state transition functions). A sensor i determines its next local state s' using the state transition function $f(s, e)$. A sensor cannot access the global time, only the physical local clocks of the sensor are obtained.

We assume all the sensors have the following features: A sensor i can send a message to sensor j at time t if and only if $(i, j) \in \mathcal{V}_t$. A sensor i produces and sends a message using its *sending function*: $\mathcal{S} \to \mathcal{M}$, where the $\mathcal{S}$ is the local state, and $\mathcal{M}$ is the message. Sending function maps the state s_k to message content. A sensor j receives messages through a *receive function* when a message event e^{m} occurs. The time difference between sending and receiving a message is assumed to be zero. The assumption is reasonable if the message contains only a few bits. Notice that, the noise caused the transmission delay leads to larger clock skew in the wireless sensor network. However, with the pulse synchronization, sensors only exchange few bits, so the transmission delay is neglectable.

A sensor i is equipped with a local physical clock with the clock period $\mu_i(c)$, where μ_i is a function, which subsumes manufacturing defects, hardware differences, and different temperatures for quartz oscillators. The variable c is a *local physical*

counter, which is initiated at 0 and increases over time whenever the local physical clock ticks. For simplicity of notation, assume that the nominal real time duration of a local physical clock tick is equal to 1. With c, sensors read time t_i as

$$t_i(c) = \sum_{m=0}^{c} \mu_i(m). \tag{3.1}$$

Assuming that $(1 - \varrho)\overline{\mu} \leq \mu_i(c) \leq (1 + \varrho)\overline{\mu}$, where ϱ represents the bound of the clock period and $\overline{\mu}$ is the manufacture specification of the clock period, one gets

$$(1 - \varrho)\overline{\mu}c \leq t_i(c) \leq (1 + \varrho)\overline{\mu}c, \tag{3.2}$$

which indicates that the uncertainty becomes larger through time when c and t_i increase.

The main problem for clock synchronization to solve is that c_i for each sensor does not increase at the same pace. Therefore, in this framework, a sensor i is able to calculate and exchange the round number k_i. The round number k_i for sensor i can be viewed as a software clock for each sensor. The value of k_i starts from zero and increases periodically for each sensor i. With state transition functions, a sensors are able to broadcast messages and try to increase k_i in the same pace.

3.2.2 Evaluation Function

Sensors synchronize the value of $k_i, i \in G_t$ and let k_i increase periodically at a similar pace using the state transition functions: $f(s, e^{\mathrm{b}})$, $f(s, e^{\mathrm{m}})$, and $f(s, e^{\mathrm{t}})$. To evaluate the precision of an algorithm, the maximum skew of rounds between sensors is taken into account:

$$\delta(k) = \max_{i_1, i_2 \in G_t} \left| \tau_{i_1}(k) - \tau_{i_2}(k) \right|, \tag{3.3}$$

where $\tau_i(k)$ represents the time instant when the value of k_i changes to k. Maximum clock skew of each round cannot reveal the accuracy of the algorithm, which needs

to consider the errors between the clock ticks and the real time.

Equation 3.3 evaluates the precision of the algorithm. Sensors converge the software clock. If the software clock is identical, then $\delta(k)$ is zero.

3.3 State-of-the-art

This section describes the state of the art of clock synchronization including Average TimeSynch Protocol (ATS) [Schenato and Fiorentin, 2011], Robust Average TimeSynch Protocol (RoATS) [Garone et al., 2015], and Average time synchronization (ATSP) [Wu et al., 2012].

3.3.1 Average TimeSynch Protocol (ATS)

In ATS [Schenato and Fiorentin, 2011], Sensor i is equipped with a local physical clock $c_i(t) \in \mathbb{R}$, which is similar to the physical clock counter c but in real value.

$$c_i(t) = \alpha_i t + \beta_i, \tag{3.4}$$

where α_i and β_i are the frequency and offset of the local physical clock, respectively. Both variables α_i and β_i might be different for each sensor due to the imperfection of the manufacturing, but the values are fixed during the synchronization. To correct this difference of the local physical clock, each sensor i also has a software clock

$$\hat{c}_i(t) = \hat{\alpha}_i(t)c_i(t) + \hat{\beta}_i(t), \tag{3.5}$$

where $\hat{\alpha}_i(t)$ and $\hat{\beta}_i$ are tunable variables. With those variables, sensor i adjusts the variables and converges its software clock with other sensors to achieve clock synchronization.

During the ATS time synchronization [Schenato and Fiorentin, 2011], each sensor sends a packet periodically. At the k-th updates, the sensor i receives a packet from its neighbors j, which is a tuple $(\mathrm{ID}_j, \hat{\alpha}_j(t_k), \hat{\beta}_j(t_k), c_j(t_k))$. The parameter ID_j

is denoted as sensor $j's$ identifier, which is a unique value for each sensor. The parameters $\hat{\alpha}_j$, $\hat{\beta}_j$ and c_j are the latest value of $\hat{\alpha}_j(t)$, $\hat{\beta}_j(t)$, and $c_j(t)$, respectively at the time instant when sensor j sends the packet. The packet is then saved to local storage with the current local physical clock reading $c_{ij}(t_k)$.

if $k > 1$, sensor i performs drift estimate to calculate a relative drift α_{ij}. The Drift and offset compensation is also applied to update the value of both $\hat{\beta}_i(t)$ and $\hat{\alpha}_i(t)$, respectively.

3.3.1.1 Drift estimation and compensation

During drift estimation, a relative drift $\alpha_{ij}(t_k)$ of sensor j to sensor i is estimated using the following equation:

$$\alpha_{ij}(t_k) = \frac{c_j(t_k) - c_j(t_{k-1})}{c_{ij}(t_k) - c_{ij}(t_{k-1})}, \tag{3.6}$$

where $c_{ij}(t_k)$ is the local clock time when sensor i receives the packet from sensor j.

With $\alpha_{ij}(t_k)$, the value of $\hat{\alpha}_i$ is updated as follows:

$$\hat{\alpha}_i(k) = \rho_\alpha \hat{\alpha}(k-1) + (1-\rho_\alpha)\alpha_{ij}(t_k)\hat{\alpha}_i(k-1), \tag{3.7}$$

where $\rho_\alpha \in (0, 1)$ is a user-defined parameter that adjust the convergence rate.

3.3.1.2 Offset compensation

In the offset compensation, the offset β_i is updated using the difference of $\hat{c}_j(t_k)$ and $\hat{c}_i(t_k)$.

$$\hat{\beta}_i(k) = \rho_\beta \hat{\beta}(k-1) + (1-\rho_\beta)\left[\hat{c}_j(t_k) - \hat{c}_i(t_k)\right] \tag{3.8}$$

Assume that α_i and β_i remain constant during the synchronization process, the transmission delay between sensors is zero, and the scenario is fully connected, then the software clocks $\hat{c}_i(t)$ in the wireless sensor network eventually converge.

ATS shows an elegant iterative method to converge the software clocks in the wireless sensor network. However, it does not take into account the transmission delay during the communication of the sensors. When considering the transmission delay, original ATS [Schenato and Fiorentin, 2011] cannot ensure the convergence of the software clocks as the incremental error is accumulated during the iteration process.

3.3.2 Robust Average TimeSynch Protocol (RoATS)

Robust Average TimeSynch Protocol from Garone et al. [2015] is proposed to improve the ATS method by adding an extra information exchange between sensors. In RoATS, sensors take an extra step to exchange α_{ij} and α_{ji}, so that the clock drift $\hat{\alpha}_i(k)$ and $\hat{\alpha}_j(k)$ is updated symmetrically to each other for convergence.

The main advance of RoATS is the drift compensation. To describe the modification of RoATS to ATS, let $\alpha^*_{ij}(t_k)$ be the relative drift which considers noise. Equation 3.7 is then reformulated as

$$\hat{\alpha}_i(k) = \hat{\alpha}(k{-}1) + (1{-}\rho_\alpha)\left[\alpha^*_{ij}(t_k)\hat{\alpha}_i(k{-}1) - \hat{\alpha}(k{-}1)\right], \tag{3.9}$$

with

$$\alpha^*_{ij}(t_k) = \epsilon_{ij}(t_k)\alpha_{ij}(t_k), \tag{3.10}$$

where

$$\epsilon_{\min} = \frac{1}{1+\frac{\delta_{ij,\max}}{T_{\min}}} \leq \epsilon_{ij}(t_k) \leq \frac{1}{1-\frac{\delta_{ij,\max}}{T_{\min}}} = \epsilon_{\max}. \tag{3.11}$$

The variable δ_{ij} is the transmission delay, and T is the update period. To consider noise during transmission, the bound of $\alpha_{ij}(t_k)$ and $\alpha_{ji}(t_k)$ can be derived using Equation 3.11 and Equation 3.10.

$$\frac{\alpha^*_{ij}(t_k)}{\epsilon_{\max}} \leq \frac{\alpha_j(t_k)}{\alpha_i(t_k)} \leq \frac{\alpha^*_{ij}(t_k)}{\epsilon_{\min}}, \tag{3.12}$$

and

$$\frac{\epsilon_{\min}}{\alpha^*_{ji}(t_k)} \leq \frac{\alpha_i(t_k)}{\alpha_j(t_k)} \leq \frac{\epsilon_{\max}}{\alpha^*_{ji}}, \tag{3.13}$$

from which it follows

$$\underline{\alpha}^*_{ij}(t_k) \leq \alpha_{ij}(t_k) = \frac{\alpha_i(t_k)}{\alpha_j(t_k)} \leq \overline{\alpha}^*_{ij}(t_k), \tag{3.14}$$

where

$$\underline{\alpha}^*_{ij}(t_k) = \max\left\{\frac{\alpha_{ij}}{\epsilon_{\max}}, \frac{\epsilon_{\min}}{\alpha_{ji}}\right\}, \tag{3.15}$$

and

$$\overline{\alpha}^*_{ij}(t_k) = \min\left\{\frac{\alpha_{ij}}{\epsilon_{\min}}, \frac{\epsilon_{\max}}{\alpha_{ji}}\right\}. \tag{3.16}$$

With the error bound calculated, the correction direction and magnitude are computed locally. The direction sign $s(k)$ is used to determine the convergence direction. The range of direction sign $s(k)$ is $(-1, 0, 1)$, which is designed as follows:

$$\begin{aligned} s(k) = &\frac{1}{2}\text{sign}\left[\underline{\alpha}^*_{ij}(t_k)\hat{\alpha}_i(t_{k-1}) - \hat{\alpha}(t_{k-1})\right] \\ &+ \frac{1}{2}\text{sign}\left[\overline{\alpha}^*_{ij}(t_k)\hat{\alpha}_i(t_{k-1}) - \hat{\alpha}(t_{k-1})\right]. \end{aligned}$$

Note that if the correction term $\alpha^*_{ij}(t_k)\hat{\alpha}_i(t_{k-1}) - \hat{\alpha}(t_{k-1})$ is close to zero, then the update direction cannot be decided due to the noise, so the two sign functions become opposite and $s(k) = 0$.

The magnitude is computed using the smallest correction for both sensors i and j to guarantee the value of update steps are the same.

$$
\begin{aligned}
m(t_k) = \min \{ & \left| \underline{\alpha}^*_{ij}(t_k)\hat{\alpha}_i(t_{k-1}) - \hat{\alpha}(t_{k-1}) \right|, \\
& \left| \overline{\alpha}^*_{ij}(t_k)\hat{\alpha}_i(t_{k-1}) - \hat{\alpha}(t_{k-1}) \right|, \\
& \left| \frac{1}{\underline{\alpha}^*_{ij}(t_k)}\hat{\alpha}_i(t_{k-1}) - \hat{\alpha}(t_{k-1}) \right|, \\
& \left| \frac{1}{\overline{\alpha}^*_{ij}(t_k)}\hat{\alpha}_i(t_{k-1}) - \hat{\alpha}(t_{k-1}) \right| \} .
\end{aligned}
$$

Finally, the drift compensation is reformulated as

$$
\hat{\alpha}_i(t_k) = \hat{\alpha}(t_{k-1}) + (1-\rho_\alpha)\left[s(t_k)m(t_k) - \hat{\alpha}(t_{k-1}) \right] \tag{3.17}
$$

With the term $s(k)m(k)$, RoATS converges $\hat{c}_i(t)$ considering the transmission delay. RoATS inherit the benefit of the ATS, which is a decentralized and one hop recursion algorithm. However, RoATS uses additional communication to perform the synchronization, which is not possible when the communication graph is direct. With the direct edge, when a sensor receives a message, it might not able to send back the calculations. In RoATS, sensors need to perform back-and-forth communication to achieve their clocks synchronized.

3.3.3 Average time synchronization (ATSP)

In ATSP [Wu et al., 2012], the two neighboring nodes exchange their clock values and calculate the average value. In the k-th update round, three packets are transmitted between sensor i and sensor j. The three packet protocol is as follows:

1. For the first packet, sensor i sends a message at its local physical clock $c_i^{\mathrm{s1}}(k-1)$ to sensor j, and sensor j receives the message at its local physical clock $c_j^{\mathrm{r1}}(k-1)$. Then $c_j^{\mathrm{r1}}(k-1) = c_i^{\mathrm{s1}}(k-1) + d + \delta$, where d is the transmission delay and δ is the clock offset.

2. For the second packet, sensor j sends the second packet back to sensor i at

$c_j^{s2}(k-1)$ as soon as the first packet is received. The second packet contains the information of both $c_j^{\mathrm{r1}}(k-1)$, and $c_j^{\mathrm{s2}}(k-1)$. When sensor i receiving the second packet at $c_i^{\mathrm{r1}}(k-1)$, it estimates the average delay as

$$d = \frac{1}{2}\left(c_j^{r1}(k-1) - c_i^{s1}(k-1)\right) + \frac{1}{2}\left(c_i^{\mathrm{r2}}(k-1) - c_j^{s2}(k-1)\right) \tag{3.18}$$

and updates its clock to

$$c_i(k) = \frac{1}{2}c_i^{\mathrm{r2}}(k-1) + \frac{1}{2}\left(c_j^{\mathrm{s2}}(k-1) + d\right), \tag{3.19}$$

3. In order to obtain the time average synchronization, the third packet is required to send the information back from sensor i to sensor j. Similar to the second packet, when sensor i receives the second packet, it sends the third packet at $c_i^{\mathrm{s3}}(k-1)$ immediately. The third packet contains the information of $c_i^{r2}(k-1)$, $c_i^{\mathrm{s3}}(k-1)$, and the updated clock $c_i(k)$. When receiving the packet, sensor j updates its clock as follows:

$$c_j(k) = \frac{1}{2}c_j^{\mathrm{r3}}(k-1) + \frac{1}{2}\left(c_i^{\mathrm{s3}}(k-1) + d\right). \tag{3.20}$$

ATSP uses pairwise messages exchanges to reduce the clock offset. ATSP apply the third packet transmission to update the clocks of both the paired sensor. The disadvantage is that the communication method is relatively complicated. In the real environment, communication between sensors in the wireless network might have many communication faults, which reduce the convergence rate significantly.

3.4 Conclusion

This chapter states the problem formulation of clock synchronization and reviews three state-of-the-art methods widely used. We also highlight their main advantages and drawbacks. Clock synchronization is still an open problem due to its complexity,

limitation of the energy of sensors, and high wireless communication fault rate compared to the static wired networks. Research on energy efficient and fault tolerant protocol that can be applied to dynamic sensor network are the objectives of many research works. Next chapter presents our pulse synchronization protocol, which intends to overcome the limitation of the state-of-the-art algorithms.

Chapter 4

Time Wheel Algorithm with Drift Compensation

In this chapter, we introduce the pulse-coupled algorithm. First, the synchronization function for different events is detailed. The synchronization function is executed by a local sensor, which predicts the pulse of next round. The synchronization function saves the timing information of pulses in the time wheel and use it to calculate the next pulse. The synchronization function calculates the offset and the time period of pulses, and then synchronizes the pulse of the next round with the neighboring sensor nodes. To synchronize the pulses, both offset and drift compensation are applied. Finally, the convergence rate of the algorithm is analyzed.

4.1 Introduction

In the previous chapter, the events $e^{\mathrm{b}}, e^{\mathrm{m}}, e^{\mathrm{t}} \subset \mathcal{E}$ are introduced, which occur during the clock synchronization. A sensor uses synchronization functions to handle the event. When executing synchronization functions, a sensor can broadcast messages to its neighbors and to setup a timer. The timer is used to estimate the timing for the next pulse to broadcast. The timer is set by the timer set function to correct the

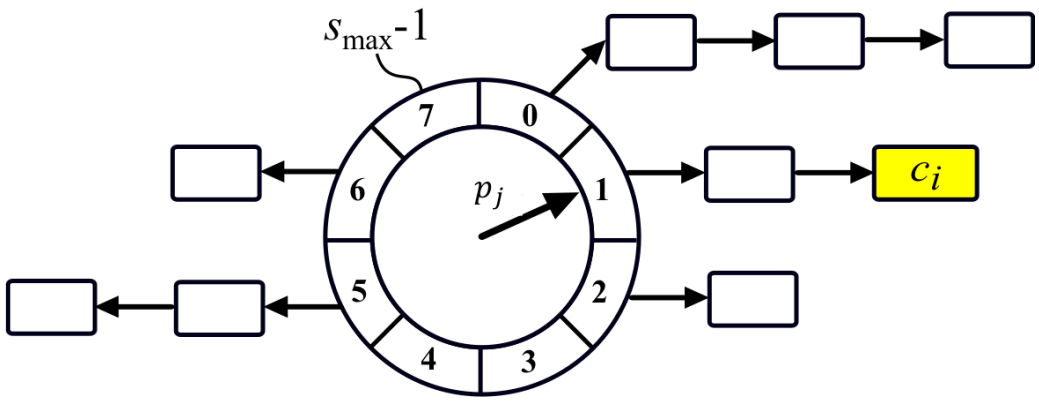


Figure 4.1: When sensor i receives a pulse with a tag p_j from sensor j, it appends the physical local clock time c_i to the list of p_j-th slot of w_i.

clock offset and drift. The sections below detail the process.

4.2 Synchronization functions

We start by describing the general round-structure that our algorithm operates in. At the beginning of each round, i.e., when the round number k_i changes, sensor i broadcasts an electromagnetic pulse to its neighbors, where the pulse is a short burst of electromagnetic energy. By recording the local time of pulses in the present round, sensors are able to converge the timing and switch to the next round.

Sensors start in round $k_i = 0$ and broadcast a pulse in round $k_i - 1$. A sensor i waits for a period $T_i(k)$, which is subject to its local clock drift. Because the durations $T_i(k)$ and the round start times may not be the same for different sensors, pulses are not perfectly synchronized with each other. Therefore, corrections must be computed using information of previous pulses from neighboring sensors.

In previous work [Függer et al., 2015b], round numbers are assumed to be known for all sensors, which is not always realistic, because rounds estimated by sensors can be faster or slower. In our work, sensor i broadcasts a value $p_i \in \mathbb{Z}, 0 \leq p \leq p_{\max} - 1$. The value $p_{\max}$ is usually small so p can be represented by different frequency of

the pulse. The value p_i serves a similar role as k_i, which increases every round, except when p_i reaches $p_{\max}$, it returns to zero for the next round. When $p_{\max} = 1$, it is not possible for sensors to distinguish the different rounds. With higher $p_{\max}$, sensors are able to converge the p_i. However, if $p_{\max}$ is higher, it takes more time to converge due to the algorithm converge process. In our experiment, we discover that $p_{\max} = 2$ is sufficient for the sensors to synchronize efficiently. If $p_{\max}$ is large, then sensors need to transmit more bits, which leads to transmission delay. If $p_{\max} = 0$ or $p_{\max} = 1$, then it is difficult for sensors to distinguish the round numbers. In a view of implementation, broadcasting $p_{\max}$ can be represented using different frequency or wave pattern of the pulses.

A sensor i uses a circular buffer called *time wheel* w_i to record p_j from neighboring sensor j. Figure 4.1 shows the structure of the time wheel w_i. A time wheel of sensor i has $p_{\max}$ slots. The pointer in the center of time wheel is directed to the value of p_j just broadcasted by sensor j. When sensor i receives a pulse with a tag p_j from sensor j, it appends the local clock time c_i to the list of p_j-th slot of w_i. The timing information in w_i will be used to estimate the pulse for the next round.

Each sensor runs the pseudo-code given in Algorithm 4.1. The code is divided into three parts: initial function (line 1), receive function (line 5), and state transition function (line 9). These three parts are not executed in sequence. Instead, they are triggered by different events. The initial function is executed once when the sensors start to synchronize the clock. The receive function is executed whenever a pulse with a pointer value p_j is received. The state transition function is executed when a new round starts.

Initial function At the beginning of the algorithm, sensor i initializes an empty time wheel w_i with $p_{\max}$ slots (line 2). The sensor reset p_i (line 3) to zero. The variable c_i is the counter of the physical clock started at initialization, which increases subject to the clock drift. The sensor also broadcasts p_i to its neighbors (line 4).

Algorithm 4.1 State transitions for sensor i

1: **if** e^{b} **then**
2: $w_i = \mathrm{TimeWheel}(s_{\max})$
3: $p_i = 0$
4: sending_function(p_i)
5: **else if** e^{m} **then**
6: Obtain p_j from m
7: Append c_i to the list in p_j-th slot
8: $\tau(k+1) =$ timer_set_function(w_i, $\tau(k)$) and set the count down timer
9: **else if** e^{t} **then**
10: Clear the p_i-th slot
11: $p_i = (p_i + 1) \bmod (p_{\max})$
12: sending_function(p_i)
13: **end if**

Receive function When sensor i receives a message from sensor j, with a tag p_j (line 6). Sensor i appends the current local time c_i of the received message to the list of the p_j-th slot (line 7). Recall that the time between broadcasting and receiving a pulse is assumed to be negligible, since sensors only exchange zero or few bit pulses. A sensor also schedules its next state transition by setting its count down timer to $\tau_i(k+1) - c_i$, which is the time period for the next pulse to broadcast. (line 8). Details of the timer set function can be found in section 4.2.

State transition function When the timer reaches zero, the sensor executes its state transition function and start the next round. First, the sensor clears the current slot from the current slot p_i (line 10), because the information in the slot is used. Second, it updates p_i (line 11). In the end, it broadcasts p_i to its neighboring sensors (line 12).

In previous work from Függer et al. [2015b], the rounds are assumed known by sensors and the pulses in the same round is assumed to have small differences, in this work, the time wheel is applied to improve the robustness of the algorithm, so the assumption is no longer needed.

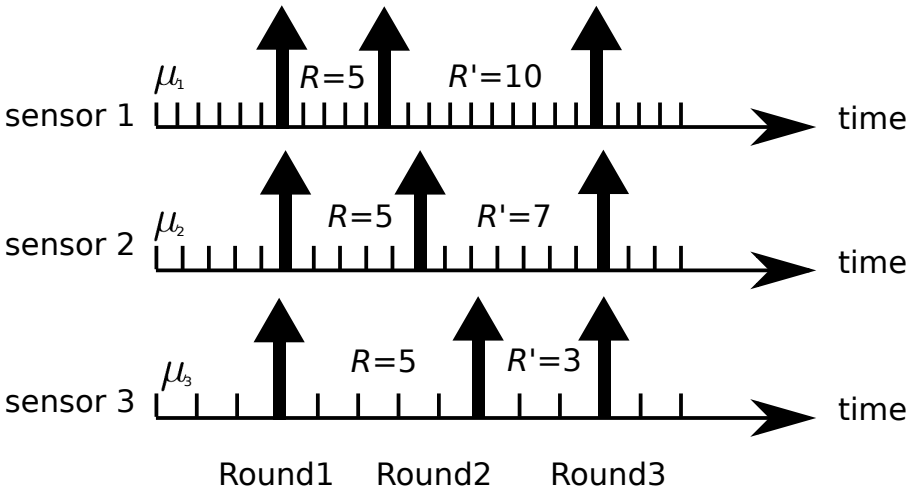


Figure 4.2: Illustration of a pulse synchronization. At the first round, sensors broadcast pulses at a similar time. In the second round, R is equal to 5 for all the sensors. However, the constant clock ticks R do not represent the constant time period due to the manufacturing defects and clock drifts. The difference of clock period causes divergence of the timing of pulses. In the third round, sensors adjust its $R' = R + \text{corr}_i^1 + \text{corr}_i^2$, so that the clock skew becomes smaller. Notice that we setup $k_{\max} = 3$, so the round number can be 3.

The timer set function in Algorithm 4.1 (line 8) estimates the counter value for next pulse $\tau(k+1)$ according to the following formula:

$$\tau_i(k+1) = \tau_i(k) + R + \text{corr}_i^1(k) + \text{corr}_i^2(k), \tag{4.1}$$

where R is a constant representing the time period between the next and present rounds. All sensors use the same value of R, but the time period between pulses $T_i(k+1)$ may be different due to frequency variation of local clocks described in Equation 3.2. The corrections $\text{corr}_i^1(k)$ and $\text{corr}_i^2(k)$ are the adjustment of the timing to reduce the clock skew. These corrections adjust the pulse period using the information saved in the time wheel w. Figure 4.2 illustrates the pulse synchronization method. In the first round, sensors broadcast pulses at a similar time. In the second round, R is equal to 5 for all the sensors. However, the constant clock ticks R

do not represent the constant time period due to the manufacturing defects and clock drifts. The difference of clock period causes divergence of timing of pulses. At the third round, sensors adjust its $R' = R + \text{corr}_i^1(k) + \text{corr}_i^2(k)$, so that the clock skew becomes smaller. Notice that both $\text{corr}_i^1(k)$ and $\text{corr}_i^2(k)$ depend on the previous round k.

Each sensor only communicates through pulses and additional round information c_i. The overall benefit of the method is that sensors do not require to identify their neighboring sensor and their pulses, so the message is short and robust.

4.2.1 Offset compensation

The term corr_i^1 is used for correcting the offset of the pulses for all the sensors in previous work [Függer et al., 2015b]. corr_i^1 is a time difference between the average pulse of the neighboring sensors and the pulse of sensor i itself in round k:

$$\text{corr}_i^1(k) = \frac{1}{|\text{In}_i(k+1)|} \sum_{j \in \text{In}_i(k+1)} (\tau_j(k) - \tau_i(k)) \tag{4.2}$$

where $|\text{In}_i(k+1)|$ is the number of incoming messages at the $k+1$-th round and $\sum_{j \in \text{In}_i(k+1)} (\tau_j(k) - \tau_i(k))$ is calculated by averaging the values in the current slot of wheel w_i. Compared to the original next pulse $\tau_i(k+1) = \tau_i(k) + R$, the equation considering corr_i^1 becomes

$$\begin{aligned} \tau_i(k+1) &= \left(\tau_i(k) + \text{corr}_i^1\right) + R \\ &= \frac{1}{|\text{In}_i(k+1)|} \sum_{j \in \text{In}_i(k+1)} \tau_j(k) + R. \end{aligned}$$

The equation shows that instead of using its previous pulse, sensor i uses the average of the previous pulses of its neighbors to estimate the timing of the next pulse. Thus, the precision of the next pulse tends to increase. However, the constant clock ticks R do not represent the constant time period due to the manufacturing defects and

clock drifts. In order to further reduce the clock skew of the next pulse, R should also adjust to adapt the variation of local physical clocks.

4.2.2 Drift compensation

The goal of drift compensation is to let sensors agree on a global clock cycle. Since sensor i converges the previous pulses $\tau_i(k)$ with $(\tau_i(k) + \text{corr}_i^1(k))$, where $\text{corr}_i^1(k)$ is the correction for the previous pulse $\tau_i(k)$. We obtain an approximation of time period (in terms of the number of clock ticks) $(R + \text{corr}_i^2(k))$ in the current round as follows:

$$R + \text{corr}_i^2(k) = \left(\tau_i(k) + \text{corr}_i^1(k)\right) \frac{1}{k}, \tag{4.3}$$

or

$$\text{corr}_i^2(k) = \left(\tau_i(k) + \text{corr}_i^1\right) \frac{1}{k} - R. \tag{4.4}$$

In our experiment, if drift compensation is enabled at the beginning, the consensus may not converge under some extreme cases, so the round filter is applied to Equation 4.4:

$$\text{corr}_i^2(k) = \begin{cases} 0 & k \leq \ell \\ (\tau_i(k) + \text{corr}_i^1(k)) \frac{1}{k} - R & \text{else} \end{cases}. \tag{4.5}$$

By using Equation 4.5, the drift compensation only enabled after the ℓ-th round. The reason to add this filter will be derived and discussed mathematically in the next section.

4.3 Convergence Analysis

In this section, we analyze first the effect of OC algorithm (the correction with only offset correction), then analyze the effect of OCDC algorithm (the correction for both

offset Correction and drift Correction).

4.3.1 Convergence Analysis for OC algorithm

In OC algorithm, only corr_i^1 is enabled. Recall that the relation between time and the counter of the local clock of sensors is as follows:

$$t_i(k) = \sum_{c=1}^{\tau_i(k)} \mu_i(c), \tag{4.6}$$

where $t_i(k)$ represents the time of k-th pulse broadcasted by sensor i. $\tau_i(k)$ is the local physical clock counter of sensor i when it broadcasts k-th pulse. $\mu_i(c)$ is the time period of a clock cycle, which is bounded by

$$(1 - \varrho)\mu \leq \mu_i(c) \leq (1 + \varrho)\mu. \tag{4.7}$$

The incremental physical clock counter in each sensor is assumed to be discrete. Each value of the physical clock counter represents a period of time.

Definition 1. Denote by $\tau_i(k)$ the local tick number of sensor i at which sensor i starts its k^{th} round and by $t_i(k)$ the corresponding real time. Denote by $\tau_j^i(k)$ the local tick number of sensor i at which it receives the pulse of sensor j for round k.

A sensor i reads the clock timing of the pulse from sensor j within the bound:

$$\sum_{c=1}^{\tau_j^i(k)-1} \mu_i(c) \leq t_j(k) \leq \sum_{c=1}^{\tau_j^i(k)} \mu_i(c), \tag{4.8}$$

with the assumption that the transmission delay between sensors is zero. The variable $t_j(k)$ is the real world time of k-th pulse broadcasted by sensor j, and $\tau_j^i(k)$ is the physical local clock counter of sensor i when it receive the k-th pulse from sensor j. The variable $t_j(k)$ and $\tau_j^i(k)$ indicate that Equation 4.8 only holds if sensor i receives the message from sensor j in round k.

Lemma 1. *The relationship of the local clock difference* $\tau_j^i(k) - \tau_i(k)$ *and the time difference* $t_j(k) - t_i(k)$ *is*

$$\tau_j^i(k) - \tau_i(k) = \left(t_j(k) - t_i(k)\right)/\beta_j^i(k) + \alpha_j^i(k), \tag{4.9}$$

where $\beta_j^i(k) \in [\mu(1-\varrho), \mu(1+\varrho)]$ is caused by clock skew and $\alpha_j^i(k) \in [-1, 1]$ is caused by imprecision of the discrete time.

Proof. According to Equation 3.1 and Equation 4.8, the timing of the pulse is assumed to be within the bound:

$$\sum_{c=1}^{\tau_j^i(k)-1} \mu_i(c) \le t_j(k) \le \sum_{c=1}^{\tau_j^i(k)} \mu_i(c), \tag{4.10}$$

and

$$\sum_{c=1}^{\tau_i(k)-1} \mu_i(c) \le t_i(k) \le \sum_{c=1}^{\tau_i(k)} \mu_i(c). \tag{4.11}$$

Combining Equation 4.11 and Equation 4.10, we have

$$\sum_{c=1}^{\tau_j^i(k)-1} \mu_i(c) - \sum_{c=1}^{\tau_i(k)} \mu_i(c) \le t_j(k) - t_i(k) \le \sum_{c=1}^{\tau_j^i(k)} \mu_i(c) - \sum_{c=1}^{\tau_i(k)-1} \mu_i(c), \tag{4.12}$$

If $\tau_j^i(k) - 1 \ge \tau_i(k)$, the left-hand side of Equation 4.12 is reformulated using Equation 4.7:

$$\sum_{c=1}^{\tau_j^i(k)-1} \mu_i(c) - \sum_{c=1}^{\tau_i(k)} \mu_i(c) = \sum_{c=\tau_i(k)+1}^{\tau_j^i(k)-1} \mu_i(c) \tag{4.13}$$

$$\ge \left(\tau_j^i(k) - \tau_i(k) - 1\right)(1-\varrho). \tag{4.14}$$

If $\tau_j^i(k) - 1 \leq \tau_i(k)$, with the similar derivation as Equation 4.14, the left-hand side of Equation 4.12 is

$$\sum_{c=1}^{\tau_j^i(k)-1} \mu_i(c) - \sum_{c=1}^{\tau_i(k)} \mu_i(c) = -\sum_{c=\tau_j^i(k)}^{\tau^i(k)} \mu_i(c) \tag{4.15}$$

$$\geq -\left(\tau_i(k) - \tau_j^i(k) + 1\right)(1-\varrho)\,\mu \tag{4.16}$$

$$= \left(\tau_j^i(k) - \tau_i(k) - 1\right)(1-\varrho)\,\mu, \tag{4.17}$$

With Equation 4.14 and Equation 4.17, we have

$$\sum_{c=1}^{\tau_j^i(k)-1} \mu_i(c) - \sum_{c=1}^{\tau_i(k)} \mu_i(c) \geq \left(\tau_j^i(k) - \tau_i(k) - 1\right)(1-\varrho)\,\mu, \tag{4.18}$$

With the similar derivation as Equation 4.18, the right-hand-side of Equation 4.12 is

$$\sum_{c=1}^{\tau_j^i(k)} \mu_i(c) - \sum_{c=1}^{\tau_i(k)-1} \mu_i(c) \leq \left(\tau_j^i(k) - \tau_i(k) + 1\right)(1+\varrho)\,\mu. \tag{4.19}$$

Combining Equation 4.18, Equation 4.19 with Equation 4.12, we have

$$\left(\tau_j^i(k) - \tau_i(k) - 1\right)(1-\varrho)\,\mu \leq t_j(k) - t_i(k) \leq \left(\tau_j^i(k) - \tau_i(k) + 1\right)(1+\varrho)\,\mu \tag{4.20}$$

From Equation 4.20, we get

$$t_j(k) - t_i(k) = \left(\tau_j^i(k) - \tau_i(k) - \alpha_j^i(k)\right)\beta_j^i(k) \tag{4.21}$$

with

$$\beta_j^i(k) \in [\mu(1-\varrho), \mu(1+\varrho)] \tag{4.22}$$

and

$$\alpha_j^i(k) \in [-1, 1] \tag{4.23}$$

to build Equation 4.9. □

Lemma 2. *The relationship of the local clock difference* $\tau_j^i(k+1) - \tau_i(k)$ *and the time difference* $t_j(k+1) - t_j(k)$ *is*

$$\tau_i(k+1) - \tau_i(k) = \left(t_i(k+1) - t_i(k)\right)/\beta_i(k) + \alpha_i(k) \tag{4.24}$$

where $\beta_i(k) \in [\mu(1-\varrho), \mu(1+\varrho)]$ and $\alpha_i(k) \in [-1, 1]$.

Proof. The timing of the pulse from sensor j to i is received as a discrete time event. According to Equation 3.1 and Equation 4.8, we get:

$$\sum_{c=1}^{\tau_i(k+1)-1} \mu_i(c) \leq t_i(k+1) \leq \sum_{c=1}^{\tau_i(k+1)} \mu_i(c). \tag{4.25}$$

Combining with Equation 4.7, we have

$$\sum_{c=1}^{\tau_i(k+1)-1} \mu_i(c) - \sum_{c=1}^{\tau_i(k)} \mu_i(c) \leq t_i(k+1) - t_i(k) \leq \sum_{c=1}^{\tau_i(k+1)} \mu_i(c) - \sum_{c=1}^{\tau_i(k)-1} \mu_i(c), \tag{4.26}$$

If $\tau_i(k+1) \geq \tau_i(k) + 1$, the left-hand side of Equation 4.26 is

$$\sum_{c=1}^{\tau_i(k+1)-1} \mu_i(c) - \sum_{c=1}^{\tau_i(k)} \mu_i(c) = \sum_{c=\tau_i(k)+1}^{\tau_i(k+1)-1} \mu_i(c) \tag{4.27}$$

$$\geq \left(\tau_i(k+1) - \tau_i(k) - 1\right)(1-\rho)\,\mu. \tag{4.28}$$

If $\tau_i(k+1) - 1 \leq \tau_i(k)$, the left-hand side of Equation 4.26 is

$$\sum_{c=1}^{\tau_i(k+1)-1} \mu_i(c) - \sum_{c=1}^{\tau_i(k)} \mu_i(c) = - \sum_{c=\tau_i(k+1)}^{\tau_i(k)} \mu_i(c) \tag{4.29}$$

$$\geq - \left(\tau_i(k) - \tau_i(k+1) + 1\right)(1-\rho)\mu \tag{4.30}$$

$$\geq \left(\tau_i(k+1) - \tau_i(k) - 1\right)(1-\rho)\mu, \tag{4.31}$$

With Equation 4.28 and Equation 4.31 we have

$$\sum_{c=1}^{\tau_i(k+1)-1} \mu_i(c) - \sum_{c=1}^{\tau_i(k)} \mu_i(c) \geq \left(\tau_i(k+1) - \tau_i(k) - 1\right)(1-\rho)\mu, \tag{4.32}$$

With the similar derivation, the right-hand-side of Equation 4.26 is

$$\sum_{c=1}^{\tau_i(k+1)} \mu_i(c) - \sum_{c=1}^{\tau_i(k)-1} \mu_i(c) \leq \left(\tau_i(k+1) - \tau_i(k) + 1\right)(1+\rho)\mu. \tag{4.33}$$

Combining Equation 4.32, Equation 4.33 with Equation 4.26, we have

$$\left(\tau_i(k+1) - \tau_i(k) - 1\right)(1-\varrho)\,\mu \leq t_j(k) - t_i(k) \leq \left(\tau_i(k+1) - \tau_i(k) + 1\right)(1+\varrho)\,\mu \tag{4.34}$$

From Equation 4.34, we get

$$t_i(k+1) - t_i(k) = \left(\tau_i(k+1) - \tau_i(k) - \alpha_i(k)\right)\beta_i(k) \tag{4.35}$$

where

$$\beta_i(k) \in [\mu(1-\varrho), \mu(1+\varrho)] \tag{4.36}$$

and

$$\alpha_i(k) \in [-1, 1] \tag{4.37}$$

to build Equation 4.24. □

For the duration of round $k+1$, the algorithm sets its timer to the following number of local clock ticks:

$$\tau_i(k+1) - \tau_i(k) = \frac{1}{|\mathrm{In}_i(k)|} \sum_{j \in \mathrm{In}_i(k)} \left(\tau_j^i(k) - \tau_i(k)\right) + R, \tag{4.38}$$

where R is the original number of clock ticks between pulses.

