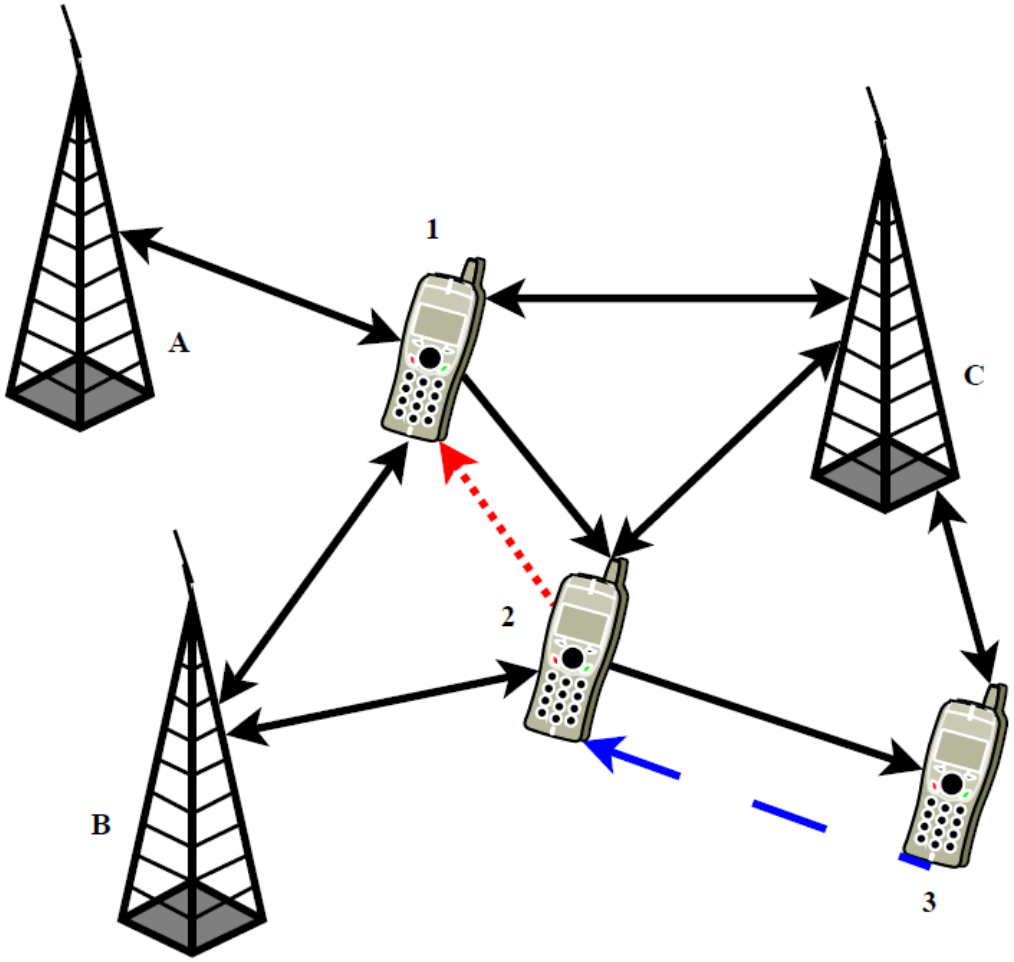
CHALMERS



Censoring for cooperative positioning

Master of Science Thesis in Communication Engineering

KALLOL DAS

Communication Systems Group
Department of Signals and Systems
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden, 2010
Report No. EX081/2010

REPORT NO. 081/2010

Censoring for Cooperative Positioning

Thesis for the degree of Master of Science

KALLOL DAS

CHALMERS

Communication Systems Group
Department of Signals and Systems
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2010

Censoring for Cooperative Positioning

KALLOL DAS

Technical report no. EX081/2010

Department of Signals and Systems
Chalmers University of Technology
SE-412 96 Gothenburg
Sweden
Telephone + 46 (0)31-772 1000

Cover:
[An abstract representation of censoring for cooperative positioning. The towers represent the positions of anchor nodes and the cell phones indicate the positions of agents. The connections between different nodes are shown by the black arrows. The blue arrow represents the transmit censoring and the red arrow represents the receive censoring scenario.]

Abstract

Cooperative positioning is an emerging topic in wireless sensor networks and navigation. It can improve the positioning accuracy and coverage in GPS-challenged conditions such as inside tunnels, in urban canyons, and indoors. Different algorithms have been proposed relying on iteratively exchanging and updating positional information. Although cooperative positioning can provide excellent accuracy, it suffers from high computational complexity due to a lot of incoming information from the neighboring devices. Cooperation in a dense network also increases network traffic. For practical implementation of cooperative positioning, a less complex algorithm with low network traffic is necessary. The complexity of the positioning algorithms are directly related to the number of used links in the iterative update phase. Not all of the incoming information from neighboring devices are required for precise positioning. In addition, the network traffic depends on the network geometry, type of positioning algorithm, number of transmissions per node. As the network geometry and type of positioning algorithm are predefined, we can control the network traffic only by reducing the number of transmissions per node. In this thesis we propose an algorithm that can minimize the number of used links for update and also the network traffic without significant performance degradation. We show that information that is not reliable should not be shared, and information that is not informative should not be used. This naturally leads to censoring schemes. We consider different censoring schemes based on the Cramér Rao bound (CRB). We find that by blocking the broadcasts of the nodes that don't have reliable estimates (transmit censoring) and selecting the best usable links after receiving signals from neighbors (receive censoring), complexity and network traffic can be reduced significantly without hampering the positioning performance or latency.

Keywords: Indoor positioning, Link selection, Cooperative positioning, Distributed wireless localization, Cramér Rao bound, Censoring, reduced network traffic.

Contents

List of Figures

List of Tables

List of Algorithms

Acknowledgments

"Break down a complex problem in to several simple parts, solve the small parts at first then combine all parts."- this is the first lesson that I got from my adviser Prof. Henk Wymeersch. I would like to express my sincere gratitude to Henk Wymeerch to give me the opportunity to work with him on positioning. He taught me how to conduct a research and also how to present the research in an interesting way. His friendly attitude and interesting way to make difficult problems to easier ones really helped me a lot to keep my concentration on the project. Throughout the entire project period his depth of knowledge on the topic, suggestions for continuous improvement, encouragement and supervision helped me to complete this research work productively. Besides, He has been very concerned on the project progress, which has made this project as a successful one.

Special thanks Prof. Erik Ström for arranging fund for my PIMRC'10 conference trip. It was really amazing experience that I gained from this conference.

There some important people in my life. Without their unconditional love and support it was impossible to reach this far. I pay my deepest gratitude to my parents for their strong support in every stages of my life. I am thankful to my fiancee, Mahua for her patience and encouragement during the entire project period. I am also grateful to my cousin Ranjita and younger brother Prodyut for their inspiration in every stage of my life.

I am indebted to my friends Amalesh, Asif, Shajib, Shuvo, Rupok, and Tilak to support me during the study period at Chalmers.

Kallol Das
Gothenburg, November 16, 2010.

Chapter 1

Introduction

1.1 Background

Suppose you are planning to explore a new city by your car and you don't have a GPS receiver- nowadays nobody can think about such a situation. In the beginning, the main interest for wireless positioning has been for military [1]. At present positional information has been considered of great importance not only in navigation but also in many applications such as search and rescue, disaster management, sensor networks, traffic routing, fleet management, supply chain monitoring etc. [2]. The Global Positioning System (GPS) can provide reliable positioning in outdoor scenarios (Figure 1.1).