To calculate the maximum pulse skew differences of sensors in round k, the Dobrushin semi-norm is introduced as follows defined on real vectors $\mathbf{x}$ by

$$\delta(\mathbf{x}) = \max_{i,j} |x_i - x_j|. \tag{4.39}$$

By setting $\mathbf{x} = \mathbf{t}(k), \mathbf{t}(k) = [t_1(k), ..., t_n(k)]^T$, we calculate the maximum pulse skew in the round k. Using properties of the Dobrushin seminorm, we are able to calculate the upper bound of the maximum pulse skew. The following conditions hold for all vector $\mathbf{x}$ and $\mathbf{y} \in \mathbb{R}^n$, and any scalar c.

- 1. $\delta(\mathbf{x}) \geq 0$,
- 2. $\delta(c\mathbf{x}) = |c|\, \delta(\mathbf{x})$, and
- 3. $\delta(\mathbf{x} + \mathbf{y}) \leq \delta(\mathbf{x}) + \delta(\mathbf{y})$

To estimate the averaging operation of sensors, the network G_t of WSN can be interpreted as $\mathbf{A}(k)$:

$$\mathbf{A}(k)_{i,j} = \begin{cases} \frac{1}{|\mathrm{In}_i(k)|} & (i,j) \in \mathcal{V} \\ 0 & \text{else} \end{cases}. \tag{4.40}$$

Note that with $(i,j) \in \mathcal{V}$, the matrix $\mathbf{A}(k)$ is stochastic because the sum of row $\sum_{j=1}^{n} \mathbf{A}(k)_{i,j}$ is equal to one for $1 \leq j \leq n$. The Dobrushin semi-norm can be apply to a real matrix $\mathbf{A}$ as follows.

$$\delta(\mathbf{A}) = \sup_{\mathbf{x} \in \mathbb{R}^n, \delta(\mathbf{x}) \neq 0} \frac{\delta(\mathbf{A}\mathbf{x})}{\delta(\mathbf{x})}. \tag{4.41}$$

The properties of the stochastic matrix and the Dobrushin semi-norm described as follows:

Lemma 3. *(Charron-Bost [2013]). Let* $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{n\times n}$. *If* $\mathbf{A}$ *is stochastic, we have* $\delta(\mathbf{AB}) \leq \delta(\mathbf{A})\delta(\mathbf{B})$.

When $\mathbf{A}$ is stochastic, we have $\delta(\mathbf{A}) \leq 1$. We call stochastic matrices as a scrambling matrices if the strict inequality holds.

Lemma 4. *(Hajnal and Bartlett [1958]) Let A be a stochastic matrix with and minimal positive entry* γ. *Then,*

$$\delta(A) = 1 - \min_{i_1,i_2} \sum_j \min\left\{A_{i_1,j}, A_{i_2,j}\right\}. \tag{4.42}$$

Since the network is a non-split graph, the matrix A has the scrambling property. With scrambling property and Lemma Equation 4.42, we have $\delta(A) \leq 1-\gamma$, *where* γ *is the minimal positive entry of all matrices* $A(k)$ *during the execution.*

Since $0 \leq A(k)_{i,j} \leq 1$ according to Equation 4.40, it is a scrambling matrix, then

$$\delta\left(\prod_{\ell=1}^{k} A(k)\right) = \prod_{\ell=1}^{k} \delta\left(A(k)\right) \leq (1-\gamma)^k, \tag{4.43}$$

In the worse case scenario, sensors are disconnect from each other. Performing averaging method will not help since $A = I$, $\delta(A) = 1$ and $\delta\left(\prod_{\ell=1}^{k} A(k)\right) = 1$. In a better scenario, a connected network (connected graph), each sensor has neighbors, i.e., $|\mathrm{In}_i(k)| > 1$, so $\gamma < 1$ and the value can be converge through the averaging process.

.

Lemma 5. *Equation 4.38 can be transformed to real time in matrix form*

$$t(k+1) = A(k)t(k) + b(k) + c(k), \tag{4.44}$$

where $A(k)$ is a weighted adjacency matrix, which satisfies

$$A_{i,j}(k) = \begin{cases} 1/|\mathrm{In}_i(k)| & \text{if } j \in \mathrm{In}_i(k) \\ 0 & \text{else .} \end{cases} \tag{4.45}$$

and

$$b_i(k) = \left(\frac{\beta_i(k)}{\overline{\beta_j^i}(k)} - 1 \right) \left(\left(A(k)t(k) \right)_i - t_i(k) \right) \tag{4.46}$$

and

$$c_i(k) = \left(-\alpha_i(k) + \overline{\alpha_j^i}(k) + R \right) \beta(k). \tag{4.47}$$

with the existence of some $\overline{\beta_j^i}(k) \in [\mu(1-\varrho), \mu(1+\varrho)]$ and $\overline{\alpha_j^i}(k) \in [-1, 1]$.

Proof. Substitute $\tau_j^i(k) - \tau_i(k)$ in Definition 1 using Lemma 1, we get

$$\begin{aligned} \tau_i(k+1) - \tau_i(k) &= \frac{1}{|\mathrm{In}_i(k)|} \sum_{j \in \mathrm{In}_i(k)} \left(\tau_j^i(k) - \tau_i(k) \right) + R \\ &= \frac{1}{|\mathrm{In}_i(k)|} \sum_{j \in \mathrm{In}_i(k)} \left(\left(t_j(k) - t_i(k) \right) / \beta_j^i(k) + \alpha_j^i(k) \right) + R. \\ &= \frac{1}{|\mathrm{In}_i(k)|} \sum_{j \in \mathrm{In}_i(k)} \left(t_j(k) - t_i(k) \right) / \beta_j^i(k) + \frac{1}{|\mathrm{In}_i(k)|} \sum_{j \in \mathrm{In}_i(k)} \alpha_j^i(k) + R. \end{aligned}$$

The lower bound can be calculated by taking the largest $\beta_j^i(k)$ and smallest $\alpha_j^i(k)$.

$$\begin{aligned}\tau_i(k+1)-\tau_i(k) &\geq \frac{1}{|\mathrm{In}_i(k)|}\sum_{j\in\mathrm{In}_i(k)}\left(t_j(k)-t_i(k)\right)/\mu\left(1+\varrho\right)-\frac{1}{|\mathrm{In}_i(k)|}\sum_{j\in\mathrm{In}_i(k)}1+R\\ &=\frac{1}{|\mathrm{In}_i(k)|}\left(\sum_{j\in\mathrm{In}_i(k)}\left(t_j(k)-t_i(k)\right)\right)\frac{|\mathrm{In}_i(k)|}{\mu\left(1+\varrho\right)}-\frac{1}{|\mathrm{In}_i(k)|}|\mathrm{In}_i(k)|+R\\ &=\frac{1}{|\mathrm{In}_i(k)|}\frac{1}{\mu\left(1+\varrho\right)}\left(\sum_{j\in\mathrm{In}_i(k)}\left(t_j(k)-t_i(k)\right)\right)-1+R. \qquad (4.48)\end{aligned}$$

With the similar derivation, the upper bound can be calculated as well.

$$\tau_i(k+1)-\tau_i(k)\leq\frac{1}{|\mathrm{In}_i(k)|}\frac{1}{\mu\left(1-\varrho\right)}\left(\sum_{j\in\mathrm{In}_i(k)}\left(t_j(k)-t_i(k)\right)\right)+1+R \qquad (4.49)$$

Combining Equation 4.48 and Equation 4.49, we get the existence of some $\overline{\beta_j^i}(k)\in[\mu(1-\varrho),\mu(1+\varrho)]$ and $\overline{\alpha_j^i}(k)\in[-1,1]$ such that

$$\tau_i(k+1)-\tau_i(k)=\frac{1}{|\mathrm{In}_i(k)|\,\overline{\beta_j^i}(k)}\sum_{j\in\mathrm{In}_i(k)}\left(t_j(k)-t_i(k)\right)+\overline{\alpha_j^i}(k)+R \qquad (4.50)$$

Combining Lemma 2 and Equation 4.50 , we get

$$\left(t_i(k+1)-t_i(k)\right)=\frac{\beta_i(k)}{|\mathrm{In}_i(k)|\,\overline{\beta_j^i}(k)}\sum_{j\in\mathrm{In}_i(k)}\left(t_j(k)-t_i(k)\right)+\left(-\alpha_i(k)+\overline{\alpha_j^i}(k)+R\right)\beta_i(k)$$

Putting $\gamma_i(k) = \frac{\beta_i(k)}{\overline{\beta_j^i}(k)} - 1$ and adding $t_i(k)$ on both sides of the equation, we get

$$\begin{aligned} t_i(k+1) &= \frac{1}{|\mathrm{In}_i(k)|} \sum_{j \in \mathrm{In}_i(k)} t_j(k) + \frac{\gamma_i(k)}{|\mathrm{In}_i(k)|} \sum_{j \in \mathrm{In}_i(k)} \big(t_j(k) - t_i(k)\big) + \Big(-\alpha_i(k) + \overline{\alpha_j^i}(k) + R\Big)\, \beta_i(k) \\ &= \big(A(k)t(k)\big)_i + \gamma_i(k)\Big(\big(A(k)t(k)\big)_i - t_i(k)\Big) + \Big(-\alpha_i(k) + \overline{\alpha_j^i}(k) + R\Big) \cdot \beta_i(k) \end{aligned} \tag{4.51}$$

or in vector form as Equation 4.44. □

Lemma 6. *Semi-norm of $b(k)$ is*

$$\delta\big(b(k)\big) = \delta\left(\left(\frac{\beta_i(k)}{\overline{\beta_j^i}(k)} - 1\right)\Big(\big(A(k)t(k)\big) - t(k)\Big)\right) \leq \frac{4\varrho}{1-\varrho}\delta(k), \tag{4.52}$$

where $\delta(k)$ is a simplified symbol represents the clock skew $\delta\left(t(k)\right)$ in round k.

Proof. According to Equation 4.39, suppose we have t_{i_1} and t_{i_2} such that $\delta(k) = (t_{i_1}(k) - t_{i_2}(k))$, and since $A(k)$ is a stochastic matrix, i.e., $\sum_j \left[A(k)\right]_{i,j} = 1$, we have $t_{i_2}(k) \leq A(k)t(k) \leq t_{i_1}(k)$, which implies

$$\delta\left(\big(A(k)t(k)\big) - t(k)\right)_i \leq \delta(k). \tag{4.53}$$

The bound of $\left(\frac{\beta_i(k)}{\overline{\beta_j^i}(k)} - 1\right)$ is

$$-\frac{2\varrho}{1+\varrho} \leq \left(\frac{\beta_i(k)}{\overline{\beta_j^i}(k)} - 1\right) \leq \frac{2\varrho}{1-\varrho}, \tag{4.54}$$

Considering Equation 4.53 and Equation 4.54 with Equation 4.46, we get

$$-\frac{2\varrho}{1-\varrho}\delta\left(k\right) \leq b_i(k) \leq \frac{2\varrho}{1-\varrho}\delta\left(k\right), \tag{4.55}$$

which means that

$$\delta\left(b(k)\right) \leq \max_{i_1} b_{i_1}(k) - \min_{i_2} b_{i_2}(k) \leq \frac{4\varrho}{1-\varrho}\delta(k). \tag{4.56}$$

□

Lemma 7. *By assuming original number of clock ticks between pulses $R > 2$, the maximum difference of the vector $c(k)$ is*

$$\delta(c(k)) \leq \left(4 + 2\varrho R\right)\mu \tag{4.57}$$

Proof. If $R > 2$, we have $c_i(k) \geq 0$ in Equation 4.47, so $c_i(k)$ is within a bound:

$$(-2+R)\mu(1-\varrho) \leq c_i(k) \leq (2+R)\mu(1+\varrho), \tag{4.58}$$

which means that

$$\begin{aligned}
\delta(c(k)) &\leq \max_{i_1,i_2} |c_{i_1}(k) - c_{i_2}(k)| && (4.59)\\
&\leq (2+R)\mu(1+\varrho) - (-2+R)\mu(1-\varrho) && (4.60)\\
&= \left(4 + 2\varrho R\right)\mu && (4.61)
\end{aligned}$$

□

Theorem 1. *The clock skew $\delta\left(k\right)$ is within a bound:*

$$\delta\left(k\right) \leq \left(1 - \gamma + \frac{4\varrho}{1-\varrho}\right)^k \delta(0) + \left(4 + 2\varrho R\right)\mu\frac{1-(1-\gamma)^k}{\gamma} \tag{4.62}$$

Proof. Transforming Equation 4.44 to precision domain using Lemma 6 and Lemma 7, we get

$$\delta\left(k+1\right) \leq \left(\delta\left(A(k)\right) + \frac{4\varrho}{1-\varrho}\right)\delta(k) + \delta\left(c(k)\right), \tag{4.63}$$

Expanding the right-hand side of the equation gives:

$$\begin{aligned}\delta(k) &\leq \prod_{m=0}^{k-1}\left(\delta\left(A(k)\right)+\frac{4\varrho}{1-\varrho}\right)\delta(0)+\sum_{\ell=0}^{k-1}\left(\prod_{m=\ell}^{k-1}\delta\left(A(m)\right)\delta\left(c(k)\right)\right)\\ &=\left(1-\gamma+\frac{4\varrho}{1-\varrho}\right)^{k}\delta(0)+(4+2\varrho R)\,\mu\sum_{\ell=0}^{k-1}(1-\gamma)^{k-\ell}\\ &\leq\left(1-\gamma+\frac{4\varrho}{1-\varrho}\right)^{k}\delta(0)+(4+2\varrho R)\,\mu\frac{1-(1-\gamma)^{k}}{\gamma},\end{aligned}\tag{4.64}$$

□

As a sanity check, by assuming $\gamma > \frac{4\varrho}{1-\varrho}$, we can let $k \to \infty$ and find

$$\lim_{k\to\infty}\delta(k)\leq(4+2\varrho R)\,\frac{\mu}{\gamma}.\tag{4.65}$$

4.3.2 Convergence Analysis for OCDC Algorithm

In previous section the converge property is analyzed for corr_i^1. In this section the converge property of the OCDC algorithm with both corr_i^1 and corr_i^2 is also analyzed. We start by a slightly different relation between $\tau_i(k+1)$ and $\tau_i(k)$ due to the additional correction corr_i^2.

Lemma 8. *With $corr_2(k)$, Equation 4.38 becomes*

$$\tau_i(k+1)-\tau_i(k)=\frac{1}{|\text{In}_i(k)|}\sum_{j\in\text{In}_i(k)}\left(\tau_j^i(k)-\tau_i(k)\right)\frac{k+1}{k}.\tag{4.66}$$

Lemma 9. *Lemma 8 can be reformulated with matrix form*

$$t(k+1)=A(k)t(k)+b(k)+c(k),\tag{4.67}$$

where

$$A_{i,j}(k) = \begin{cases} 1/|\mathrm{In}_i(k)| & \text{if } j \in \mathrm{In}_i(k) \\ 0 & \text{else .} \end{cases} \tag{4.68}$$

and

$$b_i(k) = \left(\frac{(k+1)\beta_i(k)}{k\overline{\beta_j^i}(k)} - 1 \right) \left(\left(A(k)t(k) \right)_i - t_i(k) \right) \tag{4.69}$$

and

$$c_i(k) = \left(-\alpha_i(k) + \frac{k+1}{k}\overline{\alpha_j^i}(k) \right) \beta(k) \tag{4.70}$$

with the existence of some $\overline{\beta_j^i}(k) \in [\mu(1-\varrho), \mu(1+\varrho)]$ and $\overline{\alpha_j^i}(k) \in [-1, 1]$.

Proof. Substitute $\tau_j^i(k) - \tau_i(k)$ in Definition 8 using Lemma 1, we get □

$$\begin{aligned} \tau_i(k+1) - \tau_i(k) &= \frac{1}{|\mathrm{In}_i(k)|} \sum_{j \in \mathrm{In}_i(k)} \left(\tau_j^i(k) - \tau_i(k) \right) \frac{k+1}{k} \\ &= \frac{1}{|\mathrm{In}_i(k)|} \sum_{j \in \mathrm{In}_i(k)} \left(\left(t_j(k) - t_i(k) \right) / \beta_j^i(k) + \alpha_j^i(k) \right) \frac{k+1}{k} \end{aligned}$$

For the average, we get the existence of some $\overline{\beta_j^i}(k) \in [\mu(1-\varrho), \mu(1+\varrho)]$ and $\overline{\alpha_j^i}(k) \in [-1, 1]$ such that

$$\tau_i(k+1) - \tau_i(k) = \frac{1}{|\mathrm{In}_i(k)|\,\overline{\beta_j^i}} \sum_{j \in \mathrm{In}_i(k)} \left(t_j(k) - t_i(k) \right) \frac{k+1}{k} + \frac{k+1}{k}\overline{\alpha_j^i}(k) \tag{4.71}$$

Combining Lemma 2 and Equation 4.71 , we get

$$\left(t_i(k+1) - t_i(k) \right) = \frac{\beta_i(k)}{|\mathrm{In}_i(k)|\,\overline{\beta_j^i}(k)} \sum_{j \in \mathrm{In}_i(k)} \left(t_j(k) - t_i(k) \right) \frac{k+1}{k} + \left(-\alpha_i(k) + \frac{k+1}{k}\overline{\alpha_j^i}(k) \right) \beta_i(k)$$

Putting $\gamma_i(k) = \frac{(k+1)\beta_i(k)}{k\overline{\beta_j^i}(k)} - 1$ and adding $t_i(k)$ on both sides of the equation, we get

$$\begin{aligned} t_i(k+1) &= \frac{1}{|\mathrm{In}_i(k)|}\sum_{j\in\mathrm{In}_i(k)} t_j(k) + \frac{\gamma_i(k)}{|\mathrm{In}_i(k)|}\sum_{j\in\mathrm{In}_i(k)}\big(t_j(k) - t_i(k)\big) + \left(-\alpha_i(k) + \frac{k+1}{k}\overline{\alpha_j^i}(k)\right)\beta_i(k) \\ &= \big(A(k)t(k)\big)_i + \gamma_i(k)\Big(\big(A(k)t(k)\big)_i - t_i(k)\Big) + \left(-\alpha_i(k) + \frac{k+1}{k}\overline{\alpha_j^i}(k)\right)\beta_i(k) \end{aligned} \tag{4.72}$$

or in vector form as Equation 4.67.

Lemma 10. *The variable $b(k)$ has the following constraint:*

$$\delta\big(b(k)\big) \le \frac{4\varrho}{1-\varrho}\frac{k+1}{k}\delta(k). \tag{4.73}$$

Proof. With $\delta(k)$ is the maximum difference of pulses in the same round, we have

$$\delta\left(\big(A(k)t(k)\big)_i - t_i(k)\right) \le \delta(k). \tag{4.74}$$

The bound of $\gamma_i = \frac{(k+1)\beta_i(k)}{k\overline{\beta_j^i}(k)} - 1$ is

$$-\frac{2\varrho}{1+\varrho}\frac{k+1}{k} \le \gamma_i(k) \le \frac{2\varrho}{1-\varrho}\frac{k+1}{k}, \tag{4.75}$$

Considering Equation 4.74 and Equation 4.75 with Equation 4.46, we get

$$-\frac{2\varrho}{1-\varrho}\frac{k+1}{k}\delta\left(k\right) \le b_i(k) \le \frac{2\varrho}{1-\varrho}\frac{k+1}{k}\delta\left(k\right), \tag{4.76}$$

which means that

$$\delta\left(b(k)\right) \le \max_{i_1} b_{i_1}(k) - \min_{i_2} b_{i_2}(k) \le \frac{4\varrho}{1-\varrho}\frac{k+1}{k}\delta(k). \tag{4.77}$$

□

Lemma 11. *By assuming $R > 2$, the maximum difference of the vector $c(k)$ is*

$$\delta(c(k)) \leq \left(\frac{4k+2}{k}\right)\mu \tag{4.78}$$

Proof. If $R > 2$, we get $c_i(k) \geq 0$ in Equation 4.70, so $c_i(k)$ is within a bound:

$$(-1 - \frac{k+1}{k})\mu(1-\varrho) \leq c_i(k) \leq (1 + \frac{k+1}{k})\mu(1+\varrho), \tag{4.79}$$

which means that

$$\begin{aligned}
\delta(c(k)) &\leq \max_{i_1,i_2} |c_{i_1}(k) - c_{i_2}(k)| && (4.80)\\
&\leq (1 + \frac{k+1}{k})\mu(1+\varrho) - (-1 - \frac{k+1}{k})\mu(1-\varrho) && (4.81)\\
&= \left(\frac{4k+2}{k}\right)\mu && (4.82)
\end{aligned}$$

□

Theorem 2. *The clock skew $\delta(k)$ is within a bound:*

$$\delta(k) \leq \left(1 - \gamma + \frac{4\varrho}{1-\varrho}\frac{k+1}{k}\right)^k \delta(0) + (4 + 2\varrho R)\,\mu\frac{1-(1-\gamma)^k}{\gamma} \tag{4.83}$$

Proof. Transforming Equation 4.67 to precision domain using Lemma 10 and Lemma 11, we get

$$\delta(k+1) \leq \left(\delta(A(k)) + \frac{4\varrho}{1-\varrho}\frac{k+1}{k}\right)\delta(k) + \delta(c(k)), \tag{4.84}$$

Expanding the right-hand side of the equation gives:

$$\begin{aligned}\delta(k) &\leq \prod_{m=0}^{k-1}\left(\delta\left(A(k)\right)+\frac{4\varrho}{1-\varrho}\frac{k+1}{k}\right)\delta(0)+\sum_{\ell=0}^{k-1}\left(\prod_{m=\ell}^{k-1}\delta\left(A(m)\right)\delta\left(c(k)\right)\right)\\ &=\left(1-\gamma+\frac{4\varrho}{1-\varrho}\frac{k+1}{k}\right)^{k}\delta(0)+\frac{4k+2}{k}\mu\sum_{\ell=0}^{k-1}(1-\gamma)^{k-\ell}\\ &\leq\left(1-\gamma+\frac{4\varrho}{1-\varrho}\frac{k+1}{k}\right)^{k}\delta(0)+\frac{4k+2}{k}\mu\frac{1-(1-\gamma)^{k}}{\gamma},\end{aligned}\tag{4.85}$$

□

As a sanity check, by assuming $\gamma > \frac{4\varrho}{1-\varrho}\frac{k+1}{k}$, we can let $k \to \infty$ and find

$$\lim_{k\to\infty}\delta(k) \leq 4\frac{\mu}{\gamma},\tag{4.86}$$

which is much smaller than Equation 4.65.

[1]

4.4 Conclusion

In this chapter, a new pulse-coupled clock synchronization method is proposed (OCDC algorithm), which improves the pulse-coupled synchronization (OC algorithm) by introducing a time-wheel algorithm and drift compensation. The time-wheel algorithm enables sensors to exchange the information of the rounds during clock synchronization. Drift compensation applies an additional correction to clock skew. We also

[1] To obtain the Equation 4.86, the first term in Equation 4.85 is removed because $\lim_{k\to\infty}\left(1-\gamma+\frac{4\varrho}{1-\varrho}\frac{k+1}{k}\right)^{k} = 0$, where the term $1-\gamma+\frac{4\varrho}{1-\varrho}\frac{k+1}{k} < 0$ since we assume that $\gamma > \frac{4\varrho}{1-\varrho}\frac{k+1}{k}$. It is a reasonable assumption because ϱ is much smaller than 1, so the value of $\frac{4\varrho}{1-\varrho}\frac{k+1}{k}$ is small. Moreover, the term $\mu\frac{1-(1-\gamma)^{k}}{\gamma}$ in Equation 4.85 becomes $\frac{\mu}{\gamma}$ when k approach to infinity because $\lim_{k\to\infty}(1-\gamma)^{k} = 0$. The term $\frac{4k+2}{k}$ in Equation 4.85 becomes 4 since $\lim_{k\to\infty}\frac{4k}{k} = 4$ and $\lim_{k\to\infty}\frac{2}{k} = 0$. With these calculations, Equation 4.86 is obtained.

derive the convergence rate of both algorithms to show the improvements of the algorithm. In order the view the properties of our proposed algorithm, in the next chapter, simulations are performed to compare our proposed algorithm with the previous work Függer et al. [2015a].

Chapter 5

Simulation Results

5.1 Introduction

In this chapter, we perform several simulations to compare the proposed approach to its previous work [Függer et al., 2015a]. The simulation environment is written in Python3 and run on the MacOS operating system with 2 GHz Intel dual Core i7 processor and 8 gigabyte RAM. A different scenario is build to test the limit of the algorithm. At first, a simple environment is setup and simulated. We setup different environment to test the robustness of the algorithm. In the end, a dynamic real-world vehicular network is simulated to compare the algorithms.

5.2 Dense Scene

Figure 5.1 shows the layout of the simple environment, $n = 10$ sensors are randomly scattered. The average connectivity of scene $\frac{\|\mathcal{V}\|}{\|\mathcal{N}\|}$ is 5.2 per node. Each node has different sensing range to detect the pulses from neighboring sensors. Sensors can only sense the neighbors within the distance $r_i \in \mathbb{R}$, which is bounded within a range: $r(1-\epsilon) \leq r^i \leq r(1+\epsilon)$. The average of sensing range for all the sensors $r = 30$ and $\epsilon = 0.5$. The size of the environment is 50×50 m^2. We set $T_i(k) = R(1 + \sigma_i + \sigma_k)$

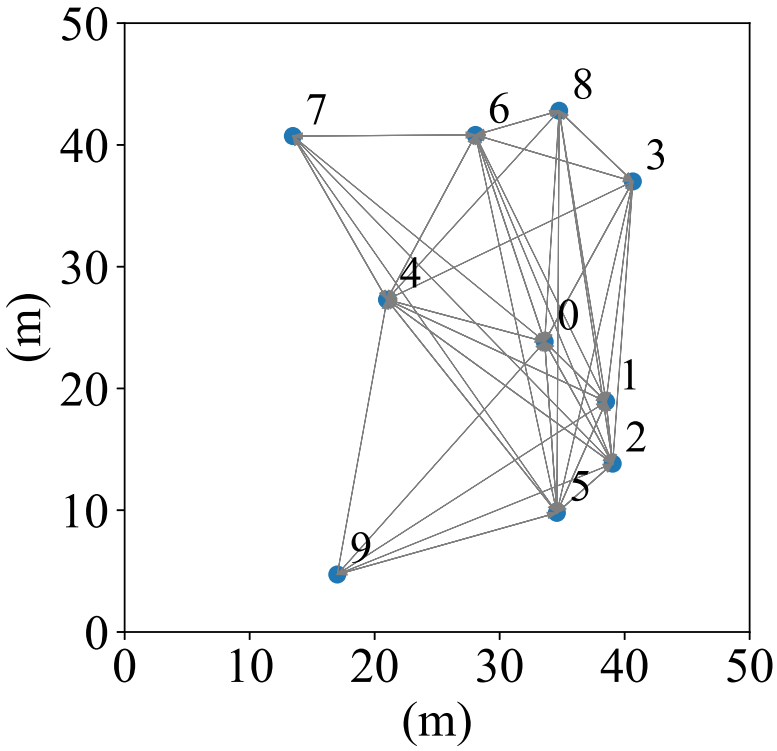


Figure 5.1: Scene in the experiment

with $|\sigma_i| \leq 30\%$ and $|\sigma_k| \leq 1\%$ and $R = 0.03$ second for different sensors. σ_i is a sensor varying variable and σ_k is a sensor and time varying variable. The variation σ_i is the different frequency for different sensor, which only be randomly chosen once when sensor is initialized. The variation σ_k is the different frequency in each round for each sensor, which affects the frequency of the clock every round. A realistic value of σ_k is $10^{-2}\%$ to $10^{-4}\%$ according to Mills [1998]. We setup all the sensors waking up at time 0 and starting to broadcast at time $T_i(0)$ for every sensor.

We compare three different algorithms in the same environment:

1. an algorithm without any correction

2. clock synchronization using only corr_i^1 (OC algorithm)

3. clock synchronization using $\text{corr}_i^1 + \text{corr}_i^2$ (OCDC algorithm)

The second algorithm is the previous work from Függer et al. [2015a], and the third is our proposed technique. We take corr_i^2 into account at the sixth round. Figure 5.2 and Figure 5.3 show the comparison of these three algorithms. Figure 5.2 shows the waveform of the pulses for the three algorithms. The x-axis indicates time in seconds, and the y-axis shows a value one whenever any pulse is broadcasted, otherwise it shows 0. The blue and red colors indicate the different round: $k = 0$ and $k = 1$. The thiner the width of a colored rectangle (blue or red) is, the more synchronized the pulses (of the various sensors) are, which means low clock skews. The clock skew of OCDC (the third figure of Figure 5.2) is much smaller than OC (the second figure of Figure 5.2). Figure 5.3 shows the timing of pulses for each round. Dots linked by lines with different colors indicate that the pulses are broadcasted by the same sensor. One can observe that if no clock synchronization technique is applied, the pulses for each round is diverging (the first figure in Figure 5.3). Pulses in the same round for algorithm corr_i^1 are successfully trapped within an interval, but the interval will not reduce as time goes on (the second figure in Figure 5.3). In contrast, pulses from the OCDC algorithm are broadcasted more orderly than OC algorithm as time goes on (the third figure in Figure 5.3).

Figure 5.4 shows the size of clock skew for each round. We calculate clock skew by considering the nearest pulse from all sensors in the same round k to the target sensor. Clock skew for first five rounds are the same, because OCDC algorithm does not enable corr_2 yet. As soon as corr_2 is enable, the clock skew reduces dramatically. The clock skew of OCDC algorithm is approximately 7 times smaller than the clock skew of OC algorithm after they are stable.

5.3 Clock Drift Analysis

In this experiment, we analyze the methods: OC and OCDC algorithm in situations that σ_i and σ_k is different. We set σ_i and σ_k at different values to verify the robustness

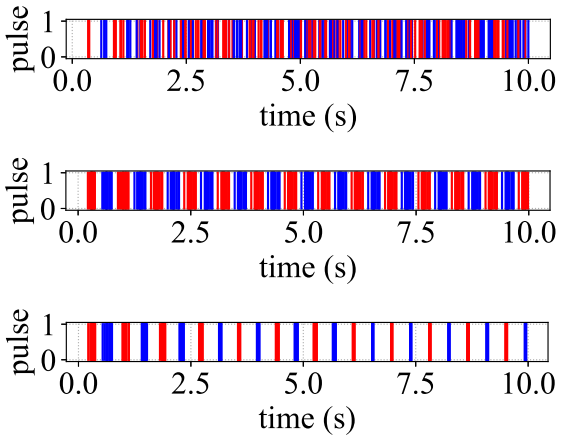


Figure 5.2: The figure of waveform of the pulses for all sensors. The x-axis indicates time in seconds and y-axis shows a value one whenever any pulse is broadcasted, otherwise it shows zero. The blue and red colors indicate the different round: $k = 0$ and $k = 1$. The first figure shows the waveform without any correction. The second figure shows the pulses are bounded in an interval thanks to corr_i^1. The last figure shows the pulses with much smaller clock skew thanks to both corr_i^1 and corr_i^2 .

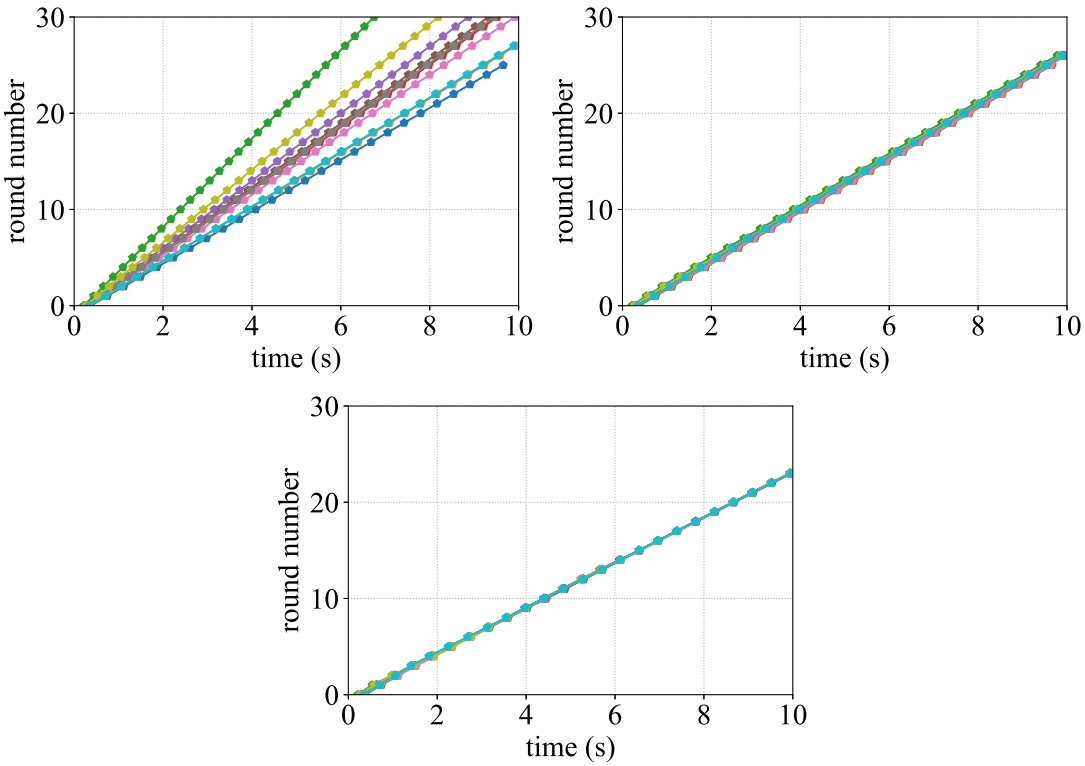


Figure 5.3: Timing of pulses at each round. This figure shows the timing of pulses for each round. Dots linked by lines with different colors indicate that the pulses is broadcasted by a same sensor. One can observe that if no clock synchronization technique is applied, the pulses for each round is diverging (column 1). Pulses in the same round for OC algorithm are successfully trapped within an interval, but the interval will not reduce as time goes on (column 2). In contrast, pulses from the OCDC algorithm are broadcasted more orderly than OC as time goes on (column 3).

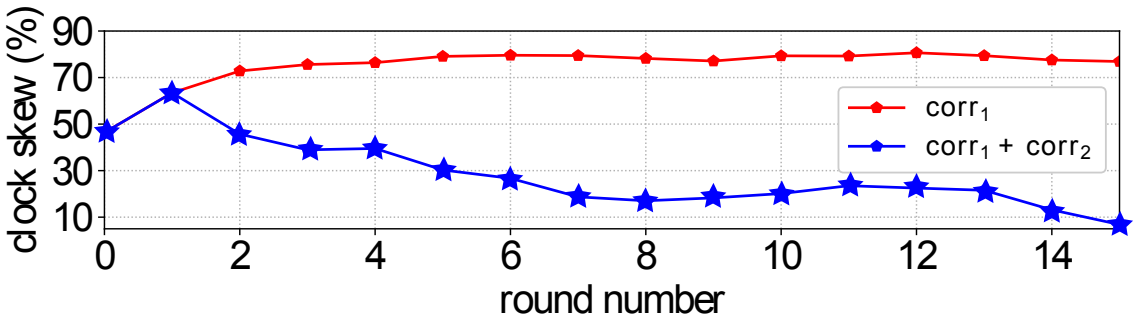
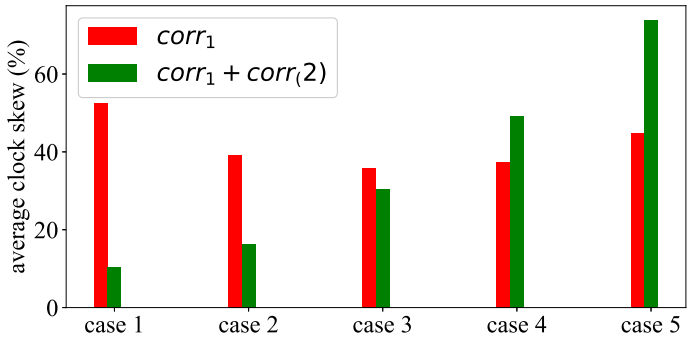


Figure 5.4: Clock skew comparison between OC algorithm (red) and OCDC algorithm (blue). The x-axis represents the round number and the y-axis represent the maximum clock skew of each round

of OC and OCDC algorithm. In order to test the edge cases, the total variation of $\sigma_i + \sigma_k$ is $\pm 20\%$ of the constant R. Figure 5.5 on the facing page shows the five different situations. The environment is the same except σ_i and σ_k are changed. In these five simulations, the error for OC algorithm becomes small when σ_i decrease, which shows that it can handle random uncertainty of clock drift σ_k better than OCDC algorithm. In contrast, the error for OCDC algorithm increases when σ_k increases. It shows that large amount of additional random clock drift has chance to let OCDC algorithm become worse than OC algorithm. Notice that we test the edge cases here, in normal case, σ_k is smaller than $10^{-4}\%$.

5.3.1 Sudden Change of Clock Frequency

Clock drift can suddenly change due to a sensor turning into power-saving mode. There we setup an extreme experiment where all sensors change their frequency in the environment with $\sigma_i = 30\%$, and $\sigma_k = 1\%$ during the 13th and 14th round in the simulation. Figure 5.6 shows the result of this experiment. Both of the methods suffer from the instant change of clock frequency at the 14-th round but converge afterward.



	case1	case2	case3	case4	case5
$\sigma_{i,\max}$	$\leq 20\%$	$\leq 15\%$	$\leq 10\%$	$\leq 5\%$	$\leq 0\%$
$\sigma_{k,\max}$	$\leq 0\%$	$\leq 5\%$	$\leq 10\%$	$\leq 15\%$	$\leq 20\%$

Figure 5.5: This figure shows the five different situations. The environment is exactly the same except σ_i and σ_k is changed. In these five simulations, the error for method corr_i^1 becomes small when the σ_i decreases, which shows that it can handle random uncertainty of clock drift σ_k better than $\text{corr}_i^1 + \text{corr}_i^2$. In contrast, the error for method $\text{corr}_i^1 + \text{corr}_i^2$ increase when σ_k increases. It shows that large amount of additional random clock drift has chance to let $\text{corr}_i^1 + \text{corr}_i^2$ become worse than corr_i^1.

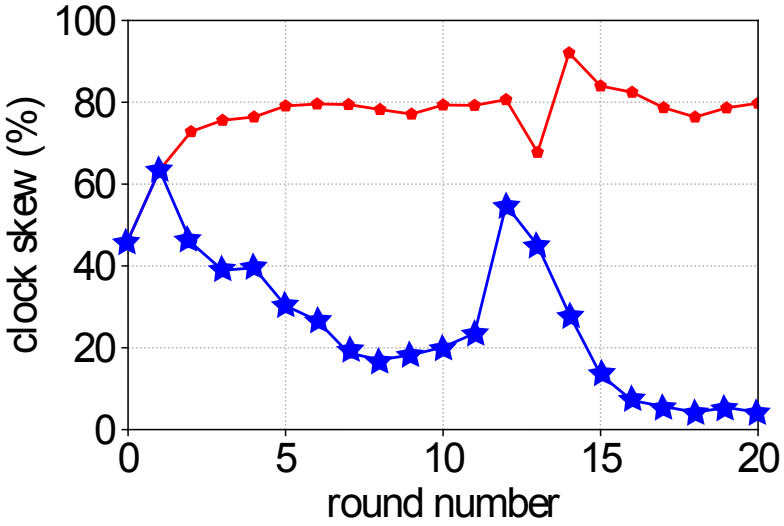


Figure 5.6: An extreme experiment where all sensors change their frequency in the environment with $\sigma_i = 30\%$, and $\sigma_k = 1\%$ during the 13th and 14th round in the simulation. If the connectivity remains zero, the pulses in the same round diverge through time if the clock skew is not zero.

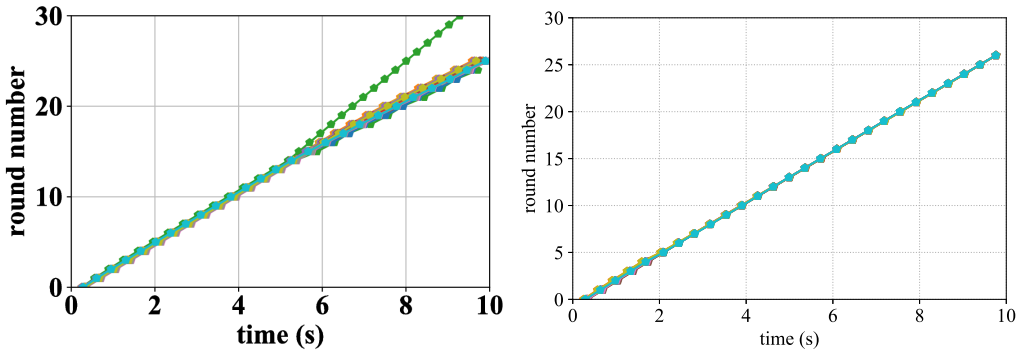


Figure 5.7: Experimental results of OC algorithm (left) and using OCDC algorithm (right). The figure on the left shows the round and the pulses using OC algorithm. There is one line diverge at the fifth second of simulation, which means that there is a sensor that does not follow the pace of the network. The figure on the right shows that with OCDC algorithm, all sensors remain stable even if the connectivity is zero because corr_i^2 modifies the local clock period according to local history data. It does not need to use the pulses from neighboring sensors to compute corr_i^2, so the round remains regular even if the network is changed. If the connectivity remains zero, the pulses in the same round diverge through time if the clock skew is not zero.

5.3.2 Sudden Change of a communication graph

In the wireless sensor network, sensors will not remain at one location, so the neighboring sensors will not be the same in different rounds. In this experiment, we want to discover the stability of the two methods OC and OCDC algorithm considering the changes of communication graph $G(k)$. In this environment, all the parameters are the same as in the first experiment. In order to test our algorithm under the extreme case, all the sensors are relocated suddenly, and communication graph $G(k)$ becomes different at the fifth second in the simulation.

Figure 5.7 shows the comparison of the two methods. Figure 5.7(left) shows the round and the pulses using OC algorithm. There is one line diverge at the fifth second of simulation, which means that there is a sensor that does not follow the pace of the network. Figure 5.7 (right) shows that with OCDC algorithm, all sensors remain stable even if the connectivity is 0, because corr_i^2 modify the local clock period

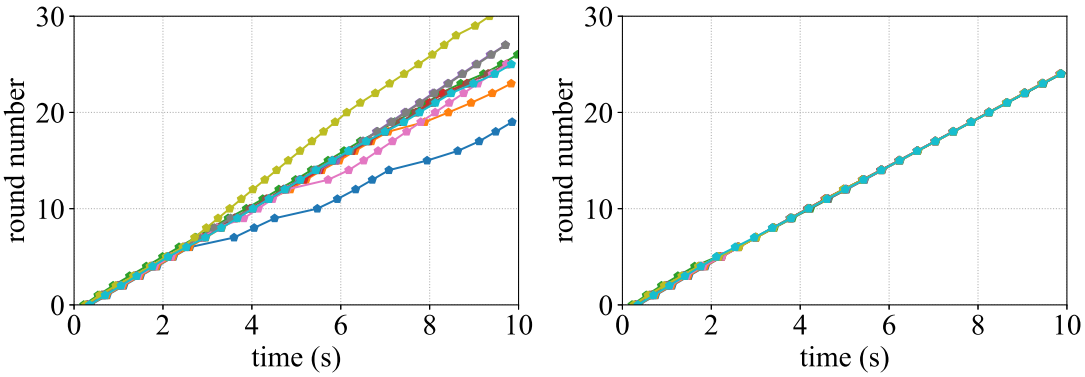


Figure 5.8: Time of each round of a random walk sensor network. The left figure shows the round and the pulses using OC algorithm. We discover that some sensors may have few or 0 neighboring node. In that case, corr_1 is difficult to correct the timing of pulses, because corr_1 only depend on the pulses from neighboring sensors. The right figure shows the round and the pulses using OCDC algorithm. It is obvious that the pulses are aligned.

according to local history data. It does not need to use the pulses from neighboring sensors to compute corr_i^2, so the round remains regular even if the network is changed.

5.3.3 Random walk communication graph

We further analysis a rational environment to approximate the movement of the animals. In the environment, sensors are initially setup as Figure 5.1 but start to random walk with a speed $s \in [0, 2]$ m/s. Figure 5.8 shows the comparison of the two methods. Figure 5.8 (left) shows the round and the pulses using only corr_1. We discover that some sensors may have few or zero neighboring node. In that case, corr_1 is difficult to correct the timing of pulses, because corr_1 only depends on the pulses from neighboring sensors. Figure 5.8 (right) shows the round and the pulses using both OC and OCDC algorithm. It is obvious that the pulses are aligned. That is because the calculation of corr_2 only involve the pulses of the sensor itself, which is accessible even if the sensor has no neighbor.

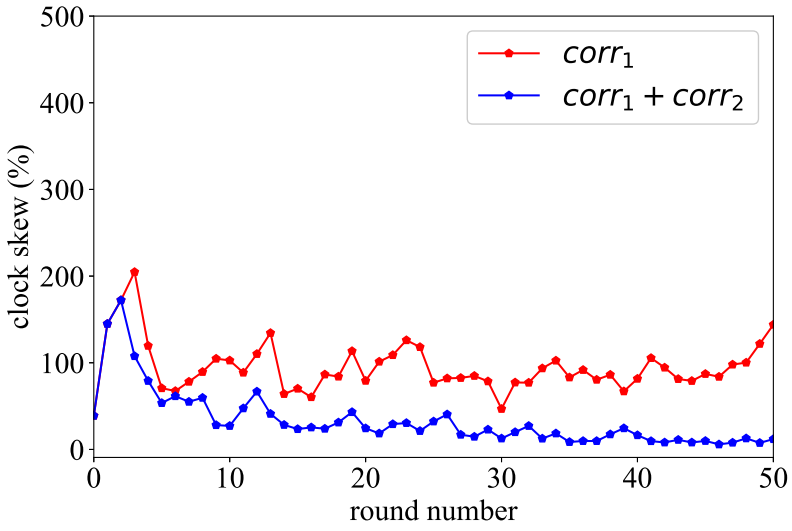


Figure 5.9: Experiment in harsh environment. A comparison between pulses broadcasted using corr_i^1 (upper curve) and using $\text{corr}_i^1 + \text{corr}_i^2$ (lower curve). The y-axis represents the value of the maximum clock skew during the simulation.

5.4 Extreme Environment

In the simulation, to drive our algorithm to the extreme, 50 sensors are randomly spread in a 217×217 m^2 environment. The connection for every sensor is distance $r = 15 \pm 7.5$ m^2. Moreover, the sensors are dynamic and randomly relocated every 0.05 second. We set $k_{max} = 3$ and $\sigma_i = 20\%$ and $\sigma_k = 1\%$. Figure 5.9 shows the convergence result, where $\text{corr}_1 + \text{corr}_2$ not only converge faster but also have much smaller clock skew then corr_1.

5.5 Vehicular Network

Vehicular network is a categories of wireless sensor network, where vehicles communicate to each other to perform data fusion. With vehicular network, the vehicles are able to corporate to increase the road safety or improve the traffic. We simulate

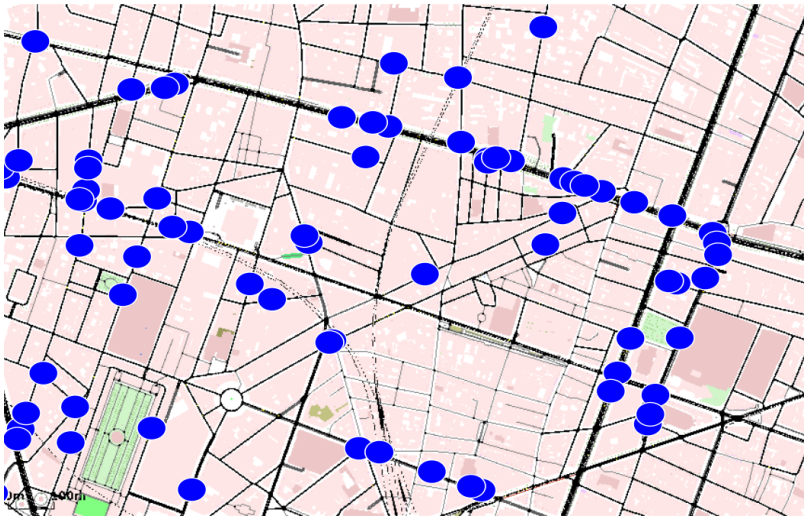


Figure 5.10: Static vehicle network simulation scenario. The width and height of the scenario is 1500 and 1000 meters. The pink squares represent the blocks. Black lines represent the road, and the blue circles are the position of the vehicles. The green area is the Jardin du Palais Royal.

a vehicle network thanks to the Simulation of Urban MObility (SUMO) [Behrisch et al., 2011]. The simulation environment is the 2nd district of Paris, downloaded from OpenStreetMap [OpenStreetMap contributors, 2017].

5.5.1 Static Vehicle Network

Figure 5.10 shows the simulation environment, where $n = 75$ vehicles are randomly placed with an average number of 2.84 neighbors per vehicle. The average sensing range for all the vehicles is $r = 300$ (m) and sensing range variation $\epsilon = 0.2$. Sensing range for each sensor node i is bounded

$$(1 - \epsilon)r \leq r_i \leq (1 + \epsilon)r. \tag{5.1}$$

The number of slots of time wheel is $s_{\max} = 2$. Figure 5.11 shows the vehicle nodes location and their connectivity. The numbers beside the nodes represent the identity

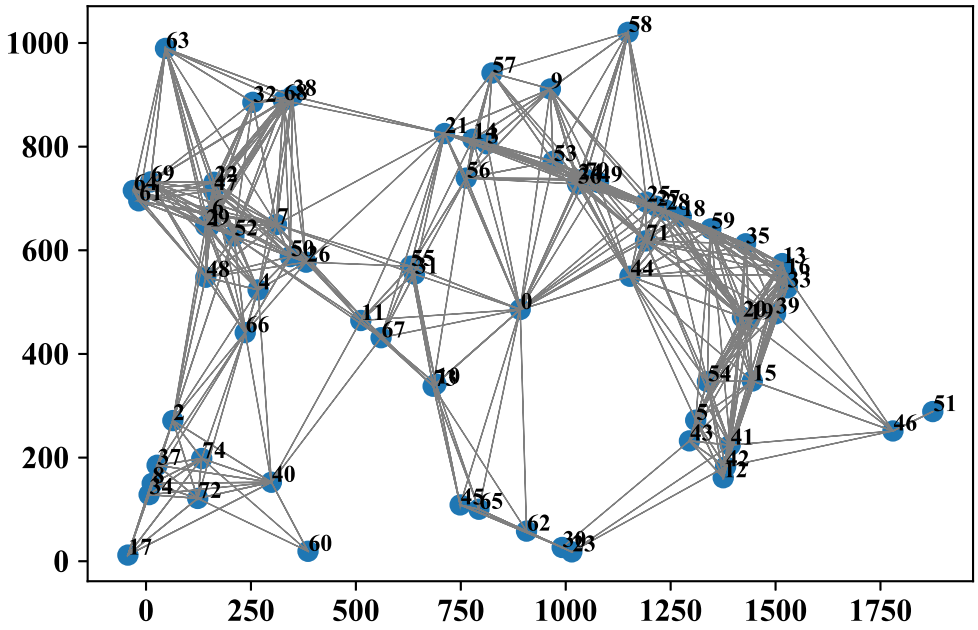


Figure 5.11: Static vehicle network, where the x-axis and the y-axis is the location represented in meters.

of the vehicles. The parameters in this simulation are listed as follows:

- $n = 75$ vehicles
- Low connectivity $c = 2.84$ (nodes in average)
- sensing range $r = 300$ (m)
- clock drift $\varrho = 0.2$
- maximum round number $s_{\max}$ is set to 2
- sensing range variation $\epsilon = 0.2$

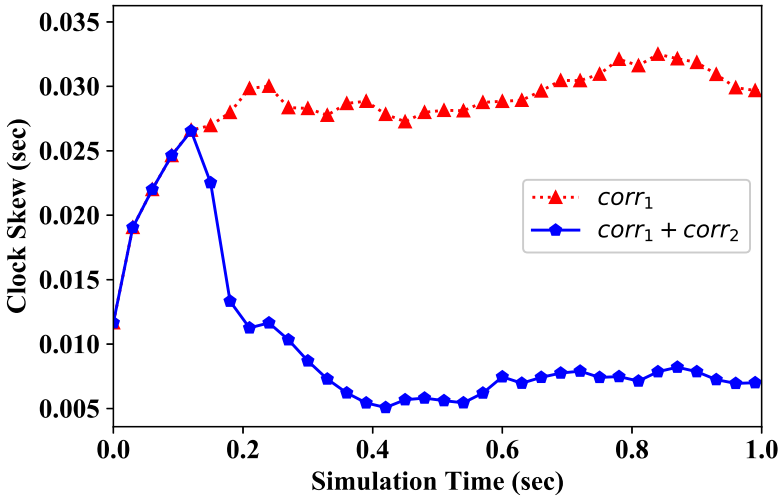


Figure 5.12: Clock skew comparison between OC (red) and OCDC (blue) algorithm. Clock skew for the first five rounds are the same because OCDC algorithm includes the term $\text{corr}_2 = 0$. As soon as corr_2 is enabled, the clock skew reduces significantly. The clock skew of OCDC algorithm is approximately 7 times smaller than the clock skew of OC algorithm after both clock skew are stabilized.

We compare the two different algorithms in the same environment:

1. OC algorithm
2. OCDC algorithm

The first algorithm is a pulse synchronization algorithm considering a highly dynamic network published by Függer et al. [2015b] and the second is our proposed technique. We take corr_i^2 into account after the sixth round. Figure 5.12 shows the comparison of these algorithms. The OC algorithm (Figure 5.12 red) successfully converges all the pulses in the same round within an interval, but the interval will not reduce as time goes on (Figure 5.12 blue). In contrast, pulses from the OCDC algorithm are broadcasted more orderly than corr_i^1 as time goes on.

Figure 5.12 shows the clock skew for each round. Clock skew for first five rounds are the same, because OCDC algorithm includes the term $\text{corr}_2 = 0$. As soon as corr_2

is enabled, the clock skew reduces significantly. The clock skew of OCDC algorithm is approximately seven times smaller than the clock skew of OC algorithm after both clock skews are stabilized.

5.5.2 Pulse Frequency and Clock Skew

In this section, different frequencies of the pulse are considered. Same parameters as the previous section are used. If the vehicles exchange pulses more frequently, then the clock skew temp to converge faster. Fig Figure 5.13 illustrates four simulations where the scene is setup the same as Fig Figure 5.10 except $T(k)$, the default time period between two pulses, which is different in the four simulations. If $T(k)$ is smaller, then the vehicle network converges faster then when $T(k)$ is larger. However, higher pulses frequently cost higher energy consumption considering the same time period. The parameter $T(k)$ can be chosen to balance the convergence rate and the energy consumption. In the vehicle network, all pulses in the same round converge regardless $T(k)$. There is also a possibility to slow down the pulse exchange after the clock synchronization has converged.

5.5.3 Moving Vehicle Network

Vehicles are initially setup as Figure 5.10. The acceleration and the deceleration of each vehicle is 2.6 (m/s^2) and 4.5 (m/s^2), respectively, which is the default parameters in SUMO simulator. Vehicles slow down when they turn and stop when traffic lights are red. There are 214 traffic lights on the map. The realist vehicle traces are generated by SUMO.

Figure 5.14 shows the clock skew of OC and OCDC algorithm in the simulation. The clock skew is initialized at 0.07 seconds and drops efficiently due to the correction term corr_i^1. The communication graph is fully connected at the beginning, so the clock skew is smaller after few seconds. During the simulation, some mobile vehicle has no or few neighbors. In that case, the clock offset and drift cannot converge, because there is no communication between the networks, so the clock skew starts

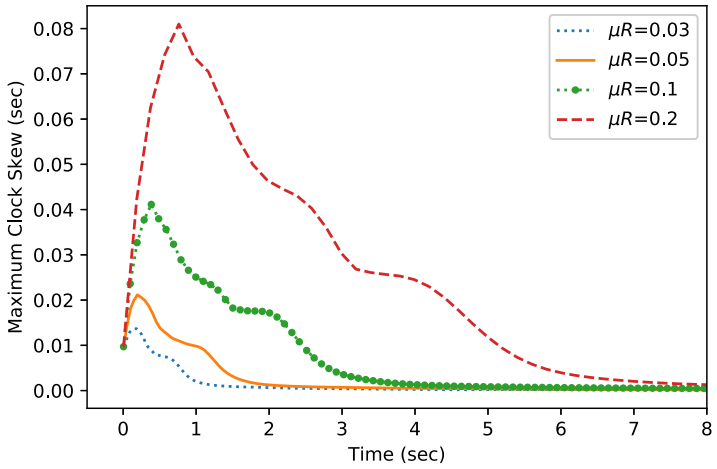


Figure 5.13: Simulation result for different frequency of pulse. If $T(k)$ is smaller, then the vehicle network converges faster then when $T(k)$ is larger.

to increase. For OC algorithm, the frequency of the pulses reverts to its original frequency, causing relatively large clock skew. For OCDC algorithm, the clock skew increase slowly due to unconnected topology of the communication graph. When the nodes in the network are connected, pulses are re-synchronized, so the clock skew drop. Overall OCDC algorithm has better performance than OC algorithm due to the additional correction of corr_i^2, which is more robust in the dynamic vehicle network.

5.6 Conclusion

In this chapter, the proposed clock synchronization is simulated and compared with the previous work of Függer et al. [2015a]. From the simulation, it is clear to observe that the clock skew is reduced dramatically compared to the previous work. A random walk environment is simulated, which shows that with corr_i^2, sensors are able to remember the corrected frequency even if they do not have any neighboring

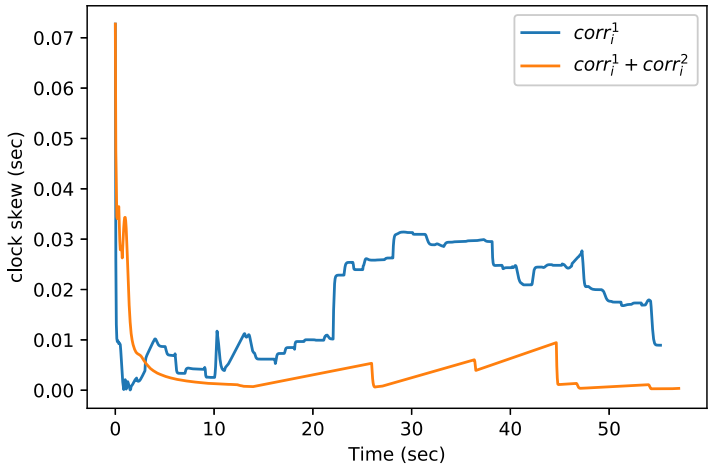


Figure 5.14: Time of each round of a dynamic vehicle network for 2 seconds. The clock skew is initialized at 0.07 seconds and drop efficiently due to the correction term corr_i^1. The communication graph is fully connected at the beginning, so the clock skew is smaller after few seconds. During the simulation, some mobile vehicle has no or few neighbors. In that case, the clock offset and drift cannot converge, because there is no communication between the networks, so the clock skew starts to increase. For OC algorithm, the frequency of the pulses reverts to its original frequency, causing relatively large clock skew. For the OCDC algorithm, the clock skew increase slowly due to the unconnected topology of the communication graph.

sensors. Two different types of clock drift are also analyzed to show the performance of the corrections. A vehicle network is also simulated to show the benefit of adding corr_i^2.

Chapter 6

Conclusion

This chapter summarizes the principal contributions of the first part of this thesis. Suggestions for further work in some of the subjects encompassed by the research contained in this thesis are also included. At the end of the chapter, a list of related publications is provided.

6.1 Conclusion and Contributions

We proposed a new pulse-coupled synchronization by introducing a time-wheel algorithm and drift compensation. The time-wheel algorithm enables sensors to exchange the information of the rounds during clock synchronization. Drift compensation applies an additional correction to clock skew. The simulation results show that the proposed algorithm performs better than the compared algorithm, especially when considering a highly dynamic wireless sensor network. The proposed algorithm is successfully applied to the vehicular network with a real geometric location and dynamic vehicles.

This thesis represents a step forward in research into pulse-coupled clock synchronization. The second correction reduces the clock skew of the sensors in a natural and elegant way. Moreover, using a comparison with the state-of-the-art technique,

it was proved that the proposed algorithm was more precise in most of the cases. This work is published on the 29th IEEE Intelligent Vehicle Symposium Han et al. [2018b].

6.2 Future Works

For future work, we intend to evaluate the algorithm using a practical wireless sensor network. The next step will be to select an appropriate communication hardware and to closely simulate it in a large scale network.

Part II

Guaranteed confidence region characterization for source localization using LSCR

Chapter 7

Introduction

Wireless sensor networks (WSNs) play an important role in applications such as environmental monitoring, source tracking, *etc.* In WSNs, the locations of sensing nodes are often required to make the collected information useful. Node localization algorithms have thus been developed, usually relying on a number of anchor nodes, whose locations are known. Localization is mainly performed considering the received signal strength (RSS), the time of arrival (TOA), the time difference of arrival (TDOA) of electromagnetic signals transmitted by a node to the anchor nodes or by the anchor nodes to the node to localize, see [Gezici, 2008, Li et al., 2016, Iliev and Paprotny, 2015].

In this chapter, one considers guaranteed confidence region characterization for the location estimate of nodes using RSS measurements. In most of the papers, location estimates are evaluated first, and approximate characterization of confidence regions are then provided. Here, one evaluates first accurate confidence regions with prescribed confidence levels. Estimates may then be obtained using classical techniques, or may be evaluated as the barycenter of the confidence region. For that purpose, the *Leave-out Sign-dominant Correlated Regions* (LSCR) [Campi and Weyer, 2005] approach is employed to define a region in the parameter space guaranteed to contain the true value of the parameter with an arbitrary confidence level. Confidence regions as defined by LSCR are valid even in the non-asymptotic regime,

i.e., when only a small set of measurements is available.

7.1 Motivation

In most of the papers, the RSS measurement noise is assumed to be normal or log-normal [Gholami et al., 2011, Vaghefi et al., 2013]. This allows one to apply classical least-squares (LS), maximum likelihood (ML), or maximum *a posteriori* (MAP) estimation techniques. The main difficulty is that the cost function associated to the LS, ML, or MAP estimation problem may be non-convex, especially when the path-loss exponent as well as the reference transmission power have to be estimated jointly with the location. The differential RSS measurement localization approach proposed in Wang et al. [2013]. With the differential RSS, one avoids estimating the reference power and facilitates estimation, *e.g.*, via the total least-squares approach considered in Rahimi et al. [2016]. Convex [Tomic et al., 2015] or semi-definite [Kim et al., 2007, Lee and Buehrer, 2009, Wang and Yang, 2009], [Ouyang et al., 2010, Vaghefi et al., 2013] relaxation techniques have been proposed, to facilitate the optimization. The non-linear localization problem is relaxed to a convex problem and solved by the semi-definite programming. Using a semi-definite relaxation, [Lohrasbipeydeh et al., 2014] presents a minimax approach involving again the differential RSS measurements. Minmax approach iteratively minimize the possible loss for a worst case senario. In most of these approaches, the estimator confidence region is then defined using the Cramér-Rao lower bound (CRLB) [Torrieri, 1990]. Nevertheless, confidence regions obtained from CRLB are usually optimistic and provide a relatively inaccurate confidence region when there are few measurements. Alternative bounded-error estimation techniques have been proposed in Mourad et al. [2009], Léger and Kieffer [2010], Mourad et al. [2011a] to get *guaranteed set* estimates. In these latter approaches, the RSS measurement errors are assumed to be bounded with known bounds. One tries to evaluate the set of all source locations that are consistent with the measurement model and the noise bounds. The resulting set is guaranteed to contain the true location of the source node, provided that the hy-

potheses on the noise bounds are satisfied. Nevertheless, in practice, noise samples do not really follow a normal or lognormal distribution, and the bounds considered in bounded-error estimation are either often violated because they are too small and result in an empty set estimate, or lead to huge solution sets when the bounds are too large.

7.2 Leave-out Sign-dominant Correlated Regions

LSCR belongs to a family of recently-proposed techniques to define non-asymptotic confidence regions (NACR) [Campi and Weyer, 2005, 2006, 2010, Csáji et al., 2012]. The main assumptions of LSCR are (i) that the system output corresponds to the model output for a given true value of the parameter vector (ii) that the measurement noise samples are independent and symmetrically distributed. Considering a given confidence level, and a finite set of measurements, LSCR defines a subset of the parameter space that is *guaranteed* to contain the true value of the parameter vector with a probability equal to the confidence level. Considering relatively mild conditions on the measurement noise, one is thus able to obtain guaranteed confidence regions. Nevertheless, the confidence region as defined by LSCR may be non-convex or even may consist of several disconnected components. As shown in Kieffer and Walter [2014], using tools from interval analysis [Jaulin et al., 2001], one is able to obtain inner and outer approximations of the confidence regions as defined by LSCR using subpavings, *i.e.*, set of non-overlapping interval vectors.

7.3 Outline

In Chapter 8, tools from Interval analysis is described, which is used by some of the localization methods. With the knowledge of Interval analysis, chapter 9 introduces RSS localization model and specify several localization approaches. In chapter 10, the methodology used in this work is shown and compared with other localization

approaches. The different correlation functions of LSCR are also simulated and analyzed. Finally, the conclusions of the survey is given in Chapter 12.

Chapter 8

Interval Analysis

To describe the problem formulation and the state-of-the-art solutions, it is necessary to understand the powerful tool, interval analysis, which is used in several localization approaches. Interval analysis help speed up the process for finding the confidence region. This chapter describes the fundamental concept and the tools in interval analysis.

8.1 Introduction

Interval analysis is a branch of mathematics considering intervals instead of numbers. It allows us to put bounds on rounding and measurements errors, thus produces reliable computed results. The guaranteed numerical methods are powerful tools for approximating solution sets. Guaranteed means outer and inner approximations of the sets of interest are obtained. It is possible for interval methods to characterize a set of solutions for a problem described by non-linear equations and inequalities. The main idea to be used for interval analysis is to enclose real numbers in intervals and vectors in boxes. One of the benefits is that it is possible to obtain guaranteed results directly using interval variables.

8.2 Interval

The interval $[x]$ is a connected subset of $\mathbb{R}$. Even when the interval is not closed, we shall keep the notation $[x]$. Whether the empty set $\emptyset$ should also be considered as an interval is still a subject of discussion. The lower bound $\text{lb}([x])$ of an interval $[x]$, also denoted by $\underline{x}$, is defined as

$$\underline{x} = \text{lb}[x] \triangleq \sup\{a \in [x] | \forall x \in [x], a \leq x\} \tag{8.1}$$

Its upper bound $\text{ub}([x])$, also denoted by $\bar{x}$, is defined as

$$\bar{x} = \text{ub}[x] \triangleq \inf\{b \in [x] | \forall x \in [x], x \leq b\} \tag{8.2}$$

Thus, $\underline{x}$ is the largest number on the left of $[x]$ and $\bar{x}$ is the smallest number on its right. The width of an interval $[x]$ is

$$w([x]) \triangleq \bar{x} - \underline{x} \tag{8.3}$$

and the centre of any bounded and non-empty interval $[x]$ is defined as

$$\text{mid}([x]) \triangleq \frac{\bar{x} - \underline{x}}{2}. \tag{8.4}$$

Also, the set-theoretic operations can be applied to intervals. The intersection of two intervals $[x]$ and $[y]$ is defined by

$$[x] \cap [y] \triangleq \{z \in \mathbb{R} | z \in [x] \text{ and } z \in [y]\}, \tag{8.5}$$

and the union of two intervals $[x]$ and $[y]$ is defined by

$$[x] \cup [y] \triangleq \{z \in \mathbb{R} | z \in [x] \text{ or } z \in [y]\}. \tag{8.6}$$

There are four classical operations of real arithmetic, namely addition $(+)$, subtraction $(-)$, multiplication $(*)$ and division $(/)$. All of them can be extended to

intervals as follows:

$$\begin{aligned}
[x]+[y] &= [\underline{x}+\underline{y}, \bar{x}+\bar{y}] \\
[x]-[y] &= [\underline{x}-\bar{y}, \bar{x}-\underline{y}] \\
[x]*[y] &= [\min(\underline{x}*\underline{y}, \underline{x}*\bar{y}, \bar{x}*\underline{y}, \bar{x}*\bar{y})] \\
[x]/[y] &= [x]*(1/[y]).
\end{aligned} \tag{8.7}$$

An interval vector $[\mathbf{x}]$ is a subset of $\mathbb{R}^n$ that can be defined as the Cartesian product of n closed intervals. An interval vector $[\mathbf{x}]$ is also called a box, which is written as $[\mathbf{x}] = [x_1] \times [x_2] \times ... \times [x_n]$, with $[x_i] = [\underline{x}, \bar{x}]$ for $i = 1, ..., n$. Its ith interval components $[x_i]$ is the projection of $[\mathbf{x}]$ onto the ith axis. Many of the operations extend without difficulty to boxes.

$$\begin{aligned}
\underline{\mathbf{x}} &= \mathrm{lb}(\underline{\mathrm{x}}) \triangleq (\mathrm{lb}(x_1), \mathrm{lb}(x_2), ..., \mathrm{lb}(x_3))^T \\
\bar{\mathbf{x}} &= \mathrm{ub}(\underline{\mathrm{x}}) \triangleq (\mathrm{ub}(x_1), \mathrm{ub}(x_2), ..., \mathrm{ub}(x_3))^T \\
[\mathbf{x}]+[\mathbf{y}] &= ([x_1]+[y_1]) \times ([x_2]+[y_2]) \times ... \times ([x_n]+[y_n]) \\
[\mathbf{x}]-[\mathbf{y}] &= ([x_1]-[y_1]) \times ([x_2]-[y_2]) \times ... \times ([x_n]-[y_n]) \\
[\mathbf{x}]^T * [\mathbf{y}] &= [x_1]*[y_1] + [x_2]*[y_2] + ... + [x_n]*[y_n] \\
\mathrm{mid}([\mathbf{x}]) &\triangleq (\mathrm{mid}(x_1), \mathrm{mid}(x_2), ..., \mathrm{mid}(x_n))^T \\
w([\mathbf{x}]) &\triangleq \max_{1 \le i \le n} w([x_i])
\end{aligned} \tag{8.8}$$

Let $\mathbb{R}^{m \times n}$ be the set of all matrix with real coefficients, m rows and n columns. A $m \times n$ dimensional interval matrix is a subset of $\mathbb{R}^{m \times n}$ that can be defined as the Cartesian product of $m \times n$ closed interval matrix. The interval matrix $[\boldsymbol{A}]$ is written in the following form:

$$[\mathbf{A}] = \begin{pmatrix} [a_{11}] & \dots & [a_{1n}] \\ \vdots & & \vdots \\ [a_{m1}] & \dots & [a_{mn}] \end{pmatrix}, \tag{8.9}$$

Many of the operations can be extended.

$$\begin{aligned}
\underline{\mathrm{A}} &= \mathrm{lb}([\mathbf{A}]) \triangleq \begin{pmatrix} [\underline{\mathrm{a}}_{11}] & \dots & [\underline{\mathrm{a}}_{1n}] \\ \vdots & & \vdots \\ [\underline{\mathrm{a}}_{m1}] & \dots & [\underline{\mathrm{a}}_{mn}] \end{pmatrix} \\
\bar{\mathbf{A}} &= \mathrm{ub}([\mathbf{A}]) \triangleq \begin{pmatrix} [\bar{a}_{11}] & \dots & [\bar{a}_{1n}] \\ \vdots & & \vdots \\ [\bar{a}_{m1}] & \dots & [\bar{a}_{mn}] \end{pmatrix} \\
[\mathbf{A}] + [\mathbf{B}] &= \begin{pmatrix} [a_{11} + b_{11}] & \dots & [a_{1n} + b_{1n}] \\ \vdots & & \vdots \\ [a_{m1} + b_{m1}] & \dots & [a_{mn} + b_{nm}] \end{pmatrix} \\
[\mathbf{A}] - [\mathbf{B}] &= \begin{pmatrix} [a_{11} - b_{11}] & \dots & [a_{1n} - b_{1n}] \\ \vdots & & \vdots \\ [a_{m1} - b_{m1}] & \dots & [a_{mn} - b_{nm}] \end{pmatrix} \\
[\mathbf{A}] * [\mathbf{x}] &= \left(\sum_{j=1}^{n} [a_{ij}] * [x_j] \right)_{1 \le i \le m} \\
\mathrm{mid}([\mathbf{A}]) &\triangleq (\mathrm{mid}([a_{ij}]))_{1 \le i \le m, 1 \le j \le n} \\
w([\mathbf{A}]) &\triangleq \max_{1 \le i \le m, 1 \le j \le n} w([a_{ij}])
\end{aligned} \tag{8.10}$$

8.3 Inclusion Function

Inclusion functions are useful to evaluate the upper and lower bound of a function with simple calculations. Consider a function $\mathbf{f}$ from $\mathbb{R}^n$ to $\mathbb{R}^m$. The inclusion

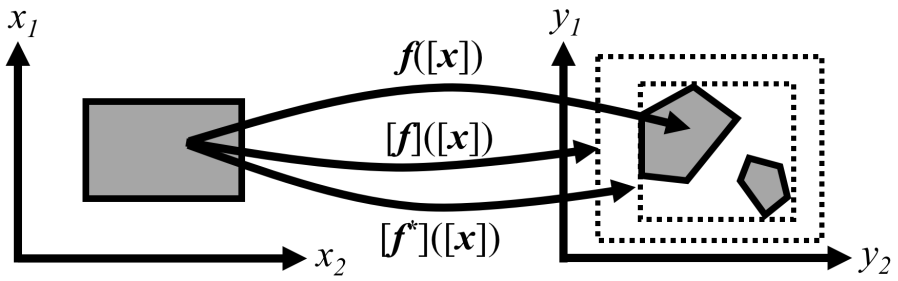


Figure 8.1: Inclusion functions

function $[\mathbf{f}]$ from $\mathbb{IR}^n$ to $\mathbb{IR}^m$ satisfy the following statement.

$$\forall [\mathbf{x}] \in \mathbb{IR}^n, \mathbf{f}([\mathbf{x}]) \subset [\mathbf{f}]([\mathbf{x}]). \tag{8.11}$$

Figure 8.1 illustrates a function $\mathbf{f}$ from $\mathbb{R}^2$ to $\mathbb{R}^2$ with variable x_1 and x_2. The image form of $f([\mathbf{x}])$ can be any shape. Whatever the shape of $\mathbf{f}([\mathbf{x}])$, $[\mathbf{f}]([\mathbf{x}])$ guarantee to contain the shape. However, $[\mathbf{f}]([\mathbf{x}])$ may be pessimistic as Figure 8.1, and the resulted box is not minimum. The minimum inclusion function for $f([\mathbf{x}])$ is denoted by $[\mathbf{f}^*]([x])$, which is far from trivial in general situation.

8.3.1 Natural Inclusion Function

Natural inclusion function is similar to normal functions, except that variables and functions are interval. One can easily convert a normal mathematical equation to Natural inclusion function by replacing the variables to intervals. Natural inclusion function are not minimum in general cases, because of the dependency effects. The accuracy of the natural inclusion function depend on the expression of $\mathbf{f}$.

Example Consider the function $f = (x+1)^2, g = x^2 + 2x + 1$. It is obvious that $f = g$. However, if one applies natural inclusion function to evaluate $[f]$ and $[g]$ with $[x] = [-1, 1]$ then we get:

$$\begin{aligned} [f]([x]) &= ([-1,1]+1)^2 = [0,4] \\ [g]([x]) &= [-1,1]^2 + 2 * [-1,1] + 1 = [0,1] + [-2,2] + 1 = [-2,4] \end{aligned} \tag{8.12}$$

The results are different due to different expression of f and g. For same function but different expression, if x occurs more times, the result tend to be more inaccuracy.

8.3.2 Centred Inclusion Function

Centred inclusion function can be applied to any function. Centred inclusion function linearize the function respect to the input parameter $[\mathbf{x}]$. Given any function $\mathbf{f}$, we can obtain its centred inclusion function as follow:

$$[\mathbf{f}]([\mathbf{x}]) = \mathbf{f}(\boldsymbol{m}) + [\boldsymbol{J_f}]([\mathbf{x}]) * ([\mathbf{x}] - \boldsymbol{m}), \tag{8.13}$$

where $\boldsymbol{m}$ can be any point in $[\mathbf{x}]$ and $[\boldsymbol{J_f}]$ is the Jacobian matrix of $[\mathbf{f}]$.

8.3.3 Mix-Centred Inclusion Function

The centred inclusion function can be improved by a more complicated formulation. The main idea for mix-centred inclusion function is to apply Equation 8.13 for n times, where n is the number of the variables. The case $n = 3$ will be treated first, to simplify the exposition. Next, we show an example for the mix-centred inclusion function for any $[f]([x_1],[x_2],[x_3])$. To derive the mix-centred form, the function $[f]([x_1],[x_2],[x_3])$ is modified to 1 variable form, for example $f_{[x_1],m_3}([x_2])$, where the $[x_2]$ is the only variable for this function and the subscripts are the fixed variables. The interval $[x_1]$ is fixed and the value of $[x_3]$ is fixed at an point m_3. According to this formation, we enumerate the following 3 valid functions using Equation 8.13:

$$[f_{m_2,m_3}]([x_1]) = f_{m_2,m_3}(m_1) + [g^1_{m_2,m_3}]([x_1])([x_1] - m_1) \tag{8.14}$$

$$[f_{[x_1],m_3}]([x_2]) = [f_{[x_1],m_3}](m_2) + [g^2_{[x_1],m_3}]([x_2])([x_2] - m_2) \tag{8.15}$$

$$[f_{[x_1],[x_2]}]([x_3]) \;=\; [f_{[x_1],[x_2]}](m_3) + [g^3_{[x_1],[x_2]}]([x_3])([x_3] - m_3) \tag{8.16}$$

Since $[f_{[x_1],[x_2]}](m_3)$ in Equation 8.16 is equal to $[f_{[x_1],m_3}]([x_2])$ in Equation 8.15, it can be substituted using Equation 8.15. Moreover, $[f_{[x_1],m_3}](m_2)$ in Equation 8.15 is equal to $[f_{m_2,m_3}]([x_1])$ in Equation 8.14, so it can be substituted to Equation 8.14. After the substitutions, the equation become:

$$\begin{aligned}[f_{[x_1],[x_2]}]([x_3]) \;= f_{m_2,m_3}(m_1) \;&+[g^1_{m_2,m_3}]([x_1])([x_1] - m_1)\\ &+[g^2_{[x_1],m_3}]([x_2])([x_2] - m_2)\\ &+[g^3_{[x_1],[x_2]}]([x_3])([x_3] - m_3)\end{aligned} \tag{8.17}$$

It is the same equation as:

$$\begin{aligned}[f]([x_1],[x_2],[x_3]) = f(m_1,m_2,m_3) \;&+\; [g_1]([x_1],m_2,m_3) * ([x_1] - m_1)\\ &+\; [g_2]([x_1],[x_2],m_3) * ([x_2] - m_2)\\ &+\; [g_3]([x_1],[x_2],[x_3]) * ([x_3] - m_3).\end{aligned} \tag{8.18}$$

The expression can be generalize as follow:

$$f([\mathbf{x}]) \subset f(\boldsymbol{m}) + \sum_{i=1}^{n} [g_i]([x_1],...,[x_i],m_{i+1},...,m_n) * ([x_i] - m_i) \tag{8.19}$$

8.3.4 Monotonic Inclusion Function

Monotonic inclusion function is a type of inclusion function that reduce the pessimism efficiently. Monotonic inclusion function performs well when the function is monotonic, because it reduce the interval pessimism by checking whether $\mathbf{f}([\mathbf{x}])$ is monotonic ($\text{lb}(\frac{\partial[f]}{\partial[x_i]}) > 0$ or $\text{ub}(\frac{\partial[f]}{\partial[x_i]}) < 0$) for each interval $[x_i]$. If $\mathbf{f}([\mathbf{x}])$ is monotonic for variable x_i, we can further find out the x_i^+, the desired value of x_i which maximize $\mathbf{f}([\mathbf{x}])$. We can also find out the x_i^- the desired value of x_i which minimize $\mathbf{f}([\mathbf{x}])$. If $\mathbf{f}([\mathbf{x}])$ is monotonic increase, then we let $x_i^+ = \bar{x}_i$ and $x_i^- = \underline{\mathrm{x}}_i$. If $\mathbf{f}([\mathbf{x}])$ is monotonic decrease, then we let $x_i^+ = \underline{\mathrm{x}}_i$ and $x_i^- = \bar{x}_i$.

Let f be a function defined on variables V of domains $[V]$. Let $X \subseteq V$ be a subset of monotonic variables. Consider the values x_i^+ and x_i^- such that: if $x_i \in X$ is an increasing variable, then $x_i^+ = \bar{x}_i$ and $x_i^- = \underline{\mathrm{x}}_i$, and if $x_i \in X$ is an decreasing variable, then $x_i^+ = \underline{\mathrm{x}}_i$ and $x_i^- = \bar{x}_i$.

Consider $W = V \backslash X$ the set of variables not detected monotonic. Then, $\mathbf{f}_{min}$ and $\mathbf{f}_{max}$ are functions defined by:

$$\begin{aligned} \mathbf{f}_{min}(W) &= \mathbf{f}(x_1^-, x_2^-, ..., x_n^-, W) \\ \mathbf{f}_{max}(W) &= \mathbf{f}(x_1^+, x_2^+, ..., x_n^+, W) \end{aligned} \tag{8.20}$$

Finally, the monotonicity-based extension $[\mathbf{f}]$ of $\mathbf{f}$ in the box $[V]$ produces the following interval vector:

$$[\mathbf{f}] = [\underline{[\mathbf{f}_{min}](W)}, \overline{[\mathbf{f}_{max}](W)}] \tag{8.21}$$

Example Consider for example $f(x_1, x_2, w) = -x_1^2 + x_1x_2 + x_2w - 3w$ in the box $[V] = [6, 8] \times [2, 4] \times [7, 15]$.

$$\begin{aligned}[f]([x_1],[x_2],[w]) &= -[6,8]^2 + [6,8] \times [2,4] + [2,4] \times [7,15] - 3 \times [7,15] \\ &= [-83,35]\end{aligned}$$

$$\frac{\partial f}{\partial x_1}(x_1,x_2) = -2x_1 + x_2 \tag{8.22}$$

$$[\frac{\partial f}{\partial x_1}]([6,8],[2,4]) = [-14,-8]. \tag{8.23}$$

Since $[-14,-8] < 0$, we deduce that f is decreasing w.r.t. x_1. Similarly, we can get

$$\frac{\partial f}{\partial x_2}(w) = 1 + w \tag{8.24}$$

$$[\frac{\partial f}{\partial x_2}]([w]) = [8,16] \tag{8.25}$$

Since $[8,16] > 0$, we deduce that x_2 is increasing w.r.t x_2. Finally, $0 \in [\frac{\partial f}{\partial w}]([x_1],[x_2],[w]) = [-1,1]$, so that w is not deduced monotonic. The monotonicity-based evaluation yields:

$$\begin{aligned}[f]([V]) &= [[f](\overline{x_1},\underline{x_2},[w]),[f](\underline{x_1},\overline{x_2},[w])] \\ &= [[f](8,2,[7,15]),[f](6,4,[7,15])] \\ &= [-79,27]\end{aligned}$$

8.4 SIVIA algorithm

A set-inversion problem can be solved by SIVIA algorithm proposed by Jaulin and Walter [1993b]. Let $\mathbf{f}$ be a possibly non-linear function from $\mathbb{R}^n$ to $\mathbb{R}^m$ and let $\mathbb{Y}$ be a subset of $\mathbb{R}^m$. Set inversion is the characterization of

$$\mathbb{X} = \{\mathbf{x} \in \mathbb{R}^n | \mathbf{f}(\mathbf{x}) \in \mathbb{Y}\} = \mathbf{f}^{-1}(\mathbb{Y}). \tag{8.26}$$

Algorithm 8.1 SIVIA algorithm

Require: $[\mathbf{f}], \mathbb{Y}, [\mathbf{x}]$
Ensure: $\underline{\mathbb{X}}, \bar{\mathbb{X}}$
1: if $[\mathbf{f}]([\mathbf{x}]) \cap \mathbb{Y} = \emptyset$ return;
2: $[\mathbf{f}]([\mathbf{x}]) \subset \mathbb{Y}$ then $\underline{\mathbb{X}} := \underline{\mathbb{X}} \cup [\mathbf{x}]; \bar{\mathbb{X}} := \bar{\mathbb{X}} \cup [\mathbf{x}]$; return;
3: $w([\mathbf{x}] < \varepsilon)$ then $\bar{\mathbb{X}} := \bar{\mathbb{X}} \cup [\mathbf{x}]$; return;
4: SIVIA$([\mathbf{f}], \mathbb{Y}, L[\mathbf{x}], \varepsilon, \underline{\mathbb{X}}, \bar{\mathbb{X}})$
5: SIVIA$([\mathbf{f}], \mathbb{Y}, R[\mathbf{x}], \varepsilon, \underline{\mathbb{X}}, \bar{\mathbb{X}})$

For any $\mathbb{Y} \subset \mathbb{R}^m$ and for any function $\mathbf{f}$ admitting a convergent inclusion function $[\mathbf{f}]$, two regular subpavings $\underline{\mathbb{X}}$ and $\bar{\mathbb{X}}$ such that $\underline{\mathbb{X}} \subset \mathbb{X} \subset \bar{\mathbb{X}}$. can be obtained with the algorithm SIVIA (Set Inverter Via Interval Analysis).

SIVIA require a search box $[\mathbf{x}]$ to which $\bar{\mathbb{X}}$ is guaranteed to belong. The general cases may be encountered:

1. If $[\mathbf{f}]([\mathbf{x}])$ has a non-empty intersection with $\mathbb{Y}$, but is not entirely in $\mathbb{Y}$, then $[\mathbf{x}]$ may contain a part of the solution set. $[\mathbf{x}]$ is said to be *undetermined*. If $[\mathbf{x}]$ has a width greater than a pre-specified percision parameter ε, then it should be bisected and the test should be recursively applied to these newly generated boxes.

2. If $[\mathbf{f}]([\mathbf{x}])$ has an empty intersection with $\mathbb{Y}$, then $[\mathbf{x}]$ does not belong to $\mathbb{X}$ and can be cut off from the solution tree.

3. If $[\mathbf{f}]([\mathbf{x}])$ is entirely in $\mathbb{Y}$, then $[\mathbf{x}]$ belongs to the solution subpaving $\mathbb{X}$, and is stored in $\underline{\mathbb{X}}$ and $\bar{\mathbb{X}}$.

4. If the box considered is undetermined, but its width is lower than ε, then it is deemed small enough to be considered in the outer approximation $\bar{\mathbb{X}}$ of $\mathbb{X}$.

Algorithm 8.1 shows the SIVIA based on an inclusion function, where $\underline{\mathbb{X}}$ and $\bar{\mathbb{X}}$ have been initialized as empty.

8.5 Contractors

Forward-backward contractor is a simple technique to contract interval using constraints. It can be used in wireless sensor localization problem. However, this contractor only consider one equation at a time and solve equation independently. On the other hand, Newton contractor and Krawczyk contractor are powerful since they can consider multiple equations at the same time. In this thesis, we aim to compare the forward-backward contractor between Newton contractor and Krawczyk contractor.

8.5.1 Constraint Satisfaction problem (CSP)

Consider n_x variables $x_i \in \mathbb{R}$, $i \in \{1, ..., n_x\}$ and n_f constraint

$$f_j(x_1, x_2, ..., x_{n_x}) = 0, j \in \{1, ..., n_f\} \tag{8.27}$$

We assume that the prior domain for x is $[\mathbf{x}] = [x_1] \times [x_2] \times ... \times [x_{n_x}]$. Let $\mathbf{f}$ be the function whose coordinate functions are the f_js. Equation 8.27 can be formulated as a constraint satisfaction problem

$$\mathcal{H} : (\mathbf{f}(\mathbf{x}) = \mathbf{0}, \mathbf{x} \in [\mathbf{x}]) \tag{8.28}$$

The solution set of $\mathcal{H}$ is defined as

$$\mathbb{S} = \{\mathbf{x} \in [\mathbf{x}] | \mathbf{f}(\mathbf{x}) = \mathbf{0}\} \tag{8.29}$$

Contracting $\mathbb{S}$ means replacing $[\mathbf{x}]$ by a smaller domain which still contain the solution set. The optimal contraction is to replace [x] by the smallest box that contains $\mathbb{S}$.

8.5.2 Forward-Backward Contractor

The forward-backward contractor is based on constraint propagation. It decomposes the function into forward constraints and backward constraints to reduce the size of the potential search space. It tries to find a consistent solution for each equation in ICSP. The forward-backward contractor only considers the equations in ICSP one by one, so it can be applied on the ICSP where the number of functions and the number of variables are different. Consider a simple constraint $x_3 = x_1 x_2$. This constraint can be rewritten as forward phase and backward phase:

$$\begin{aligned} \text{Forward phase: } x_3 &= x_1 x_2 \\ \text{Backward phase: } x_1 &= x_3 / x_2 \\ x_2 &= x_3 / x_1 \end{aligned}$$

If the box $[\mathbf{x}] = [x_1] \times [x_2] \times [x_3] = [1, 4] \times [1, 4] \times [8, 40]$ is the intervals we want to contract. The contractor can be build as

$$\begin{aligned} \text{Forward phase: } [x_3] &= [x_3] \cap ([x_1][x_2]) \\ \text{Backward phase: } [x_2] &= [x_2] \cap ([x_3]/[x_1]) \\ [x_1] &= [x_1] \cap ([x_3]/[x_2]) \end{aligned} \tag{8.30}$$

We can keep calculating Equation 8.30 until the width of $[x_1]$, $[x_2]$, and $[x_3]$ are no longer shrink. The result box is reduced to $[2, 4] \times [2, 4] \times [8, 16]$.

In Figure 8.2, two anchors a and b are located at $(4, 3)$ and $(-4, 3)$, respectively. Both of the anchors can measure the distance $d + e$ to agent and send this information to the agent, where e is the measurement noise with zero mean and variance σ^2. There is one agent located at $\mathbf{x} = (0, 0)$, whose location need to be determined. The observation location of the agent for anchors a and b are (u_{a1}, u_{a2}) and (u_{b1}, u_{b2}), re-

spectively. We can model scenario of Figure 8.2 to a constraint satisfaction problem:

$$\mathcal{H}: \begin{cases} (u_{a1}-4)^2+(u_{a2}-3)^2-(d_a^2+\mathcal{N}(0,\sigma^2))=0 \\ (u_{b1}+4)^2+(u_{b2}-3)^2-(d_b^2+\mathcal{N}(0,\sigma^2))=0 \\ u_{a1}-x_1=0 \\ u_{b1}-x_1=0 \\ u_{a2}-x_2=0 \\ u_{b2}-x_2=0 \\ d_a-5=0 \\ d_b-5=0 \end{cases} \tag{8.31}$$

Since $\mathcal{N}(0,\sigma^2)$ is uncertain, it is difficult to calculate Equation 8.31. Instead, we can model the Equation 8.31 to a interval constraint satisfaction problem (ICSP) given $3\sigma = 0.2$

$$\mathcal{H}: \begin{cases} ([u_{a1}]-4)^2+([u_{a2}]-3)^2-[d_a^2]=0 \\ ([u_{b1}]+4)^2+([u_{b2}]-3)^2-[d_b^2]=0 \\ [u_{a1}]-[x_1]=0 \\ [u_{b1}]-[x_1]=0 \\ [u_{a2}]-[x_2]=0 \\ [u_{b2}]-[x_2]=0 \\ [d_a]=[4.8,5.2],[d_b]=[4.8,5.2] \end{cases} \tag{8.32}$$

We assume the agent's location is already known within the box $[\mathbf{x}] = [[-2,2],[-2,2]]^T$, but still have high uncertainty. By applying the forward-backward contractor, we can reduce the $\mathbf{x}$ down to $[-1.1,1.1][-1.3,2.0]$. It is apparent that the result is not optimized. Forward-backward contractor fails to reduce $[\mathbf{x}]$ to the minimum size of the solution, so it is not an optimal contractor for this ICSP. Forward-backward can-

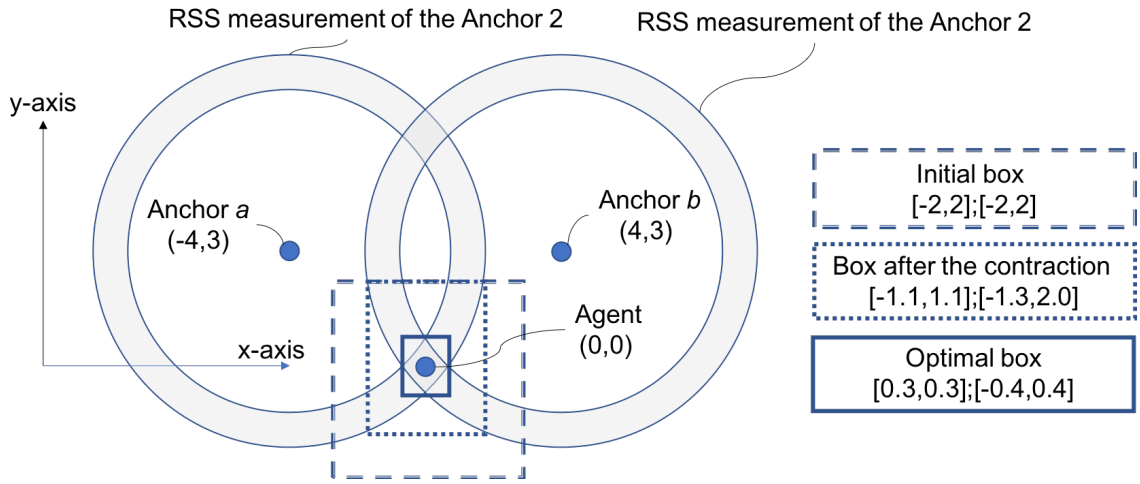


Figure 8.2: A scenario of localization CSP problem. The two anchors a and b are located at $(4, 3)$ and $(-4, 3)$, respectively. Both of the anchors can measure the distance $d + e$ to agent and send this information to the agent, where e is the measurement noise with zero mean and variance σ^2. There is one agent located at $\mathbf{x} = (0, 0)$, whose location need to be determined.

not consider multiple equations at the same time. Krawczyk and Newton contractor are numerical contractors that consider ICSP as a matrix. They use the Jacobian matrix to find the solution of ICSP. Since the Jacobian matrix can evaluate the slop using multiple constraints instead of one. However, Krawczyk and Newton contractor cannot contract successfully when the slop of the constraints are dramatically variant.

8.5.3 Newton Iteration Function

Here we introduce the newton iteration function because it is a preliminary used for building Krawczyk and Newton contractors. Figure 8.3 shows the idea of Newton iteration: one starts with an initial guess ($\hat{\boldsymbol{x}}_{\boldsymbol{n}}$) which is reasonably close to the true root, then the function ($\mathbf{f}$) is approximated by its tangent line ($\hat{\mathbf{f}}$), and one computes the x-intercept of this tangent line (a point $\hat{\mathbf{x}}_{n+1}$ such that $\hat{\mathbf{f}}(\hat{\mathbf{x}}_{n+1}) = \mathbf{0}$). This

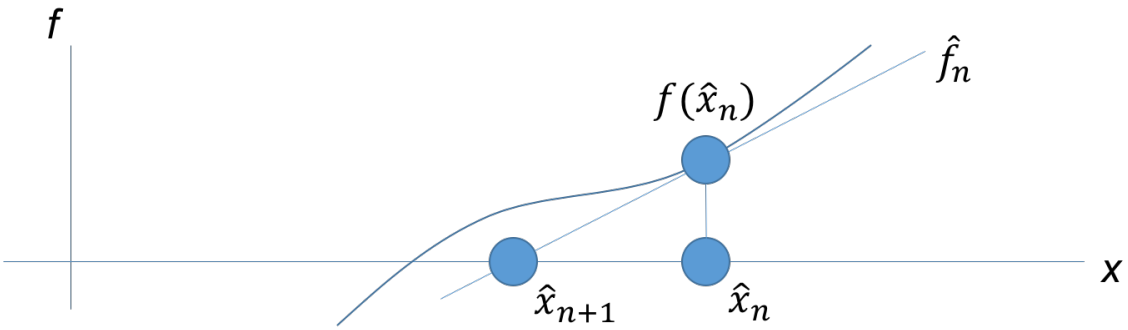


Figure 8.3: Newton Iteration in one dimensional. One starts with an initial guess ($\hat{\boldsymbol{x}}_{\boldsymbol{n}}$) which is reasonably close to the true root, then the function ($\mathbf{f}$) is approximated by its tangent line ($\hat{\boldsymbol{f}}$), and one computes the x-intercept of this tangent line (a point $\hat{\mathbf{x}}_{n+1}$ such that $\hat{\boldsymbol{f}}(\hat{\mathbf{x}}_{n+1}) = \mathbf{0}$). This x-intercept is a better approximation to the function's root than the original guess, and the method can be iterated.

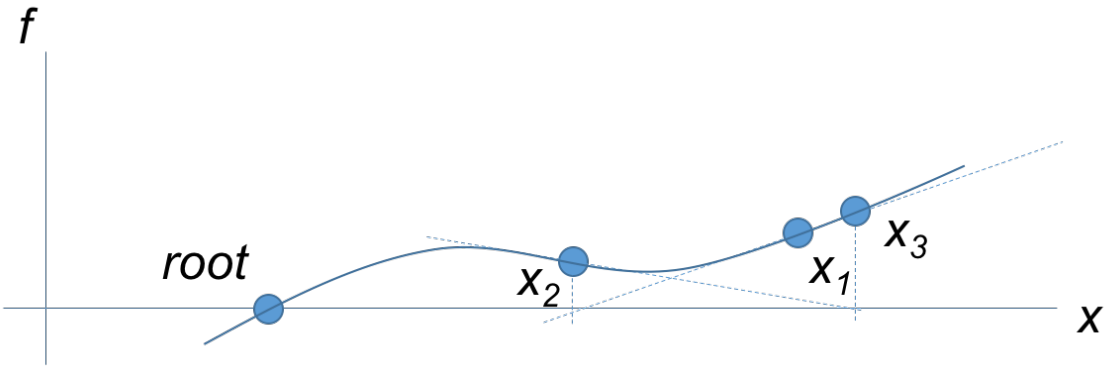


Figure 8.4: A scenario that newton iteration fails to converge the solution to root, where x_1, x_2, and x_3 are the results of each iteration. We can observe that from the second to the third iteration, x_3 is farther away from the root than x_2, so the solution is not converged.

x-intercept is a better approximation to the function's root than the original guess, and the method can be iterated. If the initial guess is far away from the true root and the function is not monotonic, this iterative procedure may not converge to the solution point due to the bad approximation of the tangent lines. Figure 8.4 shows a scenario that newton iteration fails to converge the solution to root, where x_1, x_2, and x_3 are the results of each iteration. We can observe that from the second to the third iteration, x_3 is farther away from the root than x_2, so the solution is not converged.

To introduce the newton iteration in a more formal way, we first assume that $\mathbf{f} : R^k \to R^k$ is differential, and we have the first guess of the solution $\hat{\boldsymbol{x}}_n$. Since $\mathbf{f}(\hat{\boldsymbol{x}}_n) \neq \mathbf{0}$, we want to find a better solution $\hat{\boldsymbol{x}}_{\boldsymbol{n+1}}$. First, we approximated $\mathbf{f}$ by its tangent line at $\hat{\mathbf{x}}_n$

$$\mathbf{f}(\mathbf{x}) \approx \hat{\mathbf{f}}(\mathbf{x}) = \mathbf{f}(\hat{\boldsymbol{x}}_{\boldsymbol{n}}) + \mathbf{J}_{\mathbf{f}}(\hat{\boldsymbol{x}}_n)(\mathbf{x} - \hat{\boldsymbol{x}}_{\boldsymbol{n}}) \tag{8.33}$$

Where $\mathbf{J}_{\mathbf{f}}$ is the Jacobian matrix of $\mathbf{f}$. Then, we solve $\hat{\mathbf{f}}(\hat{\boldsymbol{x}}_{n+1}) = \mathbf{0}$ to get the x-intercept of this tangent line. Equation 8.33 is reformulated as

$$\hat{\mathbf{x}}_{n+1} = \hat{\boldsymbol{x}}_n - \mathbf{J}_{\mathbf{f}}^{-1}(\hat{\boldsymbol{x}}_n)\mathbf{f}(\hat{\boldsymbol{x}}_n) \tag{8.34}$$

where $\hat{\mathbf{x}}_{n+1}$ is the better approximation of the $\hat{\boldsymbol{x}}_{\boldsymbol{n}}$. We compute Equation 8.34 iteratively until an sufficient accurate value is obtained.

In Equation 8.34, the one has multiply with inverse of the k-by-k Jacobian matrix $\mathbf{J}_{\mathbf{f}}^{-1}(\hat{\boldsymbol{x}})$. This is the Newton iteration method solving a system of k equations. Assume that the system has m variables and k equations with $m > k$. The Newton's method can still be used if the algorithms use the generalized inverse of the non-square Jacobian matrix $J^+ = ((J^T J)^{-1})J^T$ (from Wikipedia) instead of the inverse of $\mathbf{J}$. If the non-linear system has no solution, the method attempts to find a solution in the non-linear least squares sense. For subsection 8.5.4 on the next page and subsection 8.5.5 on page 100, we will demonstrate Krawczyk contractor and Newton contractor using a system with k variables and k equations (invertible matrix) for simplification.

8.5.4 Krawczyk Contractor

Theoretically, we can compute the interval version of newton iteration by the following equation, which is the inclusion function of Equation 8.34 where $\hat{\mathbf{x}}_n$ and $\hat{\mathbf{x}}_{n+1}$ are is replace by the interval $[\mathbf{x}]$ and $[\mathbf{x}']$, respectively:

$$[\mathbf{x}'] = [\mathbf{x}] - [\mathbf{J}_\mathbf{f}^{-1}]([\mathbf{x}])[\mathbf{f}]([\mathbf{x}]) \tag{8.35}$$

where $[\mathbf{x}']$ is a fixed-point solver for CSP problem. $[\mathbf{x}]$ is the interval which contain the solution set $\mathbb{S}$, $[\mathbf{f}]$ is the inclusion function of $\mathbf{f}$ and $[\boldsymbol{J_f}]$ is the inclusion function for the Jacobian matrix $\mathbf{J}$ of function $\mathbf{f}$. However, Equation 8.35 is suffered from too many occurrence of $[\mathbf{x}]$. To reduce the occurrence, Krawczyk contractor replace $\mathbf{J}_\mathbf{f}^{-1}([\mathbf{x}])$ by $\mathbf{J}_\mathbf{f}^{-1}(\mathbf{x}_0)$ in Equation 8.35 to reduce the occurrence of $\mathbf{x}$:

$$[\boldsymbol{\psi}]([\mathbf{x}]) = [\mathbf{x}] - \mathbf{J}_\mathbf{f}^{-1}(\mathbf{x}_0)[\mathbf{f}]([\mathbf{x}]) \tag{8.36}$$

where $\mathbf{x}_0$ is a point near $\mathbf{x}$. This equation is still a fixed-point solver because $[\mathbf{f}]([\mathbf{x}]) = 0$ imply $[\boldsymbol{\psi}]([\mathbf{x}]) = [\mathbf{x}] - \mathbf{J}_\mathbf{f}^{-1}(\mathbf{x}_0) * 0 = [\mathbf{x}]$. Then, it linearize Equation 8.36 using centred inclusion function. Combine Equation 8.36 and Equation 8.13 to get

$$[\boldsymbol{\psi}]([\mathbf{x}]) = \boldsymbol{\psi}(\mathbf{x}_0) + [\mathbf{J}_\psi]([\mathbf{x}]) * ([\mathbf{x}] - \mathbf{x}_0) \tag{8.37}$$

where $[\boldsymbol{J_\psi}]$ is an inclusion function for the Jacobian matrix of $\boldsymbol{\psi}$. We can derive $[\mathbf{J}_\psi]$ using Equation 8.36. Notice that the $\mathbf{J}_\mathbf{f}(\mathbf{x}_0)$ is only a scalar matrix, the differential operation of Equation 8.36 become easier than Equation 8.34 and the occurrence of $[\mathbf{x}]$ is reduced.

$$\begin{aligned} \mathbf{J}_\psi(\mathbf{x}) &= \mathit{Inclusion\_Jacobian}(\mathbf{x} - \mathbf{J}_\mathbf{f}^{-1}(\mathbf{x}_0)\mathbf{f}(\mathbf{x})) && (8.38) \\ &= I - \mathbf{J}_\mathbf{f}^{-1}(\mathbf{x}_0)\mathbf{J}_\mathbf{f}(\mathbf{x}) && (8.39) \end{aligned}$$

From the centred inclusion fix-point function is obtained, the Krawczyk contractor can be formulate as follow.

$$\mathcal{C}_K : [\mathbf{x}] \mapsto [\mathbf{x}] \cap (\boldsymbol{\psi}(\mathbf{x}_0) + [\mathbf{J}_\psi]([\mathbf{x}]) * ([\mathbf{x}] - \mathbf{x}_0)) \tag{8.40}$$

Equation 8.40 can be further modified by substitute $\boldsymbol{\psi}$ with Equation 8.36 and by substitute $[\boldsymbol{J}_{\boldsymbol{\psi}}]([\mathbf{x}])$ with Equation 8.38.

$$\mathcal{C}_K : [\mathbf{x}] \mapsto [\mathbf{x}] \cap (\mathbf{x}_0 - \mathbf{J}_\mathbf{f}^{-1}(\mathbf{x}_0)\mathbf{f}(\boldsymbol{x_0}) + (I - \mathbf{J}_\mathbf{f}^{-1}(\mathbf{x}_0)[\mathbf{J}_\mathbf{f}]([\mathbf{x}])) * ([\mathbf{x}] - \mathbf{x}_0)) \tag{8.41}$$

8.5.5 Newton Contractor

Newton contractor is not a direct interval version of the newton iteration function. Still, it approximates the function by its tangent lines. Given any $\mathbf{x}$ and $\mathbf{x}_0 \in R$ and a continuous function $\mathbf{f}$, we can find a point ξ that satisfy the following equation:

$$\mathbf{f}(\mathbf{x}) = \mathbf{f}(\mathbf{x}_0) + \boldsymbol{J}_\mathbf{f}(\boldsymbol{\xi})(\mathbf{x} - \mathbf{x}_0). \tag{8.42}$$

Since $\boldsymbol{\xi}$ is guaranteed within the interval $[\mathbf{x}]$, we turn Equation 8.42 into the interval version as follow.

$$[\mathbf{f}]([\mathbf{x}]) = \mathbf{f}(\boldsymbol{x_0}) + [\boldsymbol{J}_\mathbf{f}]([\mathbf{x}])([\mathbf{x}] - \mathbf{x}_0) \tag{8.43}$$

where $\boldsymbol{x_0}$ can be any point in the interval $[\mathbf{x}]$. According to mean-value theorem, we can find an approximated slop of function within $[\mathbf{x}]$ such that the approximate function has the same root as the original function, so $\mathbf{J}_\mathbf{f}(\boldsymbol{\xi})$ is replaced by $[\mathbf{J}_\mathbf{f}]([\mathbf{x}])$. Since the Jacobian matrix $[\mathbf{J}_\mathbf{f}]([\mathbf{x}])$ is a interval instead of a point. It enumerate all the possibility of the approximated tangent lines. In order to find all the x-intercepts of those tangent lines, we solve $[\hat{\mathbf{f}}]([\mathbf{x}]) = 0$ using Equation 8.43 to get

$$[\mathbf{x}] = \boldsymbol{x_0} - [\mathbf{J}_\mathbf{f}^{-1}]([\mathbf{x}])\mathbf{f}(\mathbf{x}_0) \tag{8.44}$$

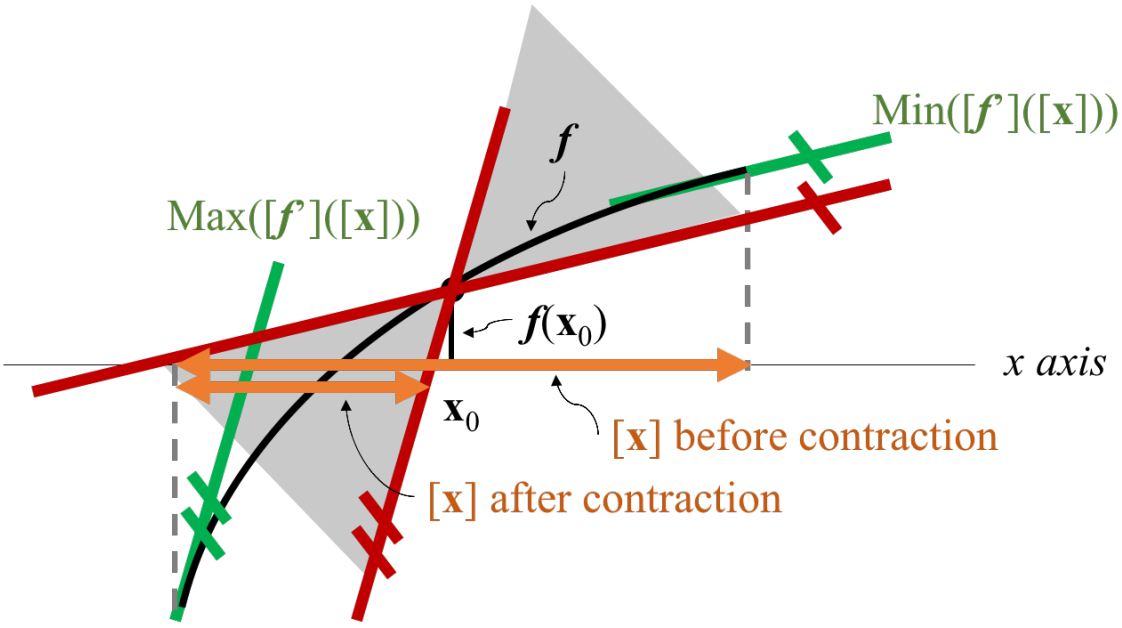


Figure 8.5: A simple example of newton contractor of a $k = 1$ (1 variable and 1 function) system. We use colors to distinguish different lines. The curve black line represents the function $\mathbf{f}$. The orange lines represent the interval $[\mathbf{x}]$ before (after) the contraction. The green lines are the tangent lines with maximum and minimum slop. The red lines are maximum and minimum slop shifted to the point $(\boldsymbol{x_0}, \mathbf{f}(\boldsymbol{x_0}))$. There are two gray triangle area represent all possibility of the tangent lines that pass through the point $(\boldsymbol{x_0}, \mathbf{f}(\boldsymbol{x_0}))$. Since the tangent lines are the approximation of function $\mathbf{f}$, we can find the better solution by finding all the x-intercepts of the tangent lines.

By applying Equation 8.30, the Newton Contractor can be derived as follows.

$$\mathcal{C}_N : [\mathbf{x}] \mapsto [\mathbf{x}] \cap (\boldsymbol{x_0} - [\mathbf{J_f^{-1}}]([\mathbf{x}])\mathbf{f}(\boldsymbol{x_0})), \hat{\boldsymbol{x}} \in [\mathbf{x}] \tag{8.45}$$

Figure 8.5 is a simple example of newton contractor of a $k = 1$ (1 variable and 1 function) system. We use colors to distinguish different lines. The curve black line represents the function $\mathbf{f}$. The orange lines represent the interval $[\mathbf{x}]$ before and after the contraction. The green lines are the tangent lines with maximum and minimum slop. The red lines are maximum and minimum slop shifted to the point $(\boldsymbol{x_0}, \mathbf{f}(\boldsymbol{x_0}))$. There are two gray triangle area represent all possibility of the tangent lines that pass through the point $(\boldsymbol{x_0}, \mathbf{f}(\boldsymbol{x_0}))$. Since the tangent lines are the approximation of function $\mathbf{f}$, we can find the better solution by intersecting x-axis and the triangle area.

8.5.6 Gauss-Seidel Contractor

Gauss-Seidel contractor can be applied to calculate $[\mathbf{J_f^{-1}}]([\mathbf{x}])\mathbf{f}(\boldsymbol{x_0})$ in Equation 8.45 efficiently. Gauss-Seidel contractor is a iterative interval matrix solver. Given a problem: $\mathbf{Ap} = \boldsymbol{b}$, where $\boldsymbol{A} \in [\mathbf{A}], \mathbf{b} \in [\mathbf{b}], \mathbf{p} \in [\mathbf{p}]$. Assume $\mathbf{A}$ is invertible. We can decompose $\mathbf{A}$ as the sum of a diagonal matrix and a matrix with zeros on its diagonal:

$$\mathbf{A} = diag(\mathbf{A}) + extdiag(\mathbf{A}) \tag{8.46}$$

Solving $\boldsymbol{Ap} = \boldsymbol{b}$ is equivalent to

$$diag(\mathbf{A})\mathbf{p} + extdiag(\mathbf{A})\mathbf{p} = \mathbf{b} \tag{8.47}$$

The equation can be rewritten as

$$\mathbf{p} = diag(\mathbf{A})^{-1}(\mathbf{b} - extdiag(\mathbf{A})\mathbf{p}) \tag{8.48}$$

An inclusion function for Equation 8.48 is

$$[\mathbf{p}] = diag([\mathbf{A}])^{-1}([\mathbf{b}] - extdiag([\mathbf{A}])[\mathbf{p}]) \tag{8.49}$$

We can contract $\mathbf{p}$ by applying Equation 8.49 iteratively until $\mathbf{p}$ is contracted.

To calculate $[\mathbf{J}_\mathbf{f}^{-1}]([\mathbf{x}])\mathbf{f}(\mathbf{x}_0)$ using Gauss-Seidel contractor, one can take $[\mathbf{A}] = [\mathbf{J}_\mathbf{f}]([\mathbf{x}])$, $[\mathbf{b}] = \mathbf{f}(\mathbf{x}_0)$, and $[\mathbf{p}] = \mathbf{x}_0 - [\mathbf{x}]$. The reason for $[\mathbf{p}] = \mathbf{x}_0 - [\mathbf{x}]$ is because Equation 8.43 imply that $[\mathbf{J}_\mathbf{f}^{-1}]([\mathbf{x}])\mathbf{f}(\mathbf{x}_0) = \mathbf{x}_0 - [\mathbf{x}]$.

8.6 Conclusion

This chapter introduces the tools of interval analysis. Interval analysis allow us to consider the uncertainty as bounds. With the inclusion function, the bounded uncertainty can propagate to different domains. A contractor is able to reduce the solution space using multiple inclusion functions. For the next section several localization algorithms, which are based on interval analysis will be discussed.

Chapter 9

Problem Formulation and Related Approaches

9.1 Introduction

Localization in a network of sensors is usually performed considering the received signal strength (RSS), the time of arrival (TOA), the time difference of arrival (TDOA) of electromagnetic or acoustic signals transmitted by some anchor nodes, see [Gezici, 2008] for an overview. In this thesis, one will consider localization from RSS, which has received significant attention for more than a decade. By measuring the received power of agent node from a specify location, we are able to estimate the location of an agent.

9.2 Problem Formulation and Assumptions

Consider a set of n_{a} anchor nodes situated on a plane, with fixed and known locations $\boldsymbol{\theta}_i$, $i = 1, \ldots, n_{\mathrm{a}}$, in some reference frame attached to the plane. This paper aims at providing a guaranteed NACR for the unknown location $\boldsymbol{\theta}_0$ of some static node in the x-y plane, *i.e.*, for a given confidence level, to provide a subset of locations to

which the node belongs with the considered confidence level. For that purpose, the node periodically emits some electromagnetic or acoustic signal, which is received by the anchors. The k-th RSS measurement performed by anchor node i is denoted $y\left(i,k\right)$. The anchor nodes transmit the RSS measurements to some central processing unit in charge of estimating $\boldsymbol{\theta}_0$ and the associated confidence region from $y\left(i,k\right)$, $i=1,\ldots,n_{\mathrm{a}}$, $k=1,\ldots,n$.

We assume that the k-th measurement obtained by anchor node i is described by the Okumura-Hata model [Okumura et al., 1968]

$$y\left(i,k\right)=P_0-10\gamma_{\mathrm{P}}\log_{10}\frac{\|\boldsymbol{\theta}_0-\boldsymbol{\theta}_i\|}{d_0}+\varepsilon\left(i,k\right), \tag{9.1}$$

where P_0 is the signal power measured at some known reference distance d_0, γ_{P} is the path-loss exponent, $\|\cdot\|$ is the Euclidian norm, and $\varepsilon\left(i,k\right)$ is the measurement noise. One assumes that γ_{P} is the same for all anchors. Usually, the $\varepsilon\left(i,k\right)$s are assumed to be realizations of some normal or log-normal random variables. In this paper, one only assumes that the $\varepsilon\left(i,k\right)$s are realizations of independently, not necessarily identically distributed random variables with distributions symmetric around zero. Since P_0 and γ_{P} are not necessarily known by the anchors nodes, one considers the parameter vector $\mathbf{p}=\left[\boldsymbol{\theta}_0^T,P_0,\gamma_{\mathrm{P}}\right]^T$ for which a confidence region has to be estimated and from which a confidence region for $\boldsymbol{\theta}_0$ may then be deduced. Notice that $d_0=1$ is a fixed parameter linked with P_0, thus it does not appear in $\mathbf{p}$.

Assuming that there exists some *true* value $\mathbf{p}^*=\left[\boldsymbol{\theta}_0^{*T},P_0^*,\gamma_{\mathrm{P}}^*\right]^T$ of the parameter vector, Equation 9.1 can be rewritten as

$$y(i,k)=y^{\mathrm{m}}\left(i,\mathbf{p}^*\right)+\varepsilon\left(i,k\right) \tag{9.2}$$

with

$$y^{\mathrm{m}}\left(i,\mathbf{p}^*\right)=P_0^*-10\gamma_{\mathrm{P}}^*\log_{10}\frac{\|\boldsymbol{\theta}_0^*-\boldsymbol{\theta}_i\|}{d_0}. \tag{9.3}$$

One assumes further that the search for the confidence region for $\mathbf{p}^*$ has to be done in some known compact set $\mathbb{P}_0$, which may be chosen arbitrarily large.

9.3 State-of-the-art Approaches

Range-based approaches consider the localization problem in mobile sensor networks using interval analysis and other tools. Range-based approaches can take outliers into account, i.e., data not matching the observation model. Some range-based approaches perform a set-membership estimation, where only the maximal number of outliers is required to be known. Mourad et al. [2011b], Léger and Kieffer [2010] Using these algorithms, estimates consist of sets of boxes whose union surely contains the correct location of the sensor, provided that the considered hypotheses are satisfied.

9.3.1 Bounded-Error Approach

BE estimation techniques Halbwachs and Meizel [1996], Jaulin and Walter [1993a] assume that the noise samples $\varepsilon(i,k)$ in Equation 9.1 are bounded with known bounds, *i.e.*, that $\varepsilon(i,k) \in [\underline{e}(i,k), \overline{e}(i,k)]$, with $\underline{w}(i,k)$ and $\overline{w}(i,k)$ known for all $i = 1 \ldots n_a$ and $k = 1 \ldots n$. The *set* of all parameter vectors $\mathbf{p}$ belonging to some initial search set $\mathbb{P}_0$ that are consistent with the measurement model, the measurements, and the noise bounds is then characterized. This set may be defined as

$$\mathbb{P}_{\text{BE}} = \{\mathbf{p} \in \mathbb{P}_0 | y_{\text{m}}(i,\mathbf{p}) \in y(i,k) - [\underline{w}(i,k), \overline{w}(i,k)], i = 1 \ldots n_a, k = 1 \ldots n\}. \tag{9.4}$$

Introducing

$$\overline{\mathbf{w}}(i) = (\bar{w}(i,1), \ldots, \bar{w}(i,n))^T$$

$$\underline{\mathbf{w}}(i) = (\underline{w}(i,1), \ldots, \underline{w}(i,n))^T$$

and

$$\overline{\mathbf{w}} = \left(\bar{\mathbf{w}}^T(1), \ldots, \bar{\mathbf{w}}^T(n_\mathrm{a})\right)^T$$

$$\underline{\mathbf{w}} = \left(\underline{\mathbf{w}}^T(1), \ldots, \underline{\mathbf{w}}^T(n_\mathrm{a})\right)^T$$

and

$$[\mathbf{y}] = [\mathbf{y} - \overline{\mathbf{w}}, \mathbf{y} - \underline{\mathbf{w}}], \tag{9.5}$$

one may formulate (Equation 9.4) as a set-inversion problem

$$\mathbb{P}_{\mathrm{BE}} = \mathbb{P}_0 \cap \mathbf{y}_\mathrm{m}^{-1}([\mathbf{y}]), \tag{9.6}$$

which may be characterized using SIVIA in section 8.4.

The resulting set $\mathbb{P}_{\mathrm{BE}}$ is guaranteed to contain the true value $\mathbf{p}^*$ of the parameter vector provided that all noise samples are within the considered bounds.

9.3.2 Robust Bounded-Error Approach

BE approaches may provide an empty set $\mathbb{P}_{\mathrm{BE}} = \emptyset$ as a result. Such result may be due to noise bounds which were violated by some measurements, to the initial search set which does not contain the solution set, or to the model which is a too coarse approximation of reality.

RBE estimation methods [Jaulin et al., 1996, Pronzato and Walter, 1996, Léger and Kieffer, 2010] have been proposed to improve the robustness to outliers of BE estimation techniques. In (Equation 9.4), the solution set is defined as the set of parameter vectors that are consistent with all measurements. With RBE estimation, one tries to find the set of parameter vectors that are consistent with all but ξ measurements and related noise bounds. In this context, the solution set may be formulated as

$$\mathbb{P}_{\mathrm{RBE},\xi} = \{\mathbf{p} \in \mathbb{P}_0 | \tau(\mathbf{p}) \in \mathbb{Y}_\xi\}, \tag{9.7}$$

where

$$\tau(\mathbf{p}) = \sum_{i=1}^{n_a} \sum_{k=1}^{n} \tau_{i,k}(\mathbf{p}), \tag{9.8}$$

$$\tau_{i,k}(\mathbf{p}) = \begin{cases} 1 & \text{if } y_{\mathrm{m}}(i,\mathbf{p}) \in y(i,k) - [\underline{w}(i,k), \overline{w}(i,k)], \\ 0 & \text{else} \end{cases} \tag{9.9}$$

and $\mathbb{Y}_\xi = [n_a n - \xi, n_a n]$.

Again, (Equation 9.4) can be viewed as a set-inversion problem

$$\mathbb{P}_{\mathrm{RBE},\xi} = \mathbb{P}_0 \cap \tau^{-1}(\mathbb{Y}_\xi), \tag{9.10}$$

which may be solved using SIVIA.

This approach does not need to specify in advance which measurements have to be considered as outliers. Considering a fixed value of ξ, one obtains an estimator robust against ξ outliers. If ξ is not fixed *a priori*, a possible strategy is to start with $\xi = 0$, and to increase ξ as long as an empty solution is provided by SIVIA. When a non-empty solution set has been obtained for some $\xi_0 \geqslant 0$, one may consider $\xi = \xi_0 + 1$ or $\xi = \xi_0 + 2$ and evaluate $\mathbb{P}_{\mathrm{RBE},\xi}$ to further increase the robustness of the estimated set $\mathbb{P}_{\mathrm{RBE},\xi}$.

The resulting set $\mathbb{P}_{\mathrm{RBE},\xi}$ is *guaranteed* to contain the true value $\mathbf{p}^*$ of the parameter vector *provided that there are actually less than ξ outliers.*

9.3.3 Nonlinear ML estimate and Cramér-Rao Lower Bound

9.3.3.1 Maximum Likelihood Estimation (MLE)

In MLE [Ziskind and Wax, 1988], Assuming that the noise samples $w(i,k)$ in Equation 9.1 are independently and identically distributed (iid) as zero-mean log-normal

with variance σ^2 , one may obtain the MLE of $\mathbf{p}^*$ as

$$\hat{\mathbf{p}}_{\mathrm{MLE}} = \arg\max \pi(\mathbf{y}|\mathbf{p}), \tag{9.11}$$

where $\pi(\mathbf{y}|\mathbf{p})$ is the likelihood function, which requires a precise knowledge of the noise probability distribution. The cost function to minimize is usually non-convex and has several local minimizers. Nevertheless, one may try to minimize it using, e.g., lsqnonlin of Matlab. The function applies Levenberg–Marquardt algorithm (LMA) to solve the optimization problem. First, considering the logarithm of $\pi(\mathbf{y}|\mathbf{p})$, we get

$$\begin{aligned} l(\mathbf{p}) &= \log(\pi(\mathbf{y}|\mathbf{p})) \\ &= \log\left(\prod_{i=1}^{n_a}\prod_{j=1}^{n}\frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(y(i,j)-y^{\mathrm{m}}(i,\mathbf{p}))^2}{2\sigma^2}\right)\right)) \qquad (9.12) \\ &= -c\sum_{i=1}^{n_a}\sum_{j=1}^{n}\left(y(i,j)-y^{\mathrm{m}}(i,\mathbf{p})\right)^2, \qquad (9.13) \end{aligned}$$

where $c = \frac{\log\left(\frac{1}{\sqrt{2\pi}\sigma}\right)}{2\sigma^2}$.

In order to find $\hat{\mathbf{p}}_{\mathrm{MLE}}$, we need to maximum $\pi(\mathbf{y}|\mathbf{p})$, which implies minimizing

$$l'(\mathbf{p}) = \sum_{i=1}^{n_a}\sum_{j=1}^{n}(y(i,j)-y^{\mathrm{m}}(i,\mathbf{p}))^2. \tag{9.14}$$

Given a start point of $\mathbf{p}$, LMA replaces $\mathbf{p}$ by a new estimation $\mathbf{p}+\boldsymbol{\delta}$. To determine $\boldsymbol{\delta}$, the functions $y^{\mathrm{m}}(i,\mathbf{p}+\delta)$ are approximated by their linearization.

$$y^{\mathrm{m}}(i,\mathbf{p}+\boldsymbol{\delta}) \approx y^{\mathrm{m}}(i,\mathbf{p}) + \frac{\partial y^{\mathrm{m}}(i,\mathbf{p})}{\partial \mathbf{p}}\delta \tag{9.15}$$

Combining Equation 9.14 and Equation 9.15, we get

$$l'(\mathbf{p}) \approx c \sum_{i=1}^{n_a} \sum_{j=1}^{n} \left(y(i,j) - y^{\mathrm{m}}(i,\mathbf{p}) - \frac{\partial y^{\mathrm{m}}(i,\mathbf{p})}{\partial \mathbf{p}} \boldsymbol{\delta} \right)^2. \tag{9.16}$$

Reformulate Equation 9.16 in vector notation, we get

$$\begin{aligned} l'(\mathbf{p}) &\approx \left\| \mathbf{y} - \mathbf{y}^{\mathrm{m}}(\mathbf{p}) - \mathbf{J}\delta \right\|^2, & (9.17) \\ &= (\mathbf{y} - \mathbf{y}^{\mathrm{m}}(\mathbf{p}))^T (\mathbf{y} - \mathbf{y}^{m}(\mathbf{p})) - 2(\mathbf{y} - \mathbf{y}^{m}(\mathbf{p}))\mathbf{J}\delta + \delta^T \mathbf{J}^T \mathbf{j}\delta & (9.18) \end{aligned}$$

where $\mathbf{J}$ is the Jacobian matrix of $\mathbf{y}^{\mathrm{m}}(i,\mathbf{p})$. The elements in the matrix is calculated as

$$[\mathbf{J}]_{ij} = \frac{[\mathbf{y}^{m}(i,\mathbf{p})]_i}{[\mathbf{p}]_j}. \tag{9.19}$$

The operator $[\bullet]_k$ in Equation 9.19 represents the k-th element in the vector or matrix. Since we want to minimize the function Equation 9.17, we can take the derivative with respect to $\boldsymbol{\delta}$ and setting the result to zero:

$$\begin{aligned} \frac{\partial l'(\mathbf{p})}{\partial \delta} &= 0 & (9.20) \\ &= -2(\mathbf{y} - \mathbf{y}^{\mathrm{m}}(\mathbf{p}))\mathbf{J} + \mathbf{J}^T \mathbf{J}\boldsymbol{\delta} & (9.21) \\ \boldsymbol{\delta} &= 2(\mathbf{J}^T \mathbf{J})^{-1} [\mathbf{y} - \mathbf{y}^{\mathrm{m}}(\mathbf{p})]\mathbf{J}. & (9.22) \end{aligned}$$

We apply Equation 9.20 for every iteration to replace $\mathbf{p}$ to $\mathbf{p}+\delta$, until $\mathbf{p}$ has converged.

9.3.3.2 Cramér-Rao Lower Bound (CRLB)

Once an estimate $\hat{\mathbf{p}}_{\mathrm{MLE}}$ has found, the CRLB Torrieri [1990] may be used to derive an ellipsoidal confidence region in the parameter space. Assuming that $\hat{\mathbf{p}}_{\mathrm{MLE}}$ is unbiased, and introducing the estimator covariance matrix Σ, one has

$$\Sigma \succcurlyeq \mathbf{J}^{-1}(\mathbf{p}), \tag{9.23}$$

where $\mathbf{J}(\mathbf{p})$ is the $\dim(\mathbf{p}) \times \dim(\mathbf{p})$ Fisher information matrix, with components

$$[\mathbf{J}(\mathbf{p})]_{i_1,i_2} = -E\left[\frac{\partial^2 \ln \pi(\mathbf{y}|\mathbf{p})}{\partial p_{i_1} \partial p_{i_2}}\right]. \tag{9.24}$$

The variable $\mathbf{y}$ demotes the observation vector in Equation 10.16, and $\pi(\mathbf{y}|\mathbf{p})$ is the join conditional probability density function of the observation given $\mathbf{p}$. The variable p_{i_1} and p_{i_2} are the i_1-th and i_2-th elements in $\mathbf{p}$, respectively. When applying CRLB, we need to determine the probability function of the noise. We assume the noises are independent and identically distributed Gaussian random variables. The probability density function of an observation for a RSS measurement in Equation 9.1 is

$$\pi_{i,j}(y(i,j)|\mathbf{p}) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(y(i,j)-y^{\mathrm{m}}(i,\mathbf{p}))^2}{2\sigma^2}}, \tag{9.25}$$

where i represents the anchor node who measures the RSS value of a agent node. Since all probability density functions for different RSS value measured by anchor nodes are independent to each other. We can multiply all probability density function to get the joint conditional probability density function.

$$\pi(\mathbf{y}|\mathbf{p}) = \prod_{i=1}^{n_a} \prod_{j=1}^{n} \pi_{i,j}(y(i,j)|\mathbf{p}) \tag{9.26}$$

Since each agent is localized individually. For a specific agent. By combining Equation 9.25, Equation 9.26 and Equation 9.1, the joint conditional probability density function of the observations is

$$\pi(\mathbf{y}|\mathbf{p}) = \prod_{a=1}^{N_a} \frac{1}{\sqrt{2\pi}\sigma} \exp\left\{-\frac{(y(a) - P_0 + 10\gamma_{\mathrm{p}} \log_{10} \frac{\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}_i\|}{d_0})^2}{2\sigma^2}\right\}. \tag{9.27}$$

After the joint conditional probability density function of the observations is build,

the fisher information matrix for localization is derived. We demonstrate fisher information matrix using three unknown variables for $\mathbf{p}$.

$$\mathbf{J}(\mathbf{p}=[\theta_{0,1},\theta_{0,2},P_0]^T)=-E\left(\begin{bmatrix} \frac{\partial^2\ln\pi(\mathbf{y}|\mathbf{p})}{\partial\theta_{0,1}\partial\theta_{0,1}} & \frac{\partial^2\ln\pi(\mathbf{y}|\mathbf{p})}{\partial\theta_{0,2}\partial\theta_{0,1}} & \frac{\partial^2\ln\pi(\mathbf{y}|\mathbf{p})}{\partial P_0\partial\theta_{0,1}} \\ \frac{\partial^2\ln\pi(\mathbf{y}|\mathbf{p})}{\partial\theta_{0,1}\partial\theta_{0,2}} & \frac{\partial^2\ln\pi(\mathbf{y}|\mathbf{p})}{\partial\theta_{0,2}\partial\theta_{0,2}} & \frac{\partial^2\ln\pi(\mathbf{y}|\mathbf{p})}{\partial P_0\partial\theta_{0,2}} \\ \frac{\partial^2\ln\pi(\mathbf{y}|\mathbf{p})}{\partial\theta_{0,1}\partial P_0} & \frac{\partial^2\ln\pi(\mathbf{y}|\mathbf{p})}{\partial\theta_{0,2}\partial P_0} & \frac{\partial^2\ln\pi(\mathbf{y}|\mathbf{p})}{\partial P_0\partial P_0} \end{bmatrix}\right) \tag{9.28}$$

and

$$\mathbf{J}(\mathbf{p}=[\theta_{0,1},\theta_{0,2},\gamma_{\mathrm{p}}]^T)=-E\left(\begin{bmatrix} \frac{\partial^2\ln\pi(\mathbf{y}|\mathbf{p})}{\partial\theta_{0,1}\partial\theta_{0,1}} & \frac{\partial^2\ln\pi(\mathbf{y}|\mathbf{p})}{\partial\theta_{0,2}\partial\theta_{0,1}} & \frac{\partial^2\ln\pi(\mathbf{y}|\mathbf{p})}{\partial\gamma_{\mathrm{p}}\partial\theta_{0,1}} \\ \frac{\partial^2\ln\pi(\mathbf{y}|\mathbf{p})}{\partial\theta_{0,1}\partial\theta_{0,2}} & \frac{\partial^2\ln\pi(\mathbf{y}|\mathbf{p})}{\partial\theta_{2}\partial\theta_{0,2}} & \frac{\partial^2\ln\pi(\mathbf{y}|\mathbf{p})}{\partial\gamma_{\mathrm{p}}\partial\theta_{0,2}} \\ \frac{\partial^2\ln\pi(\mathbf{y}|\mathbf{p})}{\partial\theta_{0,1}\partial\gamma_{\mathrm{p}}} & \frac{\partial\ln\pi(\mathbf{y}|\mathbf{p})}{\partial\theta_{0,2}\partial\gamma_{\mathrm{p}}} & \frac{\partial^2\ln\pi(\mathbf{y}|\mathbf{p})}{\partial\gamma_{\mathrm{p}}\partial\gamma_{\mathrm{p}}} \end{bmatrix}\right). \tag{9.29}$$

The derivative elements in $\mathbf{J}(\mathbf{p}=[\theta_{0,1},\theta_{0,2},P_0]^T)$ and $\mathbf{J}(\mathbf{p}=[\theta_{0,1},\theta_{0,2},\gamma_{\mathrm{p}}]^T)$ are

$$\frac{\partial^2\ln\pi(\mathbf{y}|\mathbf{p})}{\partial\theta_1\partial\theta_1}=\frac{(10\gamma_{\mathrm{p}})^2}{\sigma\ln 10}\sum_{a=1}^{N_a}\frac{(\theta_{0,1}-\theta_{a,1})^2}{\|\boldsymbol{\theta}_0-\boldsymbol{\theta}_a\|^4}, \tag{9.30}$$

$$\frac{\partial^2\ln\pi(\mathbf{y}|\mathbf{p})}{\partial\theta_2\partial\theta_2}=\frac{(10\gamma_{\mathrm{p}})^2}{\sigma\ln 10}\sum_{a=1}^{N_a}\frac{(\theta_{0,2}-\theta_{a,2})^2}{\|\boldsymbol{\theta}_0-\boldsymbol{\theta}_a\|^4}, \tag{9.31}$$

$$\frac{\partial^2\ln\pi(\mathbf{y}|\mathbf{p})}{\partial\theta_{0,1}\partial\theta_{0,2}}=\frac{\partial^2\ln\pi(\mathbf{y}|\mathbf{p})}{\partial\theta_{0,2}\partial\theta_{0,1}}=\frac{(10\gamma_{\mathrm{p}})^2}{\sigma\ln 10}\sum_{a=1}^{N_a}\frac{(\theta_{0,1}-\theta_{a,1})(\theta_{0,2}-\theta_{a,2})}{\|\boldsymbol{\theta}_0-\boldsymbol{\theta}_a\|^4}, \tag{9.32}$$

$$\frac{\partial^2\ln\pi(\mathbf{y}|\mathbf{p})}{\partial\theta_{0,1}\partial P_0}=\frac{\partial^2\ln\pi(\mathbf{y}|\mathbf{p})}{\partial P_0\partial\theta_{0,1}}=\frac{10\gamma_{\mathrm{p}}}{\sigma^2\ln 10}\sum_{a=1}^{N_a}\frac{(\theta_{0,1}-\theta_{a,1})}{\|\boldsymbol{\theta}_0-\boldsymbol{\theta}_a\|^2}, \tag{9.33}$$

$$\frac{\partial^2\ln\pi(\mathbf{y}|\mathbf{p})}{\partial\theta_{0,2}\partial P_0}=\frac{\partial^2\ln\pi(\mathbf{y}|\mathbf{p})}{\partial P_0\partial\theta_{0,2}}=\frac{10\gamma_{\mathrm{p}}}{\sigma^2\ln 10}\sum_{i=1}^{N_a}\frac{(\theta_{0,2}-\theta_{a,2})}{\|\boldsymbol{\theta}_0-\boldsymbol{\theta}_a\|^2}, \tag{9.34}$$

$$\frac{\partial^2\ln\pi(\mathbf{y}|\mathbf{p})}{\partial P_0\partial P_0}=\sum_{a=1}^{N_a}\sigma^{-2}, \tag{9.35}$$

$$\frac{\partial^2 \ln \pi(\mathbf{y}|\mathbf{p})}{\partial \gamma_{\mathrm{p}} \partial \theta_{0,1}} = \frac{\partial^2 \ln \pi(\mathbf{y}|\mathbf{p})}{\partial \theta_{0,1} \partial \gamma_{\mathrm{p}}} = \frac{100\gamma_{\mathrm{p}}}{\sigma^2 \ln(10)} \sum_{a=1}^{N_a} \frac{(\theta_{0,1} - \theta_{a,1}) \ln(\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}_a\|)}{\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}_a\|^2}, \tag{9.36}$$

$$\frac{\partial^2 \ln \pi(\mathbf{y}|\mathbf{p})}{\partial \gamma_{\mathrm{p}} \partial \theta_{0,2}} = \frac{\partial^2 \ln \pi(\mathbf{y}|\mathbf{p})}{\partial \theta_{0,2} \partial \gamma_{\mathrm{p}}} = \frac{100\gamma_{\mathrm{p}}}{\sigma^2 \ln(10)} \sum_{a=1}^{N_a} \frac{(\theta_{0,2} - \theta_{a,2}) \ln(\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}_a\|)}{\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}_a\|^2}, \tag{9.37}$$

$$\frac{\partial^2 \ln \pi(\mathbf{y}|\mathbf{p})}{\partial P_0 \partial \gamma_{\mathrm{p}}} = \frac{\partial^2 \ln p(\mathbf{y}|\mathbf{p})}{\partial \gamma_{\mathrm{p}} \partial P_0} = 0, \tag{9.38}$$

and

$$\frac{\partial^2 \ln \pi(\mathbf{y}|\mathbf{p})}{\partial \gamma_{\mathrm{p}} \partial \gamma_{\mathrm{p}}} = (\frac{10}{\sigma \ln(10)})^2 \sum_{a=1}^{N_a} \frac{\ln(\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}_a\|^2)}{\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}_a\|^2}. \tag{9.39}$$

As long as we have $\mathbf{J}(\mathbf{p} = [\theta_{0,1}, \theta_{0,2}, P_0]^T)$ and $\mathbf{J}(\mathbf{p} = [\theta_{0,1}, \theta_{0,2}, \gamma_{\mathrm{p}}]^T)$, we can obtain the minimum covariance matrix of Σ by calculating the inversion of $\mathbf{J}(\mathbf{p} = [\theta_{0,1}, \theta_{0,2}, P_0]^T)$ and $\mathbf{J}(\mathbf{p} = [\theta_{0,1}, \theta_{0,2}, \gamma_{\mathrm{p}}]^T)$, then we can illustrate CRLB as an error ellipse. An error ellipse is often used as a visual aid or performance measure to depict the accuracy or performance of an estimator, or other stochastic system. It implies that there are two (or more) variables involved, and that they are normally (Gaussian) distributed with known covariance.

Let us start with an 2-D zero mean random variable $\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$. We assume the covariance matrix of $\mathbf{x}$ is $\boldsymbol{C}_x = \begin{bmatrix} \sigma_1^2 & \sigma_1\sigma_2\rho \\ \sigma_1\sigma_2\rho & \sigma_2^2 \end{bmatrix}$. It has the probability density function:

$$p(\mathbf{x}) = \frac{1}{(2\pi)^2 \sqrt{|\mathbf{C}_x|}} \exp(-0.5\mathbf{x}^T \mathbf{C}_x^{-1} \mathbf{x}), \tag{9.40}$$

where $\mathbf{C}_x^{-1}$ is:

$$\mathbf{C}_x^{-1} = \begin{bmatrix} \sigma_1^{-2} & \frac{-\rho}{\sigma_1\sigma_2} \\ \frac{-\rho}{\sigma_1\sigma_2} & \sigma_2^{-2} \end{bmatrix}. \tag{9.41}$$

Combining Equation 9.40 and Equation 9.41, we get

$$\text{pdf}(x_1, x_2) = \frac{1}{2\pi\sigma_1\sigma_2\sqrt{1-\rho}} \exp\left(\frac{-1}{2(1-\rho^2)}\left[\frac{1}{\sigma_1^2}x_1^2 - \frac{2\rho}{\sigma_1\sigma_2}x_1x_2 + \frac{1}{\sigma_1^2}x_2^2\right]\right). \quad (9.42)$$

We fix the value of the probability, i.e., $p(x_1, x_2) = p^*$, so we can obtain

$$\text{pdf}^* = \frac{1}{2\pi\sigma_1\sigma_2\sqrt{1-\rho}} \exp\left(\frac{-1}{2(1-\rho^2)}\left[\frac{1}{\sigma_1^2}x_1^2 - \frac{2\rho}{\sigma_1\sigma_2}x_1x_2 + \frac{1}{\sigma_1^2}x_2^2\right]\right), \quad (9.43)$$

which can be rewritten as

$$k^2 = \frac{-1}{2(1-\rho^2)}\left[\frac{1}{\sigma_1^2}x_1^2 - \frac{2\rho}{\sigma_1\sigma_2}x_1x_2 + \frac{1}{\sigma_1^2}x_2^2\right], \quad (9.44)$$

where

$$k = 2\ln(p^* 2\pi\sigma_1\sigma_2\sqrt{1-\rho}). \quad (9.45)$$

We can observe that Equation 9.44 is an ellipse with x_1 and x_2 being coordinate axes. A more practical way to describe Equation 9.44 for n random variables is in terms of matrix. To draw the ellipse to unknown parameters of the localization problem, we set $\mathbf{x} = \mathbf{p} - \mathbf{p}^*$, so that $\mathbf{x}$ is a vector of random variables with zero means.

$$k^2 = \mathbf{x}^T\mathbf{C}_x\mathbf{x}. \quad (9.46)$$

Given a fixed value pdf* and a covariance matrix $\mathbf{C}_x$, we can find a set of point $\mathbf{b}$ such that $\{\mathbf{b}|k^2 = \mathbf{b}^T\mathbf{C}_x\mathbf{b}\}$ to illustrate the lower bound of the confidence level ($n = 1$ for an interval, $n = 2$ for an error ellipse, $n = 3$ for an ellipsoid).

To find the set $\mathbf{b}$, Torrieri [1990] apply a transformation matrix $\mathbf{A}$ to let $\boldsymbol{z} = \mathbf{A}\mathbf{x}$, where $\boldsymbol{z}$'s components are independent standard normal. i.e., no correlation between any of elements of $\boldsymbol{z}$, and the variations of the elements in $\boldsymbol{z}$ are 1. After finding $\mathbf{A}$, the desired $\mathbf{b}$ can easily be found using $\boldsymbol{b} = \mathbf{A}^{-1}\mathbf{a}$, where $\mathbf{a}$ is a set of point of a circle with unit radius.

To find the transformation matrix $\mathbf{A}$, we first apply the covariance of $\text{cov}(\boldsymbol{z}) = \text{I}$.

Algorithm 9.1 Error Ellipse

1: Procedure for finding error ellipse
2: Input: desired confidence probability pdf*, $\mathbf{C}_x$
3: Output: $\mathbf{b}$
4: Compute the eigenvalues $\mathbf{D}$ and eigenvectors $\mathbf{V}$.
5: Compute the transform matrix, $\mathbf{A} = \mathbf{D}^{-\frac{1}{2}}\mathbf{V}$, and its inverse, $\mathbf{A}^{-1}$.
6: Compute k according to Equation 9.45.
7: Compute some points around a circle $\boldsymbol{a} = k^2[\ \begin{matrix}\cos(\boldsymbol{\theta}) \\ \sin(\boldsymbol{\theta})\end{matrix}\ ]$, $\boldsymbol{\theta} = [0, ..., 2\pi]^T$.
8: Compute the error ellipse, $\mathbf{b} = \mathbf{A}^{-1}\mathbf{a}$.
9: Add the offset by the mean of the random variable $\mathbf{x}$.

Recall the properties of multivariate transformation. If $\mathbf{x}$ is a random vector of n variables with covariance $\mathbf{C}_x$, and $\mathbf{A}$ is an $n \times n$ matrix, then $\boldsymbol{z} = \mathbf{A}\mathbf{x}$ is a random vector of n variables with covariance $\mathbf{A}\mathbf{C}_x\mathbf{A}^T$. With this properties, we obtain

$$\text{cov}(\boldsymbol{z}) = \mathbf{A}\mathbf{C}_x\mathbf{A}^T = I. \tag{9.47}$$

Then, we turn to Eigendecomposition, which allows any square matrix $\mathbf{C}_x$ to be expressed as $\mathbf{C}_x = \mathbf{V}\mathbf{D}\mathbf{V}^{-1}$, where $\mathbf{V}$ is an orthonormal square matrix of eigenvectors, and $\mathbf{D}$ is a diagonal matrix of eigenvectors. Then we have

$$\text{cov}(\boldsymbol{z}) = \mathbf{A}\mathbf{V}\mathbf{D}\mathbf{V}^{-1}\mathbf{A}^T = \mathbf{A}\mathbf{V}\mathbf{D}^{\frac{1}{2}}\mathbf{D}^{\frac{1}{2}}\mathbf{V}^{-1}\mathbf{A}^T = I, \tag{9.48}$$

which satisfied if $\mathbf{D}^{\frac{1}{2}}\mathbf{V}\mathbf{A}^T = \mathbf{I}$. Thus, setting

$$\mathbf{A} = \mathbf{D}^{-\frac{1}{2}}\mathbf{V} \tag{9.49}$$

allows $\boldsymbol{z} = \mathbf{A}\mathbf{x}$ to have a standard normal distribution. Working with $\boldsymbol{z}$ is much easier than working with $\mathbf{x}$ because $\boldsymbol{z}$ has a standard normal distribution, and the error ellipse is a circle. on the current page summarizes the procedure for finding the error ellipse.

We also want to use CRLB to calculate the lower bound of the location error (e).

For a 2-D dimension, we can obtain the lower bound of the variance of $\theta_{0,1}$ and $\theta_{0,2}$, which are $[\mathbf{J}^{-1}]_{1,1}, [\mathbf{J}^{-1}]_{2,2}$, respectively. (The $[\mathbf{M}]_{1,1}$ means the element in first row and first column; $[\mathbf{M}]_{2,2}$ means the element in the second row and second column)

$$\begin{aligned}\sqrt{E(e^2)} &= \sqrt{E[(\theta^*_{0,1} - \theta_{0,1})^2] + E[(\theta^*_{0,2} - \theta_{0,2})^2]} \\ &= \sqrt{[\mathbf{J}^{-1}]_{1,1} + [\mathbf{J}^{-1}]_{2,2}}. \end{aligned} \tag{9.50}$$

CRLB evaluates the lower bound of the minimum estimator.

9.3.4 Semidefinite programming (SDP) with Cramér-Rao Lower Bound

The approach considered in Vaghefi et al. [2013] consists in solving the localization problem via semidefinite programming. The confidence region is then evaluated considering confidence ellipsoid associated to the CRLB. This approach is briefly recalled here.

Only $\boldsymbol{\theta}_0$ and P_0 are assumed to be unknown. First, Equation 9.1 is reformulated as

$$\log_{10} d_i^2 \lambda_i = \frac{P_0}{5\gamma_{\mathrm{P}}} + \frac{w(i,k)}{5\gamma_{\mathrm{P}}}, \tag{9.51}$$

where $d_i = \|\boldsymbol{\theta}_0 - \boldsymbol{\theta}_i\|$ and $\lambda_i = 10^{y(i,k)/5\gamma_{\mathrm{P}}}$. Introducing $\alpha = 10^{P_0/5\gamma_{\mathrm{P}}}$, one deduces that

$$d_i^2 \lambda_i = \alpha 10^{w(i,k)/5\gamma_{\mathrm{P}}}. \tag{9.52}$$

Assuming that the noise is sufficiently small, the right-hand side of Equation 9.52 is expanded as

$$d_i^2 \lambda_i = \alpha + \epsilon(i,k), \tag{9.53}$$

where $\epsilon(i,k) = \frac{\ln 10}{5\gamma_{\mathrm{P}}} w(i,k)$.

An estimator for $\mathbf{p} = \left(P_0, \boldsymbol{\theta}_0^T\right)^T$ is then expressed as

$$\widehat{\mathbf{p}} = \arg\min_{P_0,\boldsymbol{\theta}_0} \sum_{i=1\ldots n_a, k=1\ldots n} \left(d_i^2\lambda_i - \alpha\right)^2, \tag{9.54}$$

Finding $\widehat{\mathbf{p}}$ is difficult in general, since the cost function is non-linear and non-convex. To address this issue, Vaghefi et al. [2013] introduce the auxiliary variable $z = \theta_0^T\theta_0$ to get the SDP problem

$$\left(\widehat{\alpha}, \widehat{z}, \widehat{\boldsymbol{\theta}}_0\right) = \min_{\alpha,z,\boldsymbol{\theta}_0} \sum_{i=1\ldots n_a, k=1\ldots n} \left(d_i^2\lambda_i - \alpha\right)^2 \tag{9.55}$$

$$\text{subject to } d_i^2 = \begin{bmatrix} \boldsymbol{\theta}_i \\ -1 \end{bmatrix}^T \begin{bmatrix} \mathrm{I}_2 & \boldsymbol{\theta}_0 \\ \boldsymbol{\theta}_0^T & z \end{bmatrix} \begin{bmatrix} \boldsymbol{\theta}_i \\ -1 \end{bmatrix}, \tag{9.56}$$

$$\begin{bmatrix} \mathrm{I}_2 & \boldsymbol{\theta}_0 \\ \boldsymbol{\theta}_0^T & z \end{bmatrix} \succeq 0. \tag{9.57}$$

The solution of this problem may be obtained with standard SDP solvers such as SDPT3 Toh et al. [1999]. An estimate $\widehat{\mathbf{p}}_{\mathrm{SDP}}$ is then easily deduced from $\widehat{\alpha}, \widehat{z}$, and $\widehat{\boldsymbol{\theta}}_0$. The confidence region in this case is again obtained from the CRLB.

9.4 Conclusion

In this chapter, state-of-the-art algorithms are described. Some of the methods [Vaghefi et al., 2013, Torrieri, 1990, Ziskind and Wax, 1988] allows one to apply maximum likelihood (ML), or maximum *a posteriori* (MAP) estimation techniques to the localization problem. The main difficulty is that the cost function associated to the ML, or MAP estimation problem may be non-convex, especially when the path-loss exponent, as well as the reference transmission power have to be estimated jointly with the location.

In the light of this, bounded error methods Mourad et al. [2009], Léger and Kieffer

[2010], Mourad et al. [2011a] are proposed, where the RSS measurement errors are assumed to be bounded with known bounds. One tries to evaluate the set of all source locations that are consistent with the measurement model and the noise bounds. The resulting set is guaranteed to contain the true location of the source node, provided that the hypotheses on the noise bounds are satisfied. Nevertheless, in practice, noise samples do not really follow a normal or lognormal distribution, and the bounds considered in bounded-error estimation are either often violated because they are too small and result in an empty set estimate, or lead to huge solution sets when the bounds are too large. In the next chapter, we introduce our proposed technique, which only uses mild assumptions on the noise, so the localization accuracy and the robustness are improved.

Chapter 10

Proposed LSCR Approach

This chapter proposes the localization method based on LSCR [Campi and Weyer, 2005, 2006]. The confidence region is defined and characterized using the tools from Interval analysis.

10.1 Introduction

LSCR [Campi and Weyer, 2005, 2006] defines a set $\mathbb{P} \subset \mathbb{P}_0$ that can be proved to contain $\mathbf{p}^*$ with a prescribed probability. The main idea of LSCR is to consider various estimates of the correlations of the prediction residuals, *i.e.*, the differences between the measurements and model outputs for some value $\mathbf{p} \in \mathbb{P}_0$. Then regions of $\mathbb{P}_0$ are eliminated when too many correlation estimates have the same sign, which contradicts the hypothesis that the noise samples are independent and symmetrically distributed. The regions that were not eliminated constitute the non-asymptotic confidence region $\mathbb{P}$. In this chapter, LSCR for localization is analyzed. Different configuration of the correlation functions in LSCR are discussed. Moreover, the LSCR is compared to different localization approaches:

- Maximum likelihood estimator (MLE)
- Semidefinite programming (SDP)

- Bounded-error approach (BE)
- Robust Bounded-error approach (RBE)

The goal of our experiments is to show the advantages of LSCR for localizing agents using the measurements from anchors.

10.2 Confidence Regions as Defined by LSCR

First consider the vectors

$$\mathbf{y}(i) = (y(i,1), \ldots, y(i,n))^T \tag{10.1}$$

$$\mathbf{y}_{\mathrm{m}}(i,\mathbf{p}) = (y^{\mathrm{m}}(i,\mathbf{p}), \ldots, y^{\mathrm{m}}(i,\mathbf{p}))^T \tag{10.2}$$

gathering the RSS measurements and the model output for the i-th anchor node. Then, consider the vectors

$$\mathbf{y} = \left(\mathbf{y}^T(1), \ldots, \mathbf{y}^T(n_{\mathrm{a}})\right)^T \tag{10.3}$$

$$\mathbf{y}_{\mathrm{m}}(\mathbf{p}) = \left(\mathbf{y}_{\mathrm{m}}^T(1,\mathbf{p}), \ldots, \mathbf{y}_{\mathrm{m}}^T(n_{\mathrm{a}},\mathbf{p})\right)^T \tag{10.4}$$

gathering in n_{y} entries the $n_a \times n$ measurements and $n_a \times n$ model outputs. Considering some value $\mathbf{p}$ of the parameter vector, one may introduce the prediction residual

$$\mathbf{w}(\mathbf{p}) = \mathbf{y} - \mathbf{y}_{\mathrm{m}}(\mathbf{p}). \tag{10.5}$$

When $\mathbf{p} = \mathbf{p}^*$, $\mathbf{w}(\mathbf{p}^*)$ contains the $n_a \times n$ realizations of the measurement noise for all anchors at all measurement time instants.

To define a NACR, LSCR introduces three integers $s > 0$, $m > 0$, and $q > 0$, and a vector $\mathbf{e} = (e_0, ..., e_s)^T$ of non-negative integers such that at least one e_j , $0 \leq j \leq s$ is odd. Considering some $p \in P_0$, for $t = 1, ..., ny - s$, one can evaluate

$$c_{t,\mathbf{e}}^{\mathrm{e}}(\mathbf{p}) = \prod w_{t+i}^{e_i}(\mathbf{p}) \tag{10.6}$$

where $w_t(\mathbf{p})$ is the t-th component of $\mathbf{w}(\mathbf{p})$. Then, using Equation 10.6, one may compute several estimates of the correlation function of the prediction residuals, for $j = 1, \ldots, m$

$$s_{j,\mathbf{e}}(\mathbf{p}) = \frac{1}{\left|\mathbb{I}_j^{n_y-s}\right|} \sum_{\ell \in \mathbb{I}_j^{n_y-s}} c_{\ell,\mathbf{e}}(\mathbf{p}), \tag{10.7}$$

where $\mathbb{I}_j^{n_y-s}$ is a subset of the set $\mathbb{I}^{n_y-s} = \{1, \ldots, n_y - s\}$ of indexes such that the collection $\mathbb{G} = \left\{\mathbb{I}_1^{n_y-s}, \ldots, \mathbb{I}_m^{n_y-s}\right\}$ of these subsets forms a group under symmetric difference, *i.e.*, $\forall \mathbb{I}_j \in \mathbb{G}^{n_y-s}$ and $\forall \mathbb{I}_{j'} \in \mathbb{G}^{n_y-s}$, one has $(\mathbb{I}_j \cup \mathbb{I}_{j'}) - (\mathbb{I}_j \cap \mathbb{I}_{j'}) \in \mathbb{G}^{n_y-s}$. A procedure to build such $\mathbb{G}^{n_y-s}$ is described in section 10.3. In Equation 10.7, $\left|\mathbb{I}_j^{n_y-s}\right|$ is the cardinal number of $\mathbb{I}_j^{n_y-s}$.

Then, the set

$$\mathbb{P}_{\mathbf{e},q,m} = \left\{ \mathbf{p} \in \mathbb{P}_0 | \sum_{i=1}^{m} \tau_{i,\mathbf{e}}^{-}(\mathbf{p}) \geqslant q \text{ and } \sum_{i=1}^{m} \tau_{i,\mathbf{e}}^{+}(\mathbf{p}) \geqslant q \right\}, \tag{10.8}$$

where

$$\tau_{i,\mathbf{e}}^{-}(\mathbf{p}) = \begin{cases} 0 & \text{if } s_{i,\mathbf{e}}(\mathbf{p}) < 0 \\ B_{0.5} & \text{if } s_{i,\mathbf{e}}(\mathbf{p}) = 0 \\ 1 & \text{if } s_{i,\mathbf{e}}(\mathbf{p}) > 0 \end{cases} \tag{10.9}$$

and

$$\tau_{i,\mathbf{e}}^{+}(\mathbf{p}) = \begin{cases} 0 & \text{if } s_{i,\mathbf{e}}(\mathbf{p}) > 0 \\ 0 & \text{if } s_{i,\mathbf{e}}(\mathbf{p}) = 0 \\ 1 & \text{if } s_{i,\mathbf{e}}(\mathbf{p}) < 0 \end{cases} \tag{10.10}$$

has been shown in Campi and Weyer [2005] to be such that

$$\Pr(\mathbf{p}^* \in \mathbb{P}_{\mathbf{e},q,m}) = 1 - \frac{2q}{m}. \tag{10.11}$$

In (12) and (13), $B_{0.5}$ is a Bernoulli random variable with $\Pr(B_{0.5} = 0) = \Pr(B_{0.5} = 1) = 0.5$ to solve ties.

The set $\mathbb{P}_{\mathbf{e},q,m}$ contains all values of $\mathbf{p} \in \mathbb{P}_0$ such that at least q out of the m $s_{i,\mathbf{e}}(\mathbf{p})$ are positive and q are negative. It is a NACR containing $\mathbf{p}^*$ with a probability that is exactly $1 - \frac{2q}{m}$.

10.3 Construction of the groups $\mathbb{G}^N$

Original LSCR uses $\mathbb{G}^N$ to construct correlation functions $s_{j,\mathbf{e}}(\mathbf{p})$ in Equation 10.7. With the properties of $\mathbb{G}^N$, the probability of the confidence region inEquation 10.11 holds. The analysis of the behavior of the LSCR approach in the considered case where different measurement models have to be considered requires a slight adaptation of the construction of the groups $\mathbb{G}^N$ used by the LSCR approach as presented in Dalai et al. [2007b].

Consider the set $\mathbb{I}^N = 1, ..., N$ and an integer $m = 2^\mu$.To build a group $\mathbb{G}^N$ for the symmetric difference operator, as in Dalai et al. [2007b], one considers first Gordon's iterative construction [Gordon, 1974]. One starts with $\mathbf{R}(1) = [1]$ and evaluate

$$\begin{bmatrix} \mathbf{R}(k-1) & \mathbf{R}(k-1) & \mathbf{0} \\ \mathbf{R}(k-1) & \mathbf{J} - \mathbf{R}(k-1) & \mathbf{1} \\ \mathbf{0}^T & \mathbf{1}^T & \mathbf{1} \end{bmatrix}, k = 1, \ldots, \mu, \tag{10.12}$$

where $\mathbf{J}$ is the all-one matrix of appropriate size, and $\mathbf{0}$ and $\mathbf{1}$ are respectively the all-zero and all-one vectors of appropriate size. Then one considers

$$\mathbf{Q}^{m-1} = \begin{bmatrix} \mathbf{R}(k) \\ \mathbf{0}^T \end{bmatrix} = [\mathbf{q}_1, \ldots, \mathbf{q}_{m-1}], \tag{10.13}$$

where the $\mathbf{q}_i$s are the column vectors is repeated $\lceil N/(m-1) \rceil$ times so that $\mathbf{Q}$ has at least N columns. The matrix $\mathbf{G}$ obtained selecting the first N columns of $\mathbf{Q}$ is the incidence matrix of the group $\mathbb{G}^N$. $\mathbf{Q}$ is such that $Q_{i,j} = 1$ if $j \in \mathbb{I}_i^N$.

10.4 Characterizing Confidence Regions

The characterization of confidence regions as defined by LSCR is performed in Campi and Weyer [2005] by sampling points in $\mathbb{P}_0$ and determining whether they belong to $\mathbb{P}_{\mathbf{e},q,m}$. One thus gets an *inner approximation* of $\mathbb{P}_{\mathbf{e},q,m}$, consisting of a union of points in $\mathbb{P}_0$ which have been proved to belong to $\mathbb{P}_{\mathbf{e},q,m}$. When the confidence region is small, consists of several disconnected subsets, and when the number of sampling points is not enough, there is a risk to be unable to find any point in $\mathbb{P}_{\mathbf{e},q,m}$, except with a fine gridding of $\mathbb{P}_0$.

An alternative approach based on interval analysis has been proposed in Kieffer and Walter [2014]. Its main property is to yield inner and outer approximations of $\mathbb{P}_{\mathbf{e},q,m}$ consisting of unions of non-overlapping boxes. This approach is briefly recalled in what follows.

10.4.1 SIVIA

Characterizing $\mathbb{P}_{\mathbf{e},q,m}$ may be formulated as a set-inversion Jaulin and Walter [1993b] problem:

$$\mathbb{P}_{\mathbf{e},q,m} = \{\mathbf{p} \in \mathbb{P}_0 | \mathbf{f}_{r,m}(\mathbf{p}) \in \mathbb{Y}_q\} = \mathbb{P}_0 \cap \mathbf{f}_{\mathbf{e},m}^{-1}\left(\mathbb{Y}_q\right), \tag{10.14}$$

where

$$\mathbf{f}_{\mathbf{e},m}(\mathbf{p}) = \begin{bmatrix} f_{\mathbf{e},m}^{+}(\mathbf{p}) \\ f_{\mathbf{e},m}^{-}(\mathbf{p}) \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{m} \tau_{i,\mathbf{e}}^{+}(\mathbf{p}) \\ \sum_{i=1}^{m} \tau_{i,\mathbf{e}}^{-}(\mathbf{p}) \end{bmatrix}, \tag{10.15}$$

and

$$\mathbb{Y}_q = \begin{bmatrix} \mathbb{Y}_q^{+} \\ \mathbb{Y}_q^{-} \end{bmatrix} = \begin{bmatrix} [q, \infty[\\ [q, \infty[\end{bmatrix}, \tag{10.16}$$

which may be efficiently solved via interval analysis [Jaulin et al., 2001, Moore, 1966] using the SIVIA algorithm [Jaulin et al., 2001]. For that purpose, inclusion functions for the $\mathbf{f}_{\mathbf{e},m}$ are required. SIVIA in Algorithm 10.1 recursively partitions $\mathbb{P}_0$ into 3 different kinds of boxes: boxes proved to belong to $\mathbb{P}_{\mathbf{e},q,m}$, boxes proved to have no intersection with $\mathbb{P}_{\mathbf{e},q,m}$, and undetermined boxes for which no conclusion can be

Algorithm 10.1 LSCR with SIVIA

Require: $[\mathbf{f}_{\mathbf{e},m}]([\mathbf{p}]), [\mathbf{p}], q, \varepsilon$
Ensure: $\underline{\mathbb{P}}_{\mathbf{e},q,m}, \overline{\mathbb{P}}_{\mathbf{e},q,m}$
1: if $[\mathbf{f}_{\mathbf{e},m}]([\mathbf{p}]) \cap \mathbb{Y}_q = \emptyset$ return;
2: if $[\mathbf{f}_{\mathbf{e},m}]([\mathbf{p}]) \subset \mathbb{Y}_q$ then $\underline{\mathbb{P}}_{\mathbf{e},q,m} := \underline{\mathbb{P}}_{\mathbf{e},q,m} \cup [\mathbf{p}]$; $\overline{\mathbb{P}}_{\mathbf{e},q,m} := \overline{\mathbb{P}}_{\mathbf{e},q,m} \cup [\mathbf{p}]$; return;
3: if width$([\mathbf{p}]) < \varepsilon$ then $\overline{\mathbb{P}}_{\mathbf{e},q,m} := \overline{\mathbb{P}}_{\mathbf{e},q,m} \cup [\mathbf{p}]$; return;
4: SIVIA$([\mathbf{f}_{\mathbf{e},m}]([\mathbf{p}]), L[\mathbf{p}], q, \varepsilon, \underline{\mathbb{P}}_{\mathbf{e},q,m}, \overline{\mathbb{P}}_{\mathbf{e},q,m})$
5: SIVIA$([\mathbf{f}_{\mathbf{e},m}]([\mathbf{p}]), R[\mathbf{p}], q, \varepsilon, \underline{\mathbb{P}}_{\mathbf{e},q,m}, \overline{\mathbb{P}}_{\mathbf{e},q,m})$

obtained. Undetermined boxes $[\mathbf{p}]$ are bisected into two subboxes $L[\mathbf{p}]$ and $R[\mathbf{p}]$ on which SIVIA is applied again, until their width is less than some precision parameter ε.

10.4.2 Contractor

Indetermination often results from range overestimation by inclusion functions. As a consequence, boxes have to be bisected by SIVIA many times to allow one to conclude on the position of the resulting boxes with respect to $\mathbb{P}_{\mathbf{e},q,m}$. This may entail a significant computational complexity, even for a moderate dimension of $\mathbf{p}$. Contractors may be useful in this context.

One may rewrite Equation 10.14 as follows

$$\mathbb{P}_{\mathbf{e},q,m} = \mathbb{P}^+ \cap \mathbb{P}^- \tag{10.17}$$

$$\mathbb{P}^+ \;=\; \{\mathbf{p} \in \mathbb{P}_0 | f^+_{\mathbf{e},m}(\mathbf{p}) \in \mathbb{Y}^+_q\} = \mathbb{P}_0 \cap \left(f^+_{r,m}\right)^{-1} \left(\mathbb{Y}^+_q\right) \tag{10.18}$$

$$\mathbb{P}^- = \{\mathbf{p} \in \mathbb{P}_0 | f^-_{\mathbf{e},m}(\mathbf{p}) \in \mathbb{Y}^-_q\} = \mathbb{P}_0 \cap \left(f^-_{r,m}\right)^{-1} \left(\mathbb{Y}^-_q\right). \tag{10.19}$$

A contractor $C_{f^+_{\mathbf{e},m}, \mathbb{Y}^+_q}$ associated, *e.g.*, with the generic set-inversion problem Equa-

tion 10.18 is a function taking a box $[\mathbf{p}]$ as input and returning a box $C_{f^+_{\mathbf{e},m},\mathbb{Y}^+_q}([\mathbf{p}]) \subset [\mathbf{p}]$ such that $[\mathbf{p}] \cap \mathbb{P}^+ = C_{f^+_{\mathbf{e},m},\mathbb{Y}^+_q}([\mathbf{p}]) \cap \mathbb{P}^+$, so no part of $\mathbb{P}^+$ in $[\mathbf{p}]$ is lost after contraction. Contractors eliminate parts of the candidate box $[\mathbf{p}]$ that do not belong to $\mathbb{P}^+$, without bisection.

Since $f^+_{\mathbf{e},m}$ and $f^-_{\mathbf{e},m}$ are not differentiable, most classical contractors, such as the forward-backward contractor, Newton's contractor, *etc.* [Jaulin et al., 2001] and [Chabert and Jaulin, 2009] cannot be used directly. To use contractors for the characterization of $\mathbb{P}^+$, considering a generic box $[\mathbf{p}]$, one first builds a set of m subboxes of $[\mathbf{p}]$. The i-th subbox is obtained as the output of some contractor $C_{s_{i,\mathbf{e}},[0,\infty[}$, trying to eliminate all $\mathbf{p} \in [\mathbf{p}]$ such that $s_{i,\mathbf{e}}(\mathbf{p}) < 0$, $i = 1, ..., m$. Similarly, for $\mathbb{P}^-$, m subboxes of $[\mathbf{p}]$ are built, each obtained as the output of some contractor $C_{s_{i,\mathbf{e}},]-\infty,0]}$, trying to eliminate all $\mathbf{p} \in [\mathbf{p}]$ such that $s_{i,\mathbf{e}}(\mathbf{p}) > 0$, $i = 1, ..., m$. Assuming that $\mathbf{y}_{\mathrm{m}}(\mathbf{p})$ is differentiable, the forward-backward contractor [Jaulin et al., 2001], the contractor based on the centred form Kieffer and Walter [2014], and the 3BCid contractor [Chabert and Jaulin, 2009] may then be put at work. In a second step, the union of all intersections of q out of the m subboxes is evaluated and has been shown in Kieffer and Walter [2014] to provide a contractor $C_{f^+_{\mathbf{e},m},\mathbb{Y}^+_q}$ associated with the generic set-inversion problem Equation 10.18. A similar procedure may be considered for $C_{f^-_{\mathbf{e},m},\mathbb{Y}^-_q}$ associated with the generic set-inversion problem Equation 10.19.

10.5 Simulation Setup

The simulation setup considered here is the same as that in Vaghefi et al. [2013]. Five anchor nodes ($n_a = 5$) are placed in the corners and in the center of a square of 20 m $\times$ 20 m; they are represented by black circles in the following Table 10.4 and Table 10.5. $n_a = 32$ agents are regularly placed in the square; their location is indicated by crosses. At the beginning of the localization process, each agent broadcasts $n = 10$ times a message containing its identifier. We assume that, using appropriate signaling, there is no collision between messages from different agents. Each anchor measures the RSS (in dBm) of the messages as well as the agent identifier. These

data are then transmitted to a central processing unit to determine an estimate of the position of the agents and its corresponding confidence region.

One has thus $n_{\mathrm{y}} = n_a \times n = 50$ measurements available for each agent. In the simulations, the reference power $P_0 = 30$ dBm at $d_0 = 1$ m is the same for all agents. The path loss exponent is taken as $\gamma_{\mathrm{P}} = 4$. Data are corrupted by two types of noise samples, assumed independent and identically distributed. In a first set of experiments, the noise samples are realizations of iid zero-mean log-normal variables with standard deviation $\sigma_0 = 2$ dBm. In a second set of experiments, the noise samples are realizations of iid Gaussian-Bernouilli-Gaussian (GBG) variables. In the latter case, with a probability $p_0 = 0.9$, the noise is zero-mean log-normal with standard deviation $\sigma_0 = 2$ dBm and with a probability $p_1 = 0.1$, it is zero-mean log-normal with standard deviation $\sigma_1 = 5$ dBm.

Three estimation problems are considered:

1. Only the location $\boldsymbol{\theta}_{0,i}$, $i = 1, \ldots, n_a$ of each agent has to be estimated, γ_{P} and P_0 are assumed to be known. The search box for $\boldsymbol{\theta}_{0,i}$ is $[0, 25] \times [0, 25]$ m^2.

2. $\boldsymbol{\theta}_{0,i}$ and $P_{0,i}$, $i = 1, \ldots, n_a$ have to be determined for each agent, the unknown reference power is not assumed to be the same for all agents, contrary to γ_{P} which is assumed known. The search interval for the $P_0, i = 1, \ldots, n_a$ is $[0 \text{ dBm}, 40 \text{ dBm}]$.

3. $\boldsymbol{\theta}_{0,i}$ and γ_{P}, $i = 1, \ldots, n_a$ have to be determined for each agent, the reference power $P_{0,i}$ is assumed to be known for all agents (*e.g.*, transmitted in the packet broadcast by each agent). The search interval for γ_{P} is $[2, 8]$.

The goal is to get an estimate of the parameters in these three cases, as well as an estimate of the associated confidence region. The proposed LSCR technique is run for each agent taking $q = 3$, $r = 1$, and $m = 63$. To build $\mathbb{G}$ used in Campi and Weyer [2005], Gordon's procedure [Gordon, 1974] is used, but each $\mathbb{I}_j \in \mathbb{G}$ consists only of the $n_{\mathrm{y}} - r$ elements provided by the original procedure. The choice $m = 63$

allows an exploitation of all data, and avoids having duplicate elements in $\mathbb{G}$. As a consequence, $(m-2q)/m \simeq 90\,\%$ confidence regions are provided by LSCR.

10.6 Selection of the parameters of the LSCR approach

To apply the proposed LSCR technique, several parameters have to be chosen. The most difficult to choose is e, which determines the considered statistic of the prediction residuals. Considering $s = 1$ and $e = (1, 1)^T$ corresponds to the correlation function considered in Equation 9.1, which provides satisfying results in the considered application, as will be seen in what follows.

Other choices are possible, but two aspects have to be taken into account. The first is related to the characterization by SIVIA of NACRs. In the inclusion function $[\mathbf{f}_{\mathbf{e},m}]$ of $\mathbf{f}_{\mathbf{e},m}$ used by SIVIA, all occurrences of the parameters are considered as independent. The pessimism with which the range of $\mathbf{f}_{\mathbf{e},m}$ is evaluated depends on the number of occurrences of each parameter to be estimated in the formal expression of $\mathbf{f}_{\mathbf{e},m}$, see Kieffer and Walter [2014]. Thus, statistics with large values of $\sum_{i=0}^{s} e_i$ should be avoided to improve the efficiency of SIVIA. The second is related to the LSCR method itself: the correlation functions should lead to small and regularly-shaped confidence regions.

Some preliminary experiments have been conducted to evaluate the impact of $\mathbf{e}$. One has considered $s = 0$ and $s = 1$ as well as $q = 3$, and $m = 63$. To build $\mathbb{G}$ used in Dalai et al. [2007b], Campi and Weyer [2005], Gordon's procedure Gordon [1974] is used, but each $\mathbb{I}_j \in \mathbb{G}$ consists only of the $ny - s$ elements provided by the original procedure. The choice $m = 63$ allows an exploitation of all data, and avoids having duplicate elements in $\mathbb{G}$. As a consequence, $(m-2q)/m \simeq 90\%$ confidence regions are provided by LSCR.

Table 10.1 represents the NACR for the location θ_0 of an agent represented by a cross for different statistics Equation 10.6 used by LSCR. The correlation functions

are defined by the vector $\mathbf{e}$ considering the organization of the measurement vector introduced in Equation 10.3. The shape of the confidence regions is satisfying only when $e = (1, 1)^T$ or when $e = (3, 1)^T$. The sets in blue and red represent respectively the projections on the (θ_1, θ_2)-plane of the inner (blue boxes) and outer (blue and red boxes) approximations, respectively of 90% NACRs.

Table 10.2 represents similar NACR but considering the following alternative organization of the measurement vector

$$\mathbf{y}(j) = (\mathbf{y}(1, j), \ldots, \mathbf{y}(n_a, j))^T \tag{10.20}$$

$$\mathbf{y} = \left(\mathbf{y}^T(1), \ldots, \mathbf{y}^T(n)\right)^T . \tag{10.21}$$

In that case, the shapes of the confidence regions are satisfying for none of the vectors $\mathbf{e}$. To interpret the previous results, one may consider the asymptotic analysis of the behavior of the LSCR approach provided in [7]. A condition to get $\mathbb{P}_{\mathbf{e},q,m}$ shrinking to $\mathbf{p}^*$ when $n_y = n_a \times n$ tends to infinity is that for any $\mathbf{p} \neq \mathbf{p}^*$, all $s_{j,\mathbf{e}}(\mathbf{p})$ should have the same sign as n_y tends to infinity. In subsection 10.10.1 and subsection 10.10.2, one shows that this property is satisfied when considering $\mathbf{e} = (1, 1)^T$ or $\mathbf{e} = (3, 1)^T$ in Equation 10.6. Moreover, it is not possible to consider an organization of the measurement vector as described in Equation 10.20.

10.7 Evaluation of the LSCR approach

First, one evaluates NACR as defined by LSCR for $\mathbf{p} = \boldsymbol{\theta}_0$ only, using the plain SIVIA algorithm and compares its performance to a characterization via gridding, see Table 10.3. Both schemes provide sets of comparable shapes. When ε decreases, the set of boxes provided by SIVIA and the set of points obtained by gridding become similar. Nevertheless, with $\varepsilon = 0.2$ (resp. $\varepsilon = 0.05$), characterization via SIVIA is 10 (resp. 100) times faster than via gridding. Gridding is faster than SIVIA only for large values of ε, which confirms results already obtained in Kieffer and Walter [2014].

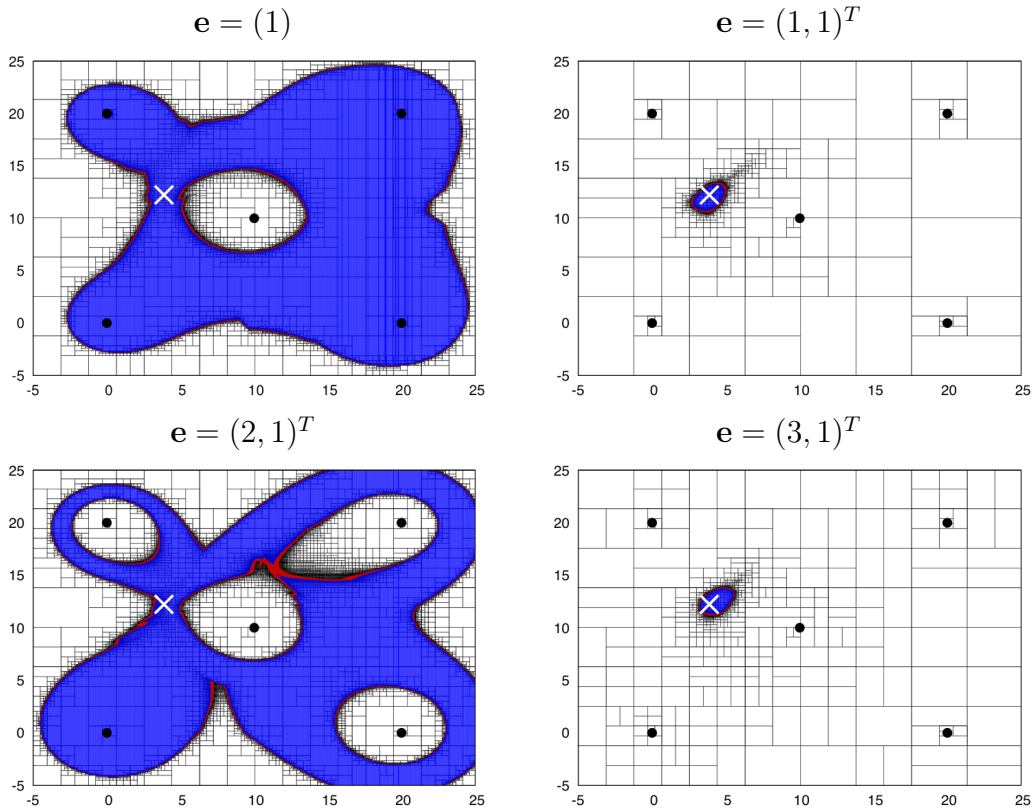


Table 10.1: NACR as defined by LSCR for the location of a given agent represented by a cross; the NACR are obtained for different statistics defined by $\mathbf{e}$, with an organization of the measurement vector as described by Equation 10.3.

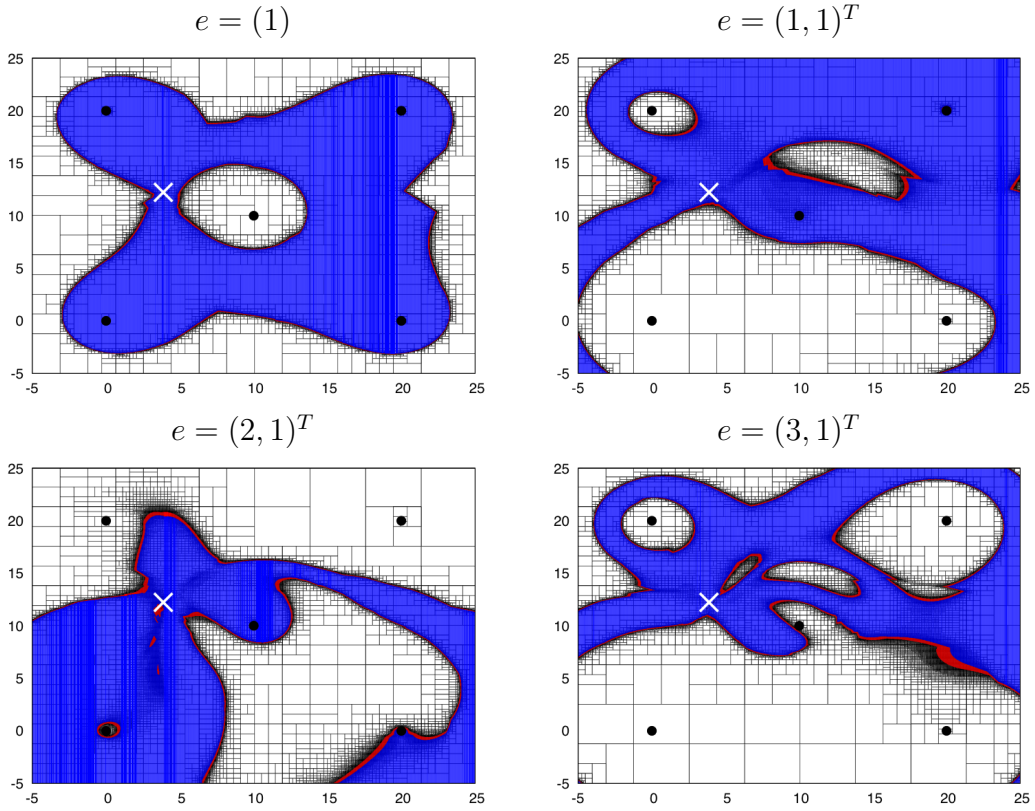


Table 10.2: NACR as defined by LSCR for the location of a given agent represented by a cross; the NACR are obtained for different statistics defined by e, considering a measurement vector obtained by stacking the data obtained by all anchor nodes at each instant.

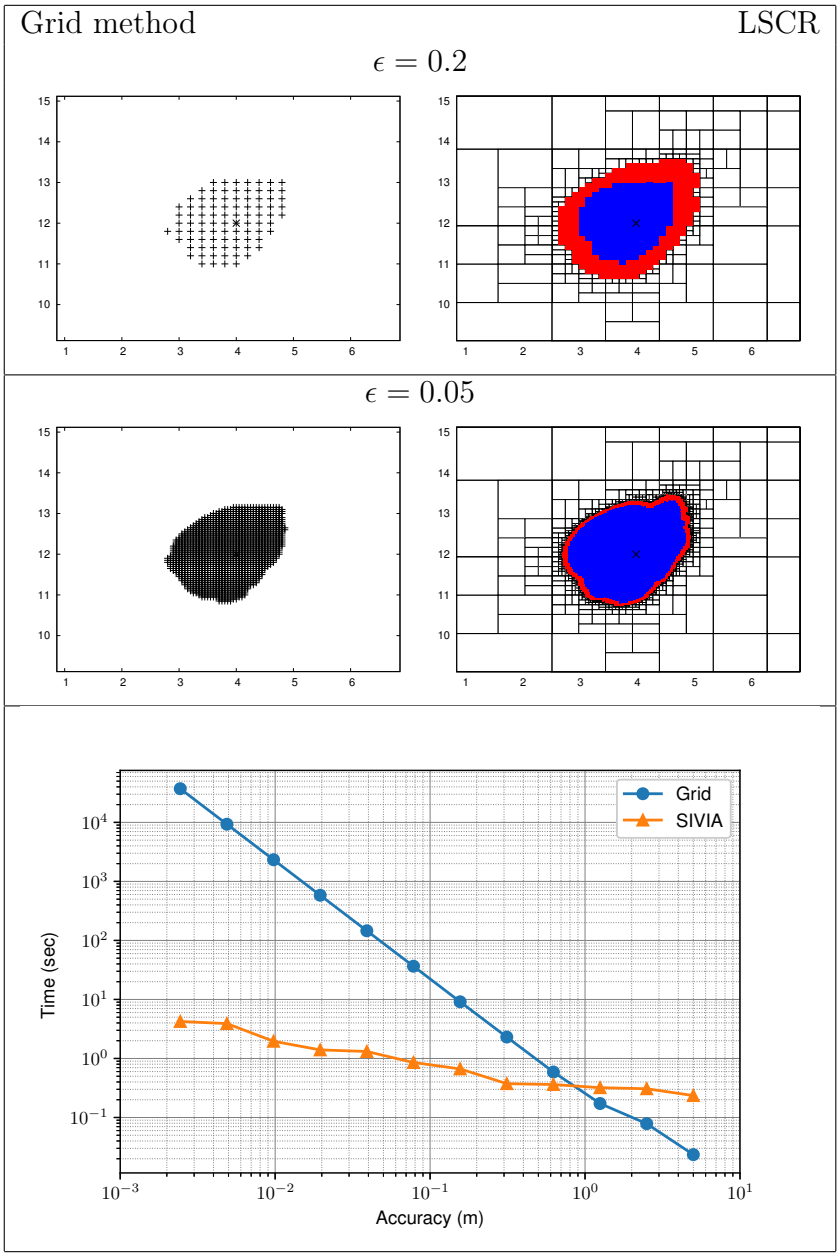


Table 10.3: NACR as defined by LSCR: comparison of the sets obtained by gridding and using SIVIA; computing time as a function of ε (left) and set of points or subpavings (right) for ε = 0.2 and ε = 0.05. Notice that this agent has five anchors. The number of the agent can affect the size of the confidence region too. With more anchors, one is able to obtain a smaller confidence region.

Now, one aims at comparing SIVIA algorithm without contractor (NC), and variant of SIVIA involving the forward-backward contractor (FB), the 3BCid contractor (3B), and the contractor based on the centred form (CF). In all cases, these contractors are used to build, for some box $[\mathbf{p}]$, the m subboxes of $[\mathbf{p}]$ such that $s^e_{i,r}([\mathbf{p}]) \geqslant 0$, $i = 1, ..., m$ and the m subboxes of $[\mathbf{p}]$ such that $s_{i,\mathbf{e}}([\mathbf{p}]) \leqslant 0$, $i = 1, ..., m$, see Equation 10.9 and Equation 10.10. The q-intersection contractor Kieffer and Walter [2014] is then applied on both subsets of m boxes.

Figure 10.1 shows the size of the confidence region for $\mathbf{p} = \boldsymbol{\theta}_0$ when using the plain SIVIA, SIVIA with FB, 3B, and CF contractors. For each technique, two lines are provided. The upper lines represent the size of the outer approximation and the lower lines represent the size of the inner approximation of the NACR. The outer and inner approximation converge when the parameter ε becomes smaller. For a given value of ε, the confidence region obtained with SIVIA and using contractors has a smaller size than without contractors. The 3B and CF contractors provide the best results.

Figure 10.2 shows the evolution of size of the outer and inner approximations of the NACR as a function of the time required to obtain it with the different variants of SIVIA without and with contractors. Clearly, NC provides the best compromise between accuracy and computing time.

To understand the relative inefficiency of contractors in the context of NACR characterization, Figure 10.3 shows the contraction obtained on each box processed by SIVIA using the CF contractor. The x-axis represents the original surface of each processed box and the y-axis represents the surface of the box after contraction. One sees that large boxes are left unchanged. The CF contractor is only able to contract relatively small boxes, which helps obtaining a more accurate description of the NACR, but with a significantly increased computational complexity. The computing time penalty to apply contractors is not compensated by the gain in terms of reduction of the size of the search space provided by contractors.

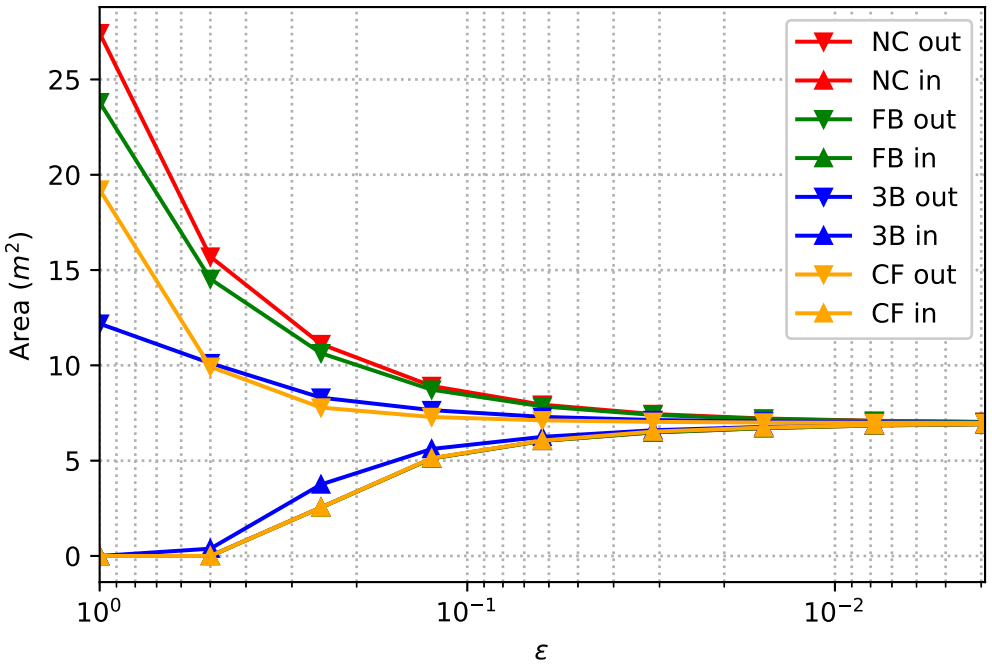


Figure 10.1: Size of the outer (upper lines) and inner (lower lines) approximation of the confidence region as defined by LSCR provided by different variants of SIVIA, without and with contractors for different values of the precision parameter ε

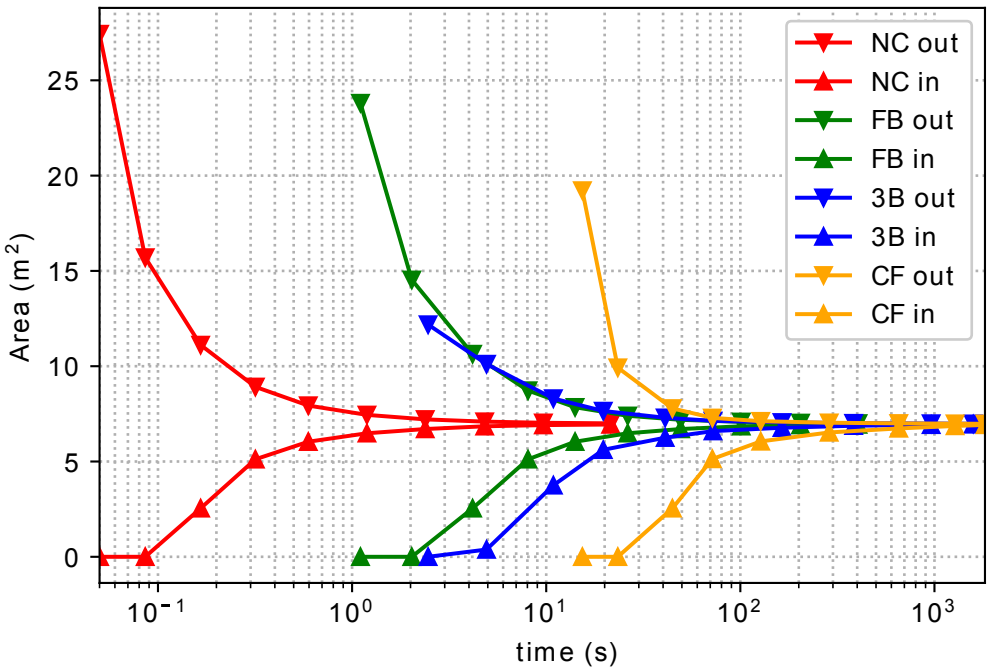


Figure 10.2: Size of the outer (upper lines) and inner (lower lines) approximation of the confidence region as defined by LSCR provided by different variants of SIVIA, without and with contractors as a function of the computing time (the curves are parametrized in ε)

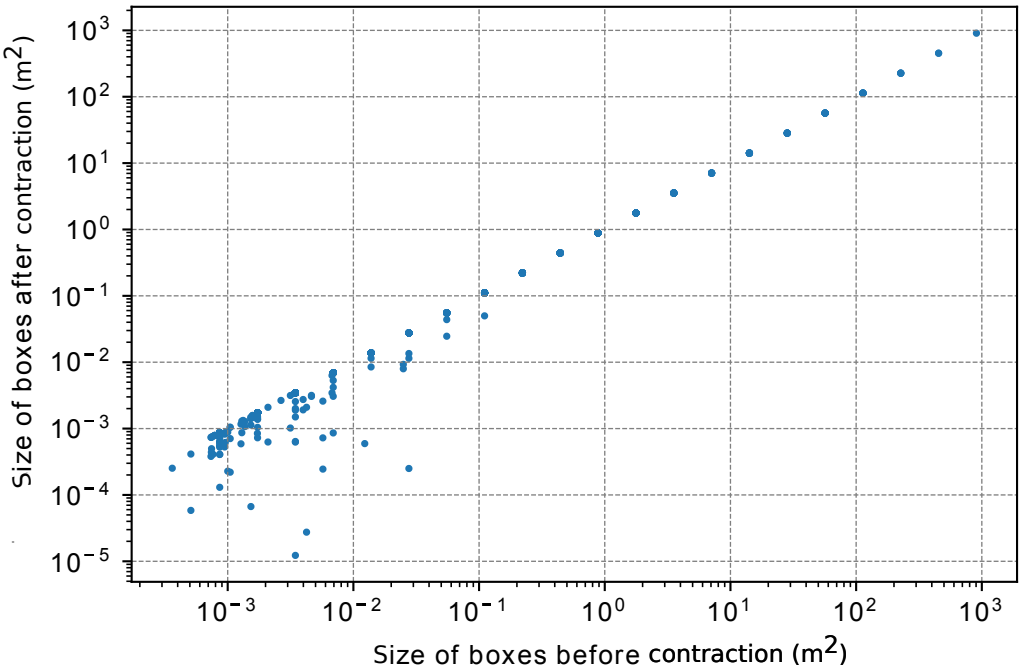


Figure 10.3: Evolution of the surface of boxes before and after processing by the CF contractor. If the contractor can not reduce the size of the box, then the size of the box remain the same after the contraction. One can see most of the boxes cannot be contracted. Only small portion of the boxes is contract, so their size become smaller.

10.8 Comparison with alternative methods

Table 10.4 summarizes an example of estimation results obtained with the approaches described in chapter 10 and section 9.3 considering Gaussian noise only. The columns correspond to different estimation problems. The rows correspond to the results provided, respectively by LSCR, the BE, and the RBE estimators as well as the estimate obtained using a MLE approach with confidence regions evaluated from the CRLB approach presented in subsection 9.3.3 on page 109.

The intervals for the measurement noise are taken as $[-3\sigma_0, 3\sigma_0]$ for the BE approach and as $[-2\sigma_0, 2\sigma_0]$ for the RBE approach. The plain SIVIA algorithm without contractors is used with the LSCR, BE, and RBE approaches with $\varepsilon = 0.05$. The sets in green and yellow represent respectively the projections on the (θ_1, θ_2)-plane of the inner (blue boxes) and outer (blue and red boxes) approximations, respectively, of 90 % NACRs (for LSCR), or of the set estimates (for the BE or RBE approaches). For some cases, yellow part is too thin to verify. A triangle close to a node location indicates that the set provided for the node is either empty or does not contain the actual node location.

The initialization for the nonlinear MLE is taken at random within the convex hull of the anchors. Optimization is performed using Matlab `lsqnonlin` non-linear least-squares optimization function, which is detailed in subsection 9.3.3. With some initialization, `lsqnonlin` provides estimates which projection on the (θ_1, θ_2)-plane are outside the $[-5, 25]^2$ box. In that case, a new search is performed with an other random initialization. This reinitialization is performed up to three times. The green ellipses represent the 90 % confidence region derived from the CRLB. The lines link the estimated positions $\hat{\theta}$ and true positions θ^* of the agents.

Table 10.5 is similar to Table 10.4 but with results obtained from measurements corrupted by GBG noise. The intervals for the measurement noise are again taken as $[-3\sigma_0, 3\sigma_0]$ for the BE approach and as $[-2\sigma_0, 2\sigma_0]$ for the RBE approach.

Table 10.6 provides the estimates of $\boldsymbol{\theta}_0$ obtained with the SDP approach assuming that both $\boldsymbol{\theta}_0$ and P_0 have to be estimated and the projection of the 90 %

confidence ellipsoid derived from the CRLB in the case of Gaussian noise only (left) and Gaussian-Bernoulli-Gaussian noise (right).

Table 10.4 shows that when the number of the unknown variables increases, the size of the sets provided by the LSCR, BE, and RBE approaches increases. This is due to the fact that an uncertainty in P_0 or γ_{P} translates directly in an uncertainty in terms of node localization. One also see that an uncertainty in P_0 or in γ_{P} have comparable impacts on the size and shape of the obtained sets. This is consistent with the identifiability issue mentioned in section 10.5. Sometimes, the sets provided by LSCR are empty or do not contain the actual node location. This is consistent with the definition of a 90% confidence region, which should contain the actual location only in 90 % of the cases.

Considering GBG noise also increases the size of the sets obtained: confidence regions in Table 10.5 tend to be larger then those in Table 10.4. This is not surprising, since data corrupted by GBG noise contain less information about the node location than data corrupted by log-normal noise.

Nonlinear MLE provides good results when only $\boldsymbol{\theta}_0$ has to be estimated. Nevertheless, when both $\boldsymbol{\theta}_0$ and P_0 or $\boldsymbol{\theta}_0$ and γ_P have to be estimated, the estimation errors increases significantly. SDP is more robust than nonlinear MLE when several parameters have to be estimated. In general, confidence regions derived from the CRLB are by far too optimistic: agents seldom belong to the 90 % confidence ellipsoid.

In Table 10.4, the sets evaluated using BE are in general larger than those obtained with RBE or than the confidence regions defined by LSCR. In the case of GBG noise, most of the sets provided by the BE approach is empty. This is not surprising, since the assumption on the noise bounds is violated in many cases. The size of the sets evaluated with the RBE approach is not much affected by the nature of the noise, showing the robustness of the approach to variations of the noise characteristics. Nevertheless, in some cases, in presence of GBG noise, the sets provided by the RBE approach do not contain the true value of the location of some nodes. This is due to the fact that not enough outliers where tolerated.

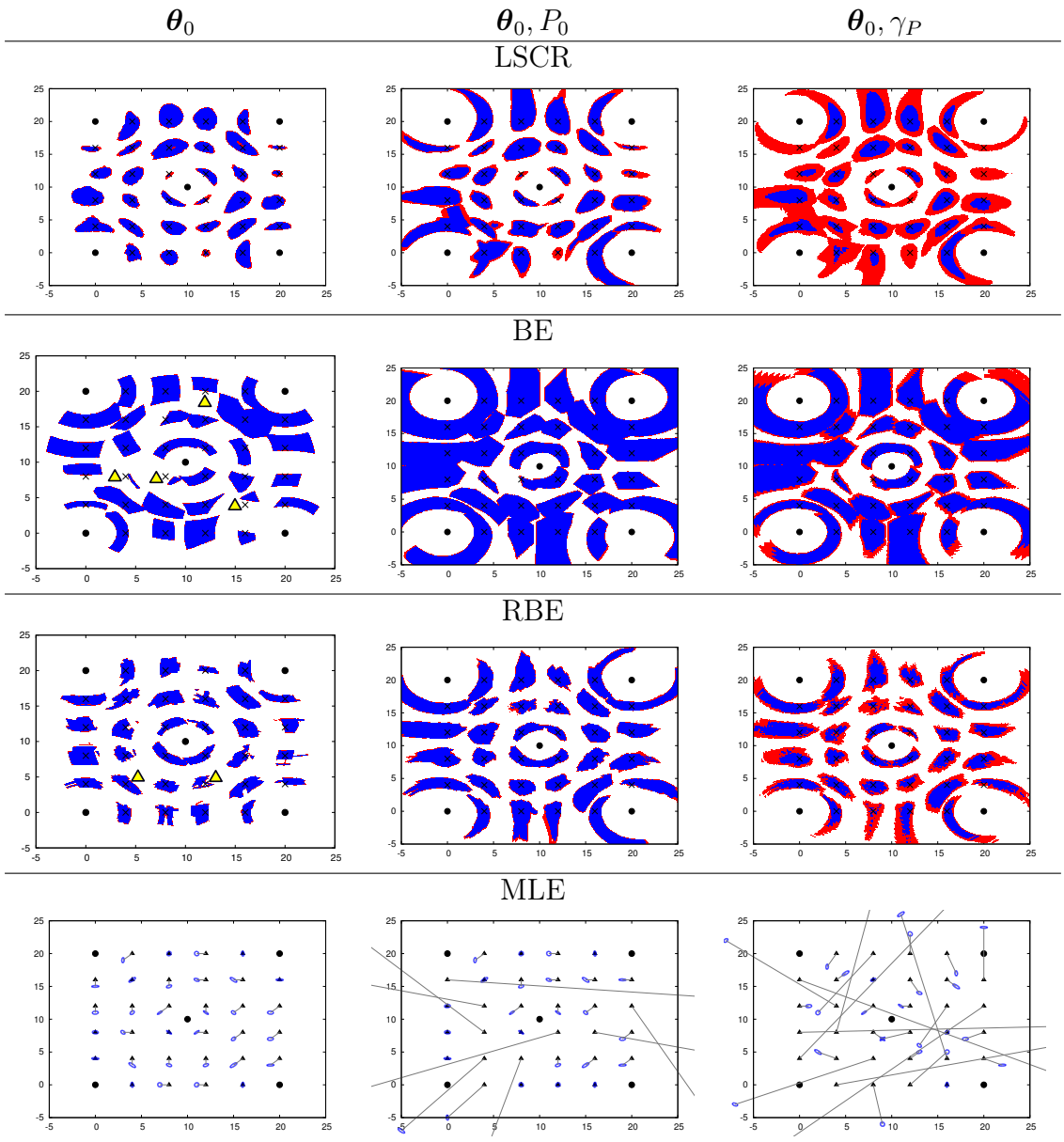


Table 10.4: Simulations considering measurements disturbed by Gaussian noise: Projections on the (θ_1, θ_2)-plane of inner and outer-approximations of set estimates obtained by SIVIA in the case NACR as defined by LSCR, and in the case of Bounded-Error (BE), and Robust Bounded-Error (RBE) estimation, as well as MLE with confidence regions derived from the CRLB; nodes for which the set estimate is empty or does not contain its actual location are marked by a triangle.

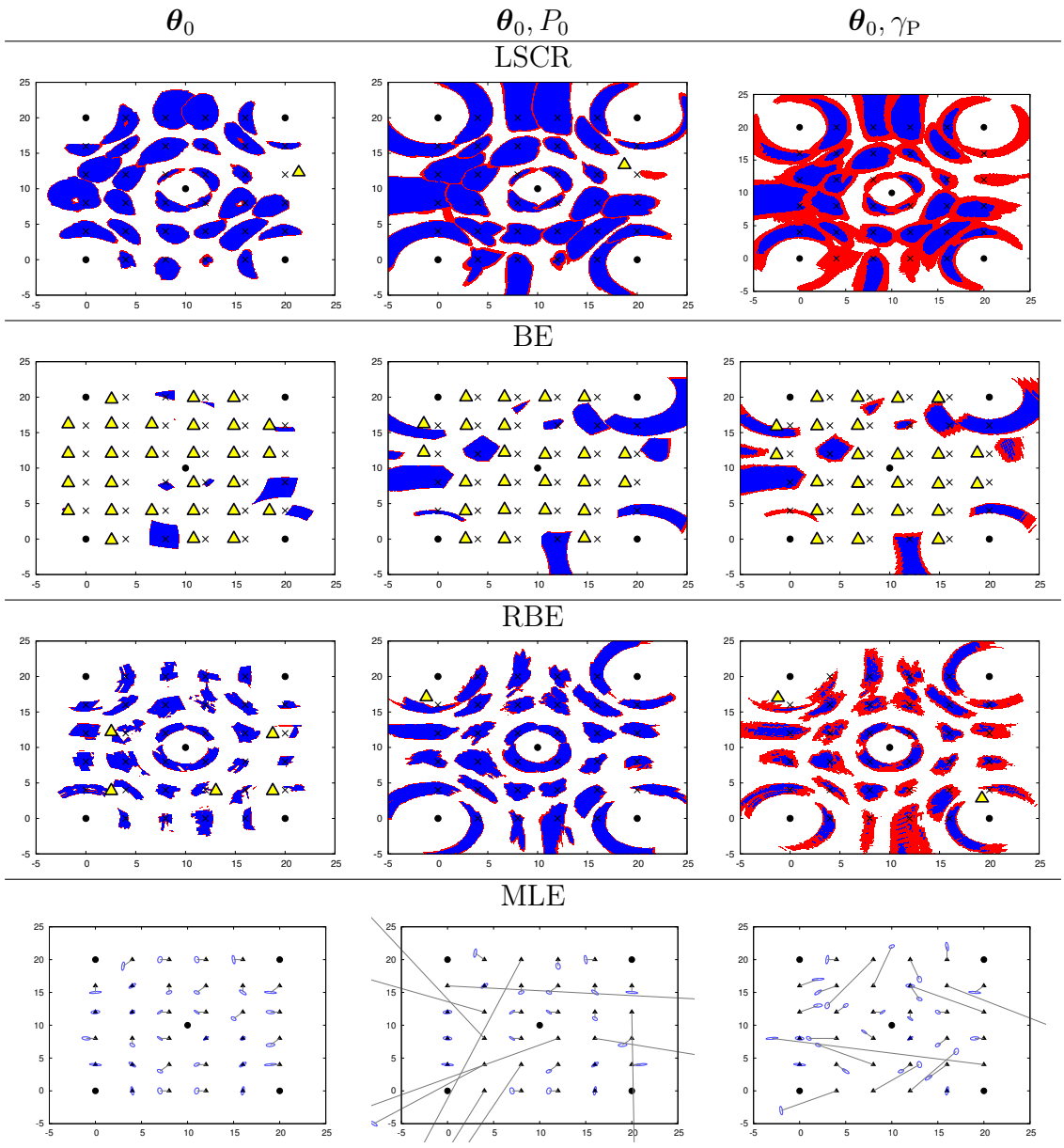


Table 10.5: Simulations considering measurements disturbed by GBG noise: Projections on the (θ_1, θ_2)-plane of inner and outer-approximations of set estimates obtained by SIVIA in the case NACR as defined by LSCR, and in the case of Bounded-Error (BE), and Robust Bounded-Error (RBE) estimation, as well as MLE with confidence regions derived from the CRLB; nodes for which the set estimate is empty or does not contain its actual location are marked by a triangle.

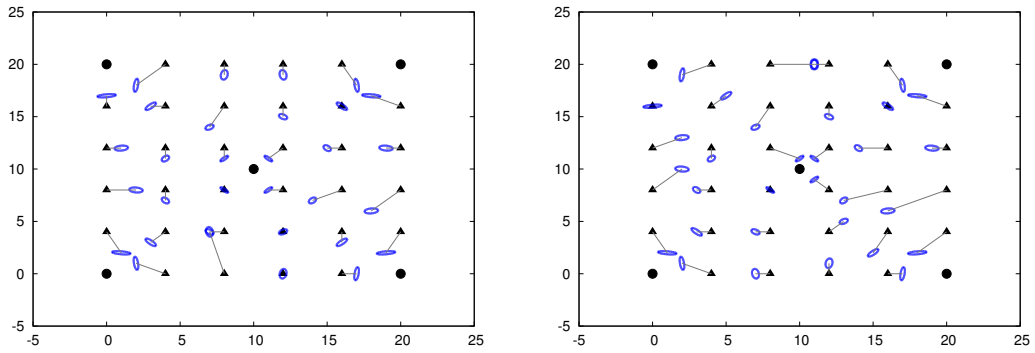


Table 10.6: Projection on the (θ_1, θ_2)-plane of estimates and confidence ellipsoids obtained by the SDP method in Gaussian noise (left) and GBG noise (right)

10.9 Effect of the noise variance

In this chapter, one studies the effect of the noise variance on the size of the confidence region and on the location estimation error. The same simulation setup as in section 10.5 is considered, but now, $N = 100$ agents to be localized are randomly placed in the square. Noise samples are realizations of iid zero-mean Gaussian random variables with standard deviation starting from $\sigma_0 = 0.5$ to $\sigma_0 = 4$ dBm for each simulation. Both $\boldsymbol{\theta}_{0,i}$ and $P_{0,i}$, $i = 1, \ldots, N$ have to be determined for each agent.

10.9.1 Localization error

To evaluate the localization error of the LSCR, BE, and RBE approaches, one considers the outer approximations $\overline{\mathbb{P}}_{r,q,m}$, $\overline{\mathbb{P}}_{\mathrm{BE}}$, and $\overline{\mathbb{P}}_{\mathrm{RBE}}$ of $\mathbb{P}_{\mathbf{e},q,m}$, $\mathbb{P}_{\mathrm{BE}}$, and $\mathbb{P}_{\mathrm{RBE}}$ and evaluate the center of mass of these sets defined, *e.g.*, for $\overline{\mathbb{P}}_{\mathrm{BE}}$ as

$$\widehat{\mathbf{p}}_{\mathrm{BE}} = \frac{1}{\mathrm{vol}\left(\overline{\mathbb{P}}_{\mathrm{BE}}\right)} \sum_{[\mathbf{p}] \in \overline{\mathbb{P}}_{\mathrm{BE}}} \mathrm{vol}\left([\mathbf{p}]\right) \mathrm{mid}\left([\mathbf{p}]\right), \tag{10.22}$$

where $[\mathbf{p}]$ is a box contained in the list of non-overlapping boxes forming $\overline{\mathbb{P}}_{\mathrm{BE}}$, $\mathrm{vol}\left([\mathbf{p}]\right)$ and $\mathrm{mid}\left([\mathbf{p}]\right)$ are the volume and midpoint of $[\mathbf{p}]$. The estimated location $\widehat{\boldsymbol{\theta}}_{0,\mathrm{BE}}$ is the deduced from $\widehat{\mathbf{p}}_{\mathrm{BE}}$. When an empty set is provided by the LSCR or the BE estimation techniques, no point estimate is evaluated.

Figure 10.4 shows the average value of the norm

$$\delta = \frac{1}{N}\sum_{i=1}^{N}\left|\widehat{\boldsymbol{\theta}}_{0,i} - \boldsymbol{\theta}_{0,i}\right| \tag{10.23}$$

of the difference between the estimated location $\widehat{\boldsymbol{\theta}}_{0,i}$ and the actual location $\boldsymbol{\theta}_{0,i}$ as a function of the measurement noise standard deviation. For the LSCR and BE techniques, the average is only evaluated when the obtained sets are not empty.

For all estimation techniques, δ decreases with σ. The LSCR and RBE methods provide the best results. The performance of SDP is due to the relaxation of the constraints. MLE fails to provide reasonable estimates for some of the sensors due to local minima of the cost function.

10.9.2 Characteristics of the confidence region

Figure 10.5 shows the proportion of agents actually located in the set estimates or in the 90 % confidence region provided by the LSCR or derived from the CRLB. For BE, RBE, and LSCR, lower and upper bounds are provided, corresponding to the inclusion in the inner and outer approximation of the sets $\mathbb{P}_{\mathrm{BE}}$, $\mathbb{P}_{\mathrm{RBE}}$, and $\mathbb{P}_{\mathbf{e},q,m}$, respectively. Figure 10.5 shows that the confidence regions associated to the estimates provided by the MLE and SDP approaches are often too optimistic. LSCR, BE and RBE can produces confidence regions or set estimates which are reliable.

Figure 10.6 shows the average surface of the projection on the (θ_1, θ_2)-plane of the NACR, of the BE and RBE set estimates, and of the confidence region derived from the CRLB. The size of the confidence regions provided by the CRLB (MLE and SDP approaches) are the smallest. Nevertheless, the regions are not realistic as shown in Figure 10.6. The LSCR and RBE approaches provide sets of similar size,

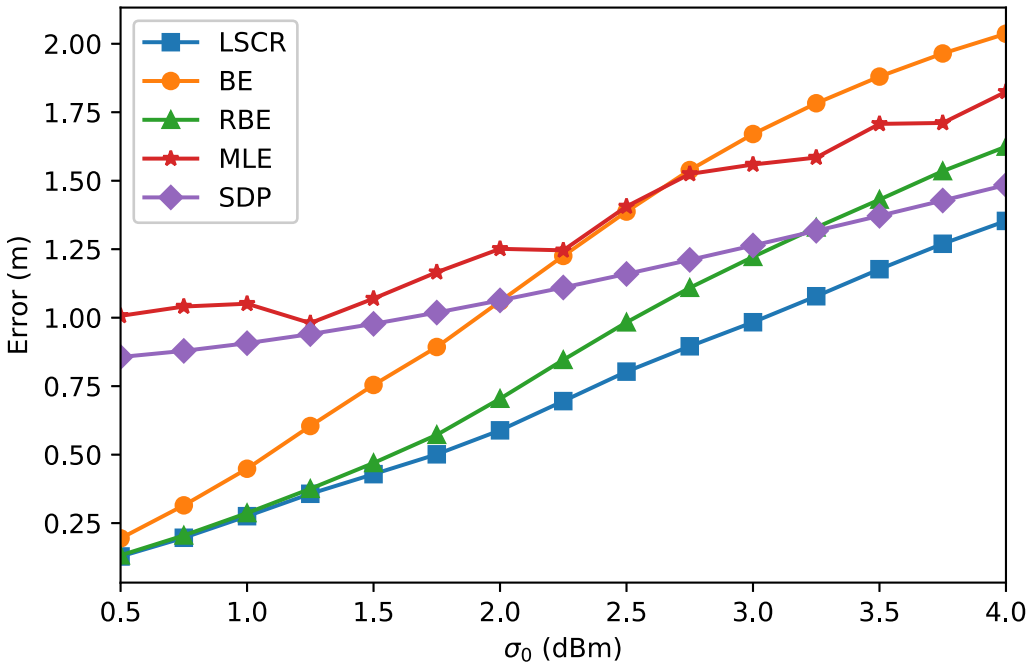


Figure 10.4: Evolution of the localization error as a function of σ_0

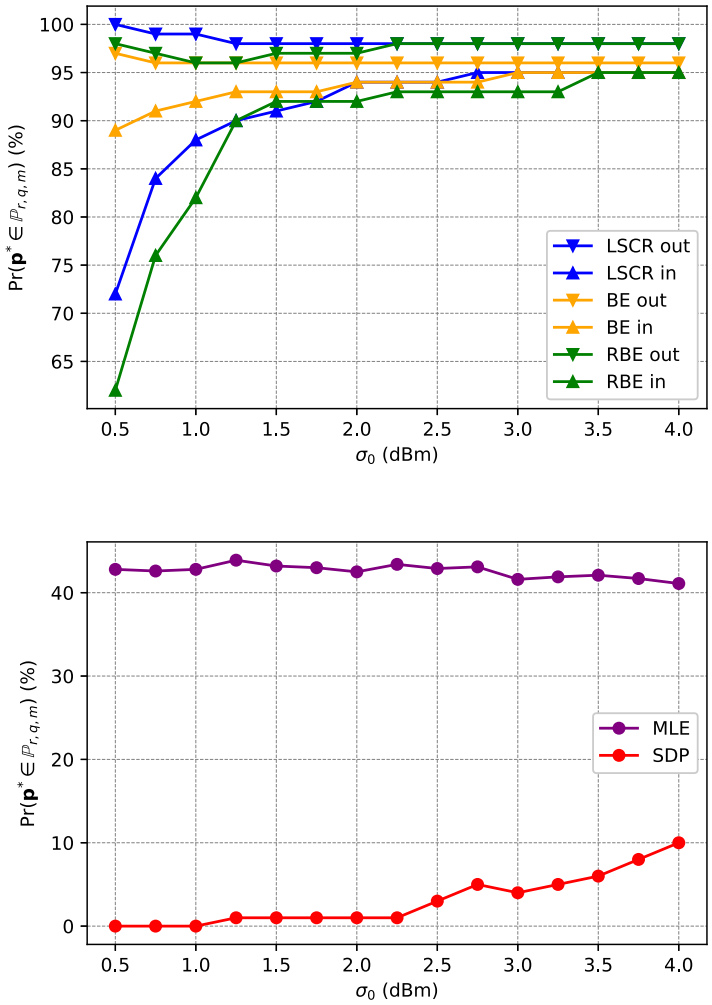


Figure 10.5: Proportions of realizations for which the true value of the agent location is contained in the 90 % NACR (left), in the BE set estimates (up), or in the 90 % confidence regions derived from the CRLB (down)

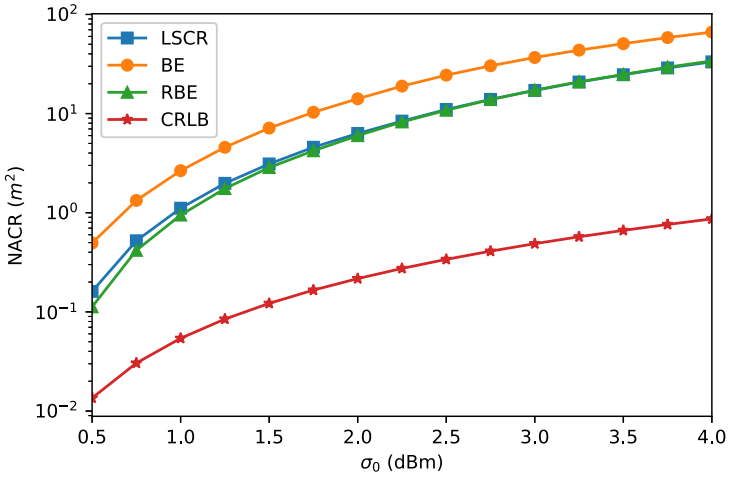


Figure 10.6: Evolution of the average surface of the projection on the (θ_1, θ_2)-plane of the NACRs, the set estimates, or the confidence region derived from the CRLB

while the size of the set provided by the BE approach is larger due to the larger noise bounds considered.

10.10 Non-Asymptotic Confidence Region Analysis

In this section, one analyzes the behavior of $s_{j,\mathbf{e}}(\mathbf{p}), j = 1, ..., m$ when n_y tends to infinity, assuming that n_a anchor nodes provide each the same number n of measurements tending to infinity. The parameters m and q are fixed. Our aim is to show that for properly chosen vectors $\mathbf{e}$, when $\mathbf{p} = \mathbf{p}^*$ the functions $s_{j,\mathbf{e}}(\mathbf{p}), j = 1, ..., m$ tend to have all the same sign as n^y tends to infinity. Moreover, to ensure that the NACR shrinks toward $\mathbf{p}^*$, one has to determine in which cases, $\mathbf{p} = \mathbf{p}^*$ is the unique solution of the equations $s_{j,\mathbf{e}}(\mathbf{p}) = 0, j = 1, ..., m$. To perform this analysis, one assumes that the w_j are independent and identically distributed.

10.10.1 First organization of the vector of measurements

One considers first an organization of the measurement vector as described in Equation 10.3 on page 122 and $\mathbf{e} = (1,1)^T$. Moreover, the groups $\mathbb{G}^{n_y-s}$ are assumed to be constructed as described in section 10.3 on page 124. For a given ℓ, one considers the elements $\mathbb{I}^{n_y-s}$ of $\mathbb{G}^{n_y-s}$ with increasing n_y and focus on values of $n_y - s = kn_a(m-1)$, with $k = 1, ...$ In that case, for a given j, one gets

$$s_{j,\mathbf{e}}(\mathbf{p}) = \frac{1}{|\mathbb{I}^{kn_a(m-1)}|} \sum_{\ell \in \mathbb{I}^{kn_a(m-1)}} c_{\ell,\mathbf{e}}(\mathbf{p}). \tag{10.24}$$

Considering $\mathbf{e} = (1,1)^T$, one get

$$c_{t,(1,1)}(\mathbf{p}) = w_t(\mathbf{p}) w_{t+1}(\mathbf{p}), \tag{10.25}$$

which, with the measurement vector (6), may take two forms, depending on whether $w_t(\mathbf{p})$ and $w_{t+1}(\mathbf{p})$ involve measurements from the same anchor node (most of the cases), or from different anchor nodes, which is the case when $t \in \{n, 2n, ..., (n_a - 1)n\}$. In the first case

$$\begin{aligned} c_{t,(1,1)}(\mathbf{p}) = & \left(\left(P_0^* - 10\gamma_{\mathrm{P}}^* \log_{10} \frac{\|\boldsymbol{\theta}_0^* - \boldsymbol{\theta}_{i_t}\|}{d_0} + w_t \right) - \left(P_0^* - 10\gamma_{\mathrm{P}}^* \log_{10} \frac{\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}_{i_t}\|}{d_0} + w_t \right) \right) \\ & \left(\left(P_0^* - 10\gamma_{\mathrm{P}}^* \log_{10} \frac{\|\boldsymbol{\theta}_0^* - \boldsymbol{\theta}_{i_t}\|}{d_0} + w_{t+1} \right) - \left(P_0^* - 10\gamma_{\mathrm{P}}^* \log_{10} \frac{\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}_{i_t}\|}{d_0} + w_t \right) \right) \\ = & \left(10\gamma_{\mathrm{P}}^* \log_{10} \frac{\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}_{i_t}\|}{\|\boldsymbol{\theta}_0^* - \boldsymbol{\theta}_{i_t}\|} + w_t \right) \left(10\gamma_{\mathrm{P}}^* \log_{10} \frac{\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}_{i_t}\|}{\|\boldsymbol{\theta}_0^* - \boldsymbol{\theta}_{i_t}\|} + w_{t+1} \right), \end{aligned} \tag{10.26}$$

where i_t is the index of the anchor node which has produced measurement t. Similarly, in the second case,

$$c_{t,(1,1)}(\mathbf{p}) = \left(10\gamma_{\mathrm{P}}^* \log_{10} \frac{\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}_{i_t}\|}{\|\boldsymbol{\theta}_0^* - \boldsymbol{\theta}_{i_t}\|} + w_t \right) \left(10\gamma_{\mathrm{P}}^* \log_{10} \frac{\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}_{i_t+1}\|}{\|\boldsymbol{\theta}_0^* - \boldsymbol{\theta}_{i_t+1}\|} + w_{t+1} \right), \tag{10.27}$$

One rewrite Equation 10.24 as

$$s_{j,\mathbf{e}}(\mathbf{p}) = s^{(1)}_{j,\mathbf{e}}(\mathbf{p}) + s^{(2)}_{j,\mathbf{e}}(\mathbf{p}) \tag{10.28}$$

with

$$s^{(1)}_{j,\mathbf{e}}(\mathbf{p}) = \frac{1}{\left|\mathbb{I}^{kn_a(m-1)}\right|} \sum_{i=1}^{n_a} \sum_{\ell \in \mathbb{I}_j^{kn_a(m-1)} \cap [(i-1)n+1 in[} c_{\ell,\mathbf{e}}(\mathbf{p}) \tag{10.29}$$

and

$$\begin{aligned} s^{(2)}_{j,\mathbf{e}} &= \frac{1}{\left|\mathbb{I}^{kn_a(m-1)}\right|} \sum_{\ell \in \mathbb{I}^{kn_a(m-1)} \cap \{n,\ldots,(n_a-1)n\}} c_{\ell,\mathbf{e}}(\mathbf{p}) \\ &= \frac{1}{k\left|\mathbb{I}^{n_a(m-1)}\right|} \sum_{\ell \in \mathbb{I}^{kn_a(m-1)} \cap \{n,\ldots,(n_a-1)n\}} c_{\ell,\mathbf{e}}(\mathbf{p}). \end{aligned}$$

In $s^{(1)}_{j,\mathbf{e}}(\mathbf{p})$, the second sum with indices $\ell \in \mathbb{I}_j^{kn_a(m-1)} \bigcap [(i-1)n+1, in[$ considers only the $c_{\ell,\mathbf{e}}(\mathbf{p})$ involving measurements from the same anchor node i. In $s^{(2)}_{j,\mathbf{e}}(\mathbf{p})$, the sum with indices $\ell \in \mathbb{I}_j^{kn_a(m-1)} \bigcap \{n, ..., (n_a-1)n\}$ considers only the $c_{\ell,\mathbf{e}}(\mathbf{p})$ involving measurements from different anchor nodes.

Since $\mathbb{I}_j^{kn_a(m-1)}$ contains sequences of kn_a consecutive indices and $\mathbb{I}_j^{n_a(m-1)}$ contains sequences of n_a consecutive indices, using $\mathbb{I}_j^{n_a(m-1)}$ one may deduce the indices present in $\mathbb{I}_j^{kn_a(m-1)}$. For example, if $\ell = 1 \in \mathbb{I}_j^{n_a(m-1)}$, then $\{1, \ldots k\} \subset \mathbb{I}_j^{kn_a(m-1)}$. More generally, if $\ell \in \mathbb{I}_j^{n_a(m-1)}$, then $\{(\ell-1)k+1, \ldots \ell k\} \subset \mathbb{I}_j^{kn_a(m-1)}$. Conse-

quently, Equation 10.29 on the preceding page may be rewritten as

$$
\begin{aligned}
s_{j,\mathbf{e}}^{(1)}(\mathbf{p}) &= \frac{1}{\left|\mathbb{I}^{kn_a(m-1)}\right|} \sum_{i=1}^{n_a} \sum_{\ell \in \mathbb{I}_j^{kn_a(m-1)} \cap [(i-1)n+1 in[} c_{\ell,\mathbf{e}}(\mathbf{p}) \\
&= \frac{1}{\left|\mathbb{I}^{kn_a(m-1)}\right|} \sum_{i=1}^{n_a} \sum_{\kappa=1}^{k} \sum_{\ell \in \mathbb{I}_j^{kn_a(m-1)}, (i-1)n+1 \leq (\ell-1)k+\kappa < in} c_{(\ell-1)k+\kappa,\mathbf{e}}(\mathbf{p}) \\
&= \frac{1}{k\left|\mathbb{I}^{kn_a(m-1)}\right|} \sum_{i=1}^{n_a} \sum_{\ell \in \mathbb{I}_j^{kn_a(m-1)}} \sum_{\kappa=1, (i-1)n+1 \leq (\ell-1)k+\kappa < in} c_{(\ell-1)k+\kappa,\mathbf{e}}(\mathbf{p}) \\
&= \frac{1}{\left|\mathbb{I}^{kn_a(m-1)}\right|} \sum_{i=1}^{n_a} \frac{1}{k} \sum_{\ell \in \mathbb{I}_j^{kn_a(m-1)}} \sum_{\kappa=1, (i-1)n+1 \leq (\ell-1)k+\kappa < in} c_{(\ell-1)k+\kappa,\mathbf{e}}(\mathbf{p}).
\end{aligned}
$$

Then, using similar arguments as those considered in [Dalai et al. [2007a] Appendix A.2], for all $\mathbf{p}$ such that the associated $\boldsymbol{\theta}_0 \neq \boldsymbol{\theta}_i$, $i = 1, ..., n_a$, in the case $\mathbf{e} = (1, 1)^T$, one has

$$
\lim_{k\to\infty} \frac{1}{k} \sum_{\kappa=1, (i-1)n+1 \leq (\ell-1)k+\kappa < in} c_{(\ell-1)k+\kappa,(1,1)}(\mathbf{p}) = \left(10\gamma_{\mathrm{P}}^* \log_{10} \frac{\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}_i\|}{\|\boldsymbol{\theta}_0^* - \boldsymbol{\theta}_i\|} + w_t \right)^2, \tag{10.30}
$$

which is the expected value of $c_{t,(1,1)}(\mathbf{p})$ for some t such that $i_t = i$.

Moreover, since $s_{j,\mathbf{e}}^{(2)}(\mathbf{p})$ consists only in n_a terms, for all $\mathbf{p}$ such that the associated $\boldsymbol{\theta}_0 \neq \boldsymbol{\theta}_i$, $i = 1, ..., n_a$, one has

$$
\lim_{k\to\infty} \frac{1}{k\left|\mathbb{I}^{n_a(m-1)}\right|} \sum_{i=1}^{n_a} \sum_{\ell \in \mathbb{I}_j^{kn_a(m-1)} \cap \{n,...,(n_a-1)n\}} c_{\ell,\mathbf{e}}(\mathbf{p}) = 0. \tag{10.31}
$$

Let $\alpha_{\ell,i} > 0$ be the proportion of indices in $\mathbb{I}^{n_y - s}$ corresponding to $c_{t,(1,1)}(\mathbf{p})$ involving measurements of anchor node i only. When $n_y - s = kn_a(m-1)$, these proportions are independent of k. As a consequence, for all $\mathbf{p}$ such that the associated

$\boldsymbol{\theta}_0 \neq \boldsymbol{\theta}_i$, $i = 1, ..., n_a$ one has

$$s^{\infty}_{j,(1,1)}(\mathbf{p}) = \lim_{k\to\infty} s_{j,(1,1)}(\mathbf{p}) = \sum_{i=1}^{n_a} \alpha_{\ell,i} \left(10\gamma^*_{\mathrm{P}} \log_{10} \frac{\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}_i\|}{\|\boldsymbol{\theta}^*_0 - \boldsymbol{\theta}_i\|} \right)^2, \tag{10.32}$$

which vanishes for all $\boldsymbol{\theta}_0$ such that

$$\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}_i\| = \|\boldsymbol{\theta}^*_0 - \boldsymbol{\theta}_i\|, i \in \{1, \ldots, n_a | \alpha_{\ell,i} > 0\}. \tag{10.33}$$

Provided that for each ℓ there are at least three non-aligned anchor nodes such that $\alpha_{\ell,i} > 0$, $\boldsymbol{\theta}_0 = \boldsymbol{\theta}_i$ is the unique solution of the system of Equation 10.33. Any $\boldsymbol{\theta}_0 \neq \boldsymbol{\theta}_i$ leads to strictly positive $s^{\infty}_{j,(1,1)}(\mathbf{p})$.

In the case $\mathbf{e} = (3, 1)^T$, using similar derivations, one obtains

$$s^{\infty}_{j,(3,1)}(\mathbf{p}) = \sum_{i=1}^{n_a} \alpha_{\ell,i} \left(10\gamma^*_{\mathrm{P}} \log_{10} \frac{\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}_i\|}{\|\boldsymbol{\theta}^*_0 - \boldsymbol{\theta}_i\|} \right)^4,$$

and again, $\boldsymbol{\theta}_0 = \boldsymbol{\theta}_i$ is the unique solution of the system of Equation 10.33.

In the case $\mathbf{e} = (2, 1)^T$, one obtains now

$$s^{\infty}_{j,(3,1)}(\mathbf{p}) = \sum_{i=1}^{n_a} \alpha_{\ell,i} \left(10\gamma^*_{\mathrm{P}} \log_{10} \frac{\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}_i\|}{\|\boldsymbol{\theta}^*_0 - \boldsymbol{\theta}_i\|} \right)^3.$$

where each term in the function can be positive or negative causing multiple solution points. For this choice of $\mathbf{e}$, the NACR will not shrink to $\mathbf{p}^*$ when n_y tends to infinity.

When both $\boldsymbol{\theta}_0$ and P_0 have to be estimated, in the case $\mathbf{e} = (1, 1)^T$, one obtains

$$
\begin{aligned}
c_{t,(1,1)}(\mathbf{p}) = & \left(\left(P_0^* - 10\gamma_{\mathrm{P}}^* \log_{10} \frac{\|\boldsymbol{\theta}_0^* - \boldsymbol{\theta}_{i_t}\|}{d_0} + w_t \right) - \left(P_0 - 10\gamma_{\mathrm{P}}^* \log_{10} \frac{\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}_{i_t}\|}{d_0} + w_t \right) \right) \\
& \left(\left(P_0^* - 10\gamma_{\mathrm{P}}^* \log_{10} \frac{\|\boldsymbol{\theta}_0^* - \boldsymbol{\theta}_{i_t}\|}{d_0} + w_{t+1} \right) - \left(P_0 - 10\gamma_{\mathrm{P}}^* \log_{10} \frac{\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}_{i_t}\|}{d_0} + w_t \right) \right) \\
= & \left(P_0^* - P_0 + 10\gamma_{\mathrm{P}}^* \log_{10} \frac{\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}_{i_t}\|}{\|\boldsymbol{\theta}_0^* - \boldsymbol{\theta}_{i_t}\|} + w_t \right) \left(P_0^* - P_0 + 10\gamma_{\mathrm{P}}^* \log_{10} \frac{\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}_{i_t}\|}{\|\boldsymbol{\theta}_0^* - \boldsymbol{\theta}_{i_t}\|} + w_{t+1} \right).
\end{aligned} \tag{10.34}
$$

when measurements from the same sensors are combined. Then one gets

$$
s_{j,(1,1)}^{\infty}(\mathbf{p}) = \sum_{i=1}^{n_a} \alpha_{\ell,i} \left(P_0^* - P_0 + 10\gamma_{\mathrm{P}}^* \log_{10} \frac{\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}_i\|}{\|\boldsymbol{\theta}_0^* - \boldsymbol{\theta}_i\|} \right)^2, \tag{10.35}
$$

which vanishes for all $(P_0, \boldsymbol{\theta}_0)$ such that

$$
P_0^* - P_0 + 10\gamma_{\mathrm{P}}^* \log_{10} \frac{\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}_i\|}{\|\boldsymbol{\theta}_0^* - \boldsymbol{\theta}_i\|} = 0, i = 1, \ldots, n_a. \tag{10.36}
$$

This condition may be rewritten as

$$
\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}_i\| = \|\boldsymbol{\theta}_0^* - \boldsymbol{\theta}_i\| \times 10^{\frac{P_0 - P_0^*}{10\gamma_{\mathrm{P}}^*}}, i = 1, \ldots, n_a. \tag{10.37}
$$

If P_0 is known to belong to some *a priori* search interval $[\underline{P}_0, \overline{P}_0]$, then $\boldsymbol{\theta}_0$ has to belong to the intersection of annulus centered in $\boldsymbol{\theta}_i$ with radius between $\|\boldsymbol{\theta}_0^* - \boldsymbol{\theta}_i\| \times 10^{\frac{\underline{P}_0 - P_0^*}{10\gamma_{\mathrm{P}}^*}}$ and $\|\boldsymbol{\theta}_0^* - \boldsymbol{\theta}_i\| \times 10^{\frac{\overline{P}_0 - P_0^*}{10\gamma_{\mathrm{P}}^*}}$. The intersection of these annuli defines the NACR for $\boldsymbol{\theta}_0$, and may also lead to a reduced confidence region for P_0. Nevertheless, in this case, the confidence region does not shrink to a single point in the search space. Similar conclusions can be obtained when both $\boldsymbol{\theta}_0$ and γ_{P} or $\boldsymbol{\theta}_0$, P_0, and γ_{P} have to be estimated.

10.10.2 Alternative organization of the vector of measurements

Consider now an organization of the measurement vector as described in Equation 10.20 and $e = (1,1)^T$. The expression of $c_{t,\mathbf{e}}(\mathbf{p})$ is still given by Equation 10.25, but now, all $c_{t,\mathbf{e}}(\mathbf{p})$ will involve measurements from two different anchor nodes. After some derivations, one obtains now:

$$s^{\infty}_{j,(1,1)}(\mathbf{p}) = \sum_{i=1}^{n_a-1} \beta_{\ell,i} \left(10\gamma^*_{\mathrm{P}} \log_{10} \frac{\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}_i\|}{\|\boldsymbol{\theta}^*_0 - \boldsymbol{\theta}_i\|}\right) \left(10\gamma^*_{\mathrm{P}} \log_{10} \frac{\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}_{i+1}\|}{\|\boldsymbol{\theta}^*_0 - \boldsymbol{\theta}_{i+1}\|}\right), \quad (10.38)$$

$$+ \beta_{\ell,n_a} \left(10\gamma^*_{\mathrm{P}} \log_{10} \frac{\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}_{n_a}\|}{\|\boldsymbol{\theta}^*_0 - \boldsymbol{\theta}_{n_a}\|}\right) \left(10\gamma^*_{\mathrm{P}} \log_{10} \frac{\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}_1\|}{\|\boldsymbol{\theta}^*_0 - \boldsymbol{\theta}_1\|}\right), \quad (10.39)$$

where $\beta_{\ell,i}$ is the proportion in $s^{\infty}_{j,(1,1)}(\mathbf{p})$ of terms $c_{t,\mathbf{e}}(\mathbf{p})$ involving a measurement provided by the i-th and the $i+1$ mod n_a anchor nodes. In that case, $s^{\infty}_{j,(1,1)}(\mathbf{p})$ is positive for values of $\boldsymbol{\theta}_0$ far away from the anchor nodes, i.e., such that $\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}_i\| > \|\boldsymbol{\theta}^*_0 - \boldsymbol{\theta}_i\|$ for all $i = 1, \ldots, n_a$. When $\boldsymbol{\theta}_0$ is close to $\boldsymbol{\theta}^*_0$ the sign of $s^{\infty}_{j,(1,1)}(\mathbf{p})$ is difficult to predict and unlikely to be the same for all $s^{\infty}_{j,(1,1)}(\mathbf{p}), j = 1, \ldots m$.

10.11 Conclusion

In this chapter, the localization algorithm using LSCR is discussed. With a user defined probability, the non-asymptotic confidence region NACR is defined. Originally, NACR is characterized using gridding method, where the points in the solution space are generated and verify whether it is in the NACR one by one. With the help of Interval analysis, the characterization is much faster when the required precision is high. Moreover, by using SIVIA, the outer and inner approximation of NACR is characterized, so the result is guaranteed.

This chapter also discusses the implementation and the result of LSCR localization. Considering the system parameters of LSCR, only a subset of parameters

are able to generate quality results. We also show the runtime of both SIVIA and gridding under different precision. SIVIA is preferred than gridding if high precision and dimension of the solution space are considered. We also compare LSCR with some state-of-the-art approaches to show the robustness of LSCR. Finally, the error of the center of the confidence region is compared, LSCR estimates the locations of the sensors with higher accuracy than the alternative approaches. Different kinds of LSCR arrangement are analyzed and show that only a subset of parameters $\mathbf{e}$ lead to a robust and precise NACR.

Chapter 11

Conclusions

The second part of the thesis showed that NACR may be obtained using the LSCR approach applied to the problem of source localization from RSS measurements. LSCR provides confidence locations of the sensors under mild assumptions on the noise distribution. The noise is assumed to be symmetrically distributed around zero. Inner and outer-approximations of the NACR are provided using tools from interval analysis. Contrary to confidence regions provided by CRLB associated to MLE or SDP techniques, NACR provided by LSCR contains the true value of the parameter vector to be estimated with the prescribed confidence level. Furthermore, LSCR provides the best localization result: its average error is smaller than other compared methods (BE, RBE, MLE, SDP). This work is published in Han et al. [2018a].

Chapter 12

Conclusion for this thesis

This thesis focus on two important problems related to WSN: localization and time synchronization. For clock synchronization, robustness of the algorithm is essential. In WSN, mobile and low-cost sensors cause communication errors. Considering the dynamicity, a distributed algorithm is proposed, which extends the previous work from Függer et al. [2015b] and reduce the clock skew. In the new pulse-coupled synchronization, time-wheel algorithm is introduced for sensors to exchange the round number and average the pulses of the same round. Moreover drift compensation is applied to reduce the clock skew. The convergence analysis shows that the drift compensation reduces the clock skew and improves the convergence rate. When considering a highly dynamic WSN, the algorithm converges faster than in previous works. The proposed algorithm is successfully applied to a vehicular network.

For the second part of the thesis, we estimated the location of each sensor using LSCR, which produces a non-asymptotic confidence region. The benefit of applying LSCR to localization problem is that the assumptions on the noise of RSS signal is mild, so the confidence region can be estimated accurately. We analyze different correlation functions of LSCR and found that only some correlation function can produce a satisfying confidence region. We also implement alternative localization algorithms, such as maximum likelihood estimator (MLE), semi-definite programming (SDP), bounded error estimator (BE), robust bounded error estimator (RBE). We

compare the alternative methods and found that LSCR provides the best localization result: its average error is smaller than other compared methods.

12.1 Summary and Contribution

WSNs have been widely studied in the past years, but research still needs to find solutions to some of the challenges posed by power limitation, dynamicity, and low computing clock rate. With the aim of contributing to the research on WSN, this thesis proposes new algorithms for both clock synchronization and localization. The detailed contributions of this thesis are summarized as follows.

- Chapter 3 described the formal model of the wireless sensor network and specified the problem of clock synchronization. The local physical clocks and the formal model of the wireless sensor network is defined. An error function is presented to evaluate the quality of the algorithm. Also, we summarize the existing state-of-the-art techniques such as ATS, RoATS, ATSP.

- Chapter 4 introduced time-wheel algorithm and drift compensation to extend the existing pulse-coupled synchronization [Függer et al., 2015a]. With our time-wheel algorithm, sensors can synchronize round number during consensus. With drift compensation, sensors estimate a better timing to broadcast pulse, which improves the accuracy of clock synchronization.

- Chapter 5 gave the simulation result that compare this work to the previous work [Függer et al., 2015a]. The result shows that the clock skew is reduced dramatically compared to the previous work. A random walk environment is simulated, which shows that with corr_i^2, sensors are able to remember the corrected frequency even if they do not have any neighboring sensors. A vehicle network is also simulated to show the benefit of adding corr_i^2.

- Chapter 8 introduced the tools of interval analysis. Interval analysis allow us to consider the uncertainty as bounds. With the inclusion function, the bounded

uncertainty can propagate to different domains. A contractor is able to reduce the solution space considering multiple inclusion functions. SIVIA algorithm is introduced to characterize the solution set.

- Chapter 9 described the localization algorithms. Numerical methods such as MLE and SDP with CRLB are discussed. Interval-based methods such as BE and RBE are also described. The main benefit and limitation of these algorithm is described.

- Chapter 10 proposed the localization algorithm using LSCR. With a user defined probability, the non-asymptotic confidence region NACR is defined under mild assumptions. With the help of Interval Analysis, the characterization is much faster when the required precision is high. Moreover, by using SIVIA, the outer and inner approximation of NACR is characterized, so the result is guaranteed. This chapter also discusses the implementation and the result of LSCR localization. Applying LSCR to solve localization problem achieve higher accuracy than the alternative approaches. This chapter also analyzed different kind of LSCR arrangement and find the best correlation function arrangement.

12.2 Future Work

This thesis represents a small contribution to the field of wireless sensor network. Therefore, future research is certainly needed. Some subjects where the research discussed in this thesis can continue are described in the following subsections.

12.2.1 Improve characterization speed of LSCR

The characterization of NACR using interval analysis is for the time being relatively time consuming. More efficient contractors are currently searched to speed-up the characterization process. Applications of the LSCR to recursive estimation is also considered. In this thesis, we used many contractors and tried to reduce the runtime.

However, the correlation functions are difficult to contractor, because there are too many occurrence of the same variable in these functions. With many occurrence, it is difficult for contractor to contract the boxes.

12.2.2 Applying LSCR to TOA or TDOA

This thesis applies LSCR to RSS signal measurements, one can also apply the technique to time of arrival (TOA) or time-difference of arrival (TDOA) localization. With TOA, the distance of sensors is calculated from the time of arrival as radiation signal travel with a known velocity. The transmission delay estimated from TOA or TDOA can also be used in clock synchronization algorithms.

12.2.3 Implement the Approaches in Real World

Simulation is not enough to micmic the noise samples in the real world environment. For future work, we intend to evaluate the algorithm using a practical wireless sensor network. The first step will be to select an appropriate communication hardware and to closely simulate it in a large scale network. With the real world environment, we can start to consider the measurements that is affected by different obstacles such as buildings in the vehicular network or walls in the indoor localization problem.

12.2.4 Dynamic Pulse Frequency for Clock Synchronization

For the wireless sensor network, the power that drive the sensors is usually limited. With limited battery, it is difficult for sensors to explore the real world. Our proposed clock synchronization method handles the problem of clock skew with the fixed pulse frequency. If the frequency is dynamic, then sensors are able to reduce the pulse frequency after the clock is synchronized, so the energy is preserved.

12.2.5 Considering Transmission Delay

In the proposed method for clock synchronization, we assume the transmission delay is zero because sensors only communicate with pulses. However, in order to communicate with pulses, the new protocol for wireless sensor network has to be setup. If the proposed clock synchronization algorithm is applied on the regular WIFI or ZEEBEE, then transmission delay is unavoidable. In order to increase the robustness of the algorithm, we can also consider transmission delay in the future work.

12.2.6 Combining Approach

In the thesis, the clock synchronization and localization are two separated methods. However, many research shows that the clock synchronization and localization face the same difficulties and can be solved together. The both of the algorithms need to consider transmit delay. It is possible for sensors to exchange messages for both clock synchronization and localization at once to reduce the required bandwidth. For moving sensors, better clock synchronization reduce the clock skew, which leads to better localization result. It is interesting to merge the two different techniques to solve the problems at once, so the algorithm becomes more efficient and robust.

12.3 Publications

- Guaranteed confidence region characterization for source localization using LSCR.
  - Han, Cheng-Yu, Michel Kieffer, and Alain Lambert.
  - Summer Workshop on Interval Methods. 2016.
- Pulse synchronization for vehicular networks.
  - Cheng-Yu Han, Thomas Nowak, and Alain Lambert.
  - In Intelligent Vehicle Symposium, 2018b.

- Guaranteed confidence region characterization for source localization using RSS measurements.
  - Cheng-Yu Han, Michel Kieffer, and Alain Lambert.
  - Signal Processing, 152:104–117, 2018a.

Bibliography

Ian F Akyildiz, Weilian Su, Yogesh Sankarasubramaniam, and Erdal Cayirci. Wireless sensor networks: a survey. *Computer networks*, 38(4):393–422, 2002.

Jamal N Al-Karaki and Ahmed E Kamal. Routing techniques in wireless sensor networks: a survey. *IEEE wireless communications*, 11(6):6–28, 2004.

Michael Behrisch, Laura Bieker, Jakob Erdmann, and Daniel Krajzewicz. Sumo–simulation of urban mobility. In *International Conference on Advances in System Simulation, Barcelona, Spain*, volume 42, 2011.

Alexander Bertrand. Applications and trends in wireless acoustic sensor networks: A signal processing perspective. In *IEEE Symposium on Communications and Vehicular Technology*, pages 1–6, 2011.

M. C. Campi and E. Weyer. Guaranteed non-asymptotic confidence regions in system identification. *Automatica*, 41(10):1751–1764, 2005.

M. C. Campi and E. Weyer. Non-asymptotic confidence sets for input-output transfer functions. In *Proceedings of IEEE Conference on Decision and Control*, pages 157–162, 2006.

M. C. Campi and E. Weyer. Non-asymptotic confidence sets for the parameters of linear transfer functions. *IEEE Transactions on Automatic Controls*, 55(12): 2708–2720, 2010.

Ruggero Carli and Sandro Zampieri. Network clock synchronization based on the second-order linear consensus algorithm. *IEEE Transactions on Automatic Control*, 59(2):409–422, 2014.

G. Chabert and L. Jaulin. Contractor programming. *Artificial Intelligence*, 173(11): 1079–1100, 2009.

Bernadette Charron-Bost. Orientation and connectivity based criteria for asymptotic consensus. *arXiv preprint arXiv:1303.2043*, 2013.

Flaviu Cristian. Probabilistic clock synchronization. *Distributed computing*, 3(3): 146–158, 1989.

B. C. Csáji, M. C. Campi, and E. Weyer. Non-asymptotic confidence regions for the least-squares estimate. In *IFAC Symposium on System Identification*, pages 227–232, Brussels, Belgium, 2012.

M. Dalai, E. Weyer, and M. C. Campi. Parameter identification for nonlinear systems: Guaranteed confidence regions through LSCR. *Automatica*, 43:1418 – 1425, 2007a.

Marco Dalai, Erik Weyer, and Marco C Campi. Parameter identification for nonlinear systems: Guaranteed confidence regions through lscr. *Automatica*, 43(8):1418–1425, 2007b.

Jeremy Elson and Kay Römer. Wireless sensor networks: A new regime for time synchronization. *ACM SIGCOMM Computer Communication Review*, 33(1):149–154, 2003.

Jeremy Elson, Lewis Girod, and Deborah Estrin. Fine-grained network time synchronization using reference broadcasts. *ACM SIGOPS Operating Systems Review*, 36 (SI):147–163, 2002.

Matthias Függer, Thomas Nowak, and Bernadette Charron-Bost. Diffusive clock synchronization in highly dynamic networks. In *Information Sciences and Systems (CISS), 2015 49th Annual Conference on*, pages 1–6. IEEE, 2015a.

Matthias Függer, Thomas Nowak, and Bernadette Charron-Bost. Diffusive clock synchronization in highly dynamic networks. In *Conference on Information Sciences and Systems*, pages 1–6. IEEE, 2015b.

Saurabh Ganeriwal, Ram Kumar, and Mani B Srivastava. Timing-sync protocol for sensor networks. In *ACM Conference on Embedded networked sensor systems*, pages 138–149, 2003.

Deepak Ganesan, Sylvia Ratnasamy, Hanbiao Wang, and Deborah Estrin. Coping with irregular spatio-temporal sampling in sensor networks. *ACM SIGCOMM Computer Communication Review*, 34(1):125–130, 2004.

Emanuele Garone, Andrea Gasparri, and Francesco Lamonaca. Clock synchronization protocol for wireless sensor networks with bounded communication delays. *Automatica*, 59:60–72, 2015.

S. Gezici. A survey on wireless position estimation. *Wireless personal communications*, 44(3):263–282, 2008.

M. R. Gholami, H. Wymeersch, E. G. Ström, and M. Rydström. Wireless network positioning as a convex feasibility problem. *Wireless Communications and Networking*, 2011(1):1–15, 2011.

L. Gordon. Completely separating groups in subsampling. *The Annals of Statistics*, 2(3):572–578, 1974.

John Hajnal and MS Bartlett. Weak ergodicity in non-homogeneous markov chains. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 54, pages 233–246. Cambridge University Press, 1958.

E. Halbwachs and D. Meizel. Bounded-error estimation for mobile vehicule localization. *Computational Engineering in Systems Applications*, pages 1005–1010, 1996.

Cheng-Yu Han, Michel Kieffer, and Alain Lambert. Guaranteed confidence region characterization for source localization using rss measurements. *Signal Processing*, 152:104–117, 2018a.

Cheng-Yu Han, Thomas Nowak, and Alain Lambert. Pulse synchronization for vehicular networks. In *intelligent vehicle symposium*, 2018b.

Jinchi Han, Jun Hu, Yang Yang, Zhongxu Wang, Shan X Wang, and Jinliang He. A nonintrusive power supply design for self-powered sensor networks in the smart grid by scavenging energy from ac power line. *IEEE Transactions on Industrial Electronics*, 62(7):4398–4407, 2015.

Jianping He, Peng Cheng, Ling Shi, Jiming Chen, and Youxian Sun. Time synchronization in wsns: A maximum-value-based consensus approach. *IEEE Transactions on Automatic Control*, 59(3):660–675, 2014.

Kai-Chun Huang, Yun-Sheng Yen, Han-Chieh Chao, and Tree-Clustered Data Gathering Protocol. for wireless sensor networks. *Journal of software*, 4(7):1282–1291, 2003.

N. Iliev and I. Paprotny. Review and comparison of spatial localization methods for low-power wireless sensor networks. *IEEE Sensors Journal*, 15(10):5971–5987, 2015.

L. Jaulin and E. Walter. Guaranteed nonlinear parameter estimation from bounded-error data via interval analysis. *Mathematics and Computers in Simulation*, 35(2): 123–137, 1993a.

L. Jaulin and E. Walter. Set inversion via interval analysis for nonlinear bounded-error estimation. *Automatica*, 29(4):1053–1064, 1993b.

L. Jaulin, E. Walter, and O. Didrit. Guaranteed robust nonlinear parameter bounding. In *Computational Engineering in Systems Applications*, 1996.

L. Jaulin, M. Kieffer, O. Didrit, and E. Walter. *Applied interval analysis: with examples in parameter and state estimation, robust control and robotics*, volume 1. Springer Science & Business Media, 2001.

M. Kieffer and E. Walter. Guaranteed characterization of exact non-asymptotic confidence regions as defined by LSCR and SPS. *Automatica*, 50(2):507–512, 2014.

S. Kim, H. Jeon, and J. Ma. Robust localization with unknown transmission power for cognitive radio. In *Military Communications Conference*, pages 1–6. IEEE, 2007.

Hermann Kopetz and Wilhelm Ochsenreiter. Clock synchronization in distributed real-time systems. *IEEE Transactions on Computers*, 100(8):933–940, 1987.

J. H. Lee and R. M. Buehrer. Location estimation using differential RSS with spatially correlated shadowing. In *IEEE International Conference on Global Telecommunications*, 2009.

J. Léger and M. Kieffer. Guaranteed robust distributed estimation in a network of sensors. In *IEEE International Conference on Acoustics Speech and Signal Processing*, 2010.

Qun Li and Daniela Rus. Global clock synchronization in sensor networks. *IEEE Transactions on computers*, 55(2):214–226, 2006.

X. Li, Z. D. Deng, L. T. Rauchenstein, and T. J. Carlson. Contributed review: Source-localization algorithms and applications using time of arrival and time difference of arrival measurements. *Review of Scientific Instruments*, 87(4):041502, 2016.

H. Lohrasbipeydeh, T. A. Gulliver, and H. Amindavar. A minimax SDP method for energy based source localization with unknown transmit power. *IEEE Wireless Communications Letters*, 3(4):433–436, 2014.

David L Mills. Internet time synchronization: the network time protocol. *IEEE Transactions on communications*, 39(10):1482–1493, 1991.

David L Mills. The network computer as precision timekeeper. Technical report, DELAWARE UNIV NEWARK DEPT OF ELECTRICAL ENGINEERING, 1996.

David L Mills. Adaptive hybrid clock discipline algorithm for the network time protocol. *IEEE/ACM Transactions on Networking (TON)*, 6(5):505–514, 1998.

R. E. Moore. Interval analysis. englewood cliff, 1966.

F. Mourad, H. Snoussi, F. Abdallah, and C. Richard. Anchor-based localization via interval analysis for mobile ad-hoc sensor networks. *IEEE Transactions on Signal Processing*, 57(8):3226–3239, 2009.

F. Mourad, H. Snoussi, M. Kieffer, and C. Richard. Robust interval-based localization algorithms for mobile sensor networks. *International Journal of Distributed Sensor Networks*, 8(1):303895, 2011a.

Farah Mourad, Hichem Snoussi, Michel Kieffer, and Cédric Richard. Robust interval-based localization algorithms for mobile sensor networks. *International Journal of Distributed Sensor Networks*, 2012, 2011b.

Saad Ahmed Munir, Biao Ren, Weiwei Jiao, Bin Wang, Dongliang Xie, and Jian Ma. Mobile wireless sensor network: Architecture and enabling technologies for ubiquitous computing. In *IEEE Conference on Advanced Information Networking and Applications*, volume 2, pages 113–120, 2007.

Y. Okumura, E. Ohmori, T. Kawano, and K. Fukuda. Field strength and its variability in VHF and UHF land-mobile radio service. *Rev. Elec. Commun. Lab*, 16 (9):825–73, 1968.

OpenStreetMap contributors. Planet dump retrieved from https://planet.osm.org . https://www.openstreetmap.org, 2017.

R. W. Ouyang, A. K. Wong, and C. Lea. Received signal strength-based wireless localization via semidefinite programming: Noncooperative and cooperative schemes. *IEEE Transactions on Vehicular Technology*, 59(3):1307–1318, 2010.

Uroš Pešović, Jože Mohorko, Karl Benkič, and Žarko Čučej. Effect of hidden nodes in ieee 802.15. 4/zigbee wireless sensor networks. In *Proceedings of telecommunications forum*, pages 161–164, 2009.

L. Pronzato and E. Walter. Robustness to outliers of bounded-error estimators and consequences on experiment design. In *Bounding Approaches to System Identification*, pages 199–212, 1996.

G Rahimi, M. R. Danaee, and S. Bayat. A generalized total least squares algorithm for RSS-based localization with unknown path-loss model parameters. In *Iranian Conference on Electrical Engineering*, pages 521–524, 2016.

Luca Schenato and Federico Fiorentin. Average timesynch: A consensus-based protocol for clock synchronization in wireless sensor networks. *Automatica*, 47(9): 1878–1886, 2011.

Luca Schenato and Giovanni Gamba. A distributed consensus protocol for clock synchronization in wireless sensor network. In *Proceedings of IEEE Conference on Decision and Control*, pages 2289–2294. IEEE, 2007.

Osvaldo Simeone and Umberto Spagnolini. Distributed time synchronization in wireless sensor networks with coupled discrete-time oscillators. *EURASIP Journal on Wireless Communications and Networking*, 2007(1):057054, 2007.

Osvaldo Simeone, Umberto Spagnolini, Gesualdo Scutari, and Yeheskel Bar-Ness. Physical-layer distributed synchronization in wireless networks and applications. *Physical Communication*, 1(1):67–83, 2008.

Roberto Solis, Vivek S Borkar, and PR Kumar. A new distributed time synchronization protocol for multihop wireless networks. In *Conference on Decision and Control*, pages 2734–2739. IEEE, 2006.

Weilian Su and Ian F Akyildiz. Time-diffusion synchronization protocol for wireless sensor networks. *IEEE/ACM Transactions on Networking*, 13(2):384–397, 2005.

Kun Sun, Peng Ning, and Cliff Wang. Fault-tolerant cluster-wise clock synchronization for wireless sensor networks. *Transactions on Dependable and Secure Computing*, 2(3):177–189, 2005.

Bharath Sundararaman, Ugo Buy, and Ajay D Kshemkalyani. Clock synchronization for wireless sensor networks: a survey. *Ad hoc networks*, 3(3):281–323, 2005.

Amulya Ratna Swain and RC Hansdah. A model for the classification and survey of clock synchronization protocols in wsns. *Ad Hoc Networks*, 27:219–241, 2015.

Affan A Syed, John S Heidemann, et al. Time synchronization for high latency acoustic networks. In *Infocom*, volume 6, pages 1–12, 2006.

K. C. Toh, M. J. Todd, and R. H. Tütüncü. SDPT3-a matlab software package for semidefinite programming, version 1.3. *Optimization methods and software*, 11 (1-4):545–581, 1999.

S. Tomic, M. Beko, and R. Dinis. RSS-based localization in wireless sensor networks using convex relaxation: Noncooperative and cooperative schemes. *IEEE Transactions on Vehicular Technology*, 64(5):2037–2050, 2015.

D. J. Torrieri. Statistical theory of passive location systems. In *Autonomous robot vehicles*, pages 151–166. 1990.

R. M. Vaghefi, M. R. Gholami, R. M. Buehrer, and E. G. Strom. Cooperative received signal strength-based sensor localization with unknown transmit powers. *IEEE Transactions on Signal Processing*, 61(6):1389–1403, 2013.

Nicola Varanese, Umberto Spagnolini, and Yeheskel Bar-Ness. Distributed frequency-locked loops for wireless networks. *IEEE Transactions on Communications*, 59(12): 3440–3451, 2011.

G. Wang and K. Yang. Efficient semidefinite relaxation for energy-based source localization in sensor networks. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 2257–2260, 2009.

J. Wang, Q. Gao, Y. Yu, L. Cheng, P.and Wu, and H. Wang. Robust device-free wireless localization based on differential RSS measurements. *IEEE transactions on industrial electronics*, 60(12):5943–5952, 2013.

Yongqiang Wang, Felipe Nunez, and Francis J Doyle. Energy-efficient pulse-coupled synchronization strategy design for wireless sensor networks through reduced idle listening. *IEEE Transactions on Signal Processing*, 60(10):5293–5306, 2012.

Jianshe Wu, Licheng Jiao, and Ranran Ding. Average time synchronization in wireless sensor networks by pairwise messages. *Computer Communications*, 35(2):221–233, 2012.

Yik-Chung Wu, Qasim Chaudhari, and Erchin Serpedin. Clock synchronization of wireless sensor networks. *IEEE Signal Processing Magazine*, 28(1):124–138, 2011.

Wei Ye, John Heidemann, and Deborah Estrin. An energy-efficient MAC protocol for wireless sensor networks. In *Proceedings of IEEE Infocom*, volume 3, pages 1567–1576, 2002.

Jennifer Yick, Biswanath Mukherjee, and Dipak Ghosal. Wireless sensor network survey. *Computer networks*, 52(12):2292–2330, 2008.

Ilan Ziskind and Mati Wax. Maximum likelihood localization of multiple sources by alternating projection. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 36(10):1553–1560, 1988.

université PARIS-SACLAY
ÉCOLE DOCTORALE
Sciences et technologies de l'information et de la communication (STIC)

Titre : Synchronisation d'horloge et localisation pour réseau de capteurs sans fil

Mots clés : synchronisation d'horloge, localisation, couplage d'impulsions synchronisation, LSCR, réseau de capteurs sans fil

Résumé : Les réseaux de capteurs sans fil (WSN) jouent un rôle important dans des applications telles que la surveillance de l'environnement, le suivi de sources et la surveillance médicale. Dans les WSN, les capteurs ont la capacité d'acquérir des mesures, de réaliser des traitements simples et de communiquer les résultats de ces traitements. Pour effectuer ces tâches, la localisation et la synchronisation sont essentiels.

Les WSN ont été largement étudiés ces dernières années et la littérature scientifique rapporte de nombreux résultats qui les rendent applicables pour de nombreuses applications. Pour d'autres, la recherche doit encore trouver des solutions à certains des défis posés par la limitation énergétique, la dynamicité et la faible puissance de calcul. Dans le but de contribuer à la recherche sur les WSN, cette thèse propose de nouveaux algorithmes pour la synchronisation d'horloge et la localisation.

La synchronisation d'horloge est utile pour résoudre les problèmes de localisation, de fusion de données et de consensus. En appliquant l'algorithme de synchronisation d'horloge, les capteurs etablissent un consensus temporel et travaillent donc au même rythme.

Compte tenu de la dynamicité, des faibles capacités de calcul et de la parcimonie des WSN, un nouvel algorithme de synchronisation décentralisée à impulsions couplées est proposé pour réduire le décalage d'horloge de la synchronisation. L'avantage de ce type d'algorithme est que les capteurs échangent des impulsions au lieu de paquets, de sorte que non seulement la communication est efficace, mais aussi robuste à toute défaillance des capteurs dans le réseau.

La localisation de capteurs a été largement étudiée dans la littérature scientifique. Cependant, la qualité et la précision de la localisation peuvent encore être améliore. Cette thèse applique l'algorithme LSCR (Régression de régions corrélées à signes dominants) au problème de localisation. Avec LSCR, on évalue des régions de confiance avec des niveaux de confiance prescrits, qui fournissent non seulement on emplacement mais aussi la confiance en cet emplacement. Dans cette thèse, plusieurs approches de localisation sont implémentées et comparées. Le résultat de la simulation montre que, sous hypothèses modérées, LSCR obtient des résultats compétitifs par rapport à d'autres méthodes.

Cette thèse se concentre sur deux problèmes importants liés à WSN: la localisation et la synchronisation de l'heure. Pour la synchronisation d'horloge, la robustesse de l'algorithme est essentielle. Dans WSN, les capteurs mobiles et à faible coût provoquent des erreurs de communication. Compte tenu de la dynamicité, un algorithme distribué est proposé, qui prolonge les travaux précédents de Függer et al. (2015) et réduire le décalage d'horloge. Dans la nouvelle synchronisation à couplage d'impulsions, un algorithme de roue temporelle est introduit pour que les capteurs échangent le nombre de tours et calculent la moyenne des impulsions du même tour. De plus, une compensation de dérive est appliquée pour réduire le décalage d'horloge. L'analyse de convergence montre que la compensation de dérive réduit le décalage d'horloge et améliore le taux de convergence. Quand on considère un WSN hautement dynamique, l'algorithme converge plus rapidement que dans les travaux précédents. L'algorithme proposé est appliqué avec succès à un réseau de véhicules.

Pour la deuxième partie de la thèse, nous estimons l'emplacement de chaque capteur à l'aide de LSCR, ce qui produit une région de confiance non asymptotique. L'application de la correction LSCR au problème de localisation présente l'avantage que les hypothèses sur le bruit du signal RSS sont faibles, de sorte que la région de confiance peut être estimée avec précision. Nous analysons différentes fonctions de corrélation de LSCR et avons constaté que seule une fonction de corrélation peut produire une région de confiance satisfaisante. Nous mettons également en œuvre d'autres algorithmes de localisation, tels que l'estimateur de maximum de vraisemblance (MLE), la programmation semi-définie (SDP), l'estimateur d'erreur bornée (BE), l'estimateur robuste d'erreur bornée (RBE). Nous comparons les méthodes alternatives et avons constaté que LSCR fournit le meilleur résultat de localisation: son erreur moyenne est inférieure à celle des autres méthodes comparées.

Université Paris-Saclay
Espace Technologique / Immeuble Discovery
Route de l'Orme aux Merisiers RD 128 / 91190 Saint-Aubin, France

université PARIS-SACLAY
ÉCOLE DOCTORALE
Sciences et technologies de l'information et de la communication (STIC)

Title : Clock Synchronization and Localization for Wireless Sensor Network

Keywords : clock synchronization, localization, pulse coupled synchronization, LSCR, wireless sensor network

Abstract : Wireless sensor networks (WSNs) play an important role in applications such as environmental monitoring, source tracking, and health care, etc . In WSN, sensors have the ability to perform data sampling, distributed computing and information fusion. To perform such complex tasks, clock synchronization and localization are two fundamental and essential algorithms.

WSNs have been widely studied in the past years, and the scientific literature reports many outcomes that make them applicable for various fields such as environmental monitoring, health care, and the internet of things. For some others, research still needs to find solutions to some of the challenges posed by the limitations of sensors, such as battery limitation, dynamicity, and low computing clock rate. With the aim of contributing to the research on WSN, this thesis proposes new algorithms for both clock synchronization and localization.

Synchronized clocks are useful for many reasons. WSN is often designed to realize some synchronized behavior, especially in real-time processing in factories, aircraft, space vehicles, and military applications. For clock synchronization, sensors synchronize their local physical clock to perform data fusion. By applying the clock synchronization algorithm, sensors synchronize the time difference and therefore work at the same rate. The difficulty comes from the fact that clocks embedded on WSN are usually low cost, and prone to drift.

In view of dynamicity, low computing and low connectivity of WSN, a new pulse-coupled decentralized synchronization algorithm is proposed to reduce the clock skew of the synchronization. The benefit of this kind of algorithm is that sensors only exchange zero-bit pulse instead of packets, so not only the communication is efficient but also robust to any failure of the sensors in the network. The proposed synchronization algorithm is able to tolerant a low connectivity dynamic WSN with mush smaller clock skew than its previous work.

Localization of sensors has been widely studied. However, the quality and the accuracy of the localization still have a large room to improve. Characterization of estimation uncertainty is usually overlooked. Mainly asymptotic techniques are employed, assuming that the noise is Gaussian and that many measurements are available. These hypotheses are seldom satisfied. This thesis apply Leave-out Sign-dominant Correlated Regions (LSCR) algorithm to localization problem. With LSCR, one evaluates the accurate estimates of confidence regions with prescribed confidence levels, which provide not only the location but also the confidence of the estimation. In this thesis, several localization approaches are implemented and compared. The results show under mild assumptions, LSCR obtains competitive localization performance compared to other methods.

Université Paris-Saclay
Espace Technologique / Immeuble Discovery
Route de l'Orme aux Merisiers RD 128 / 91190 Saint-Aubin, France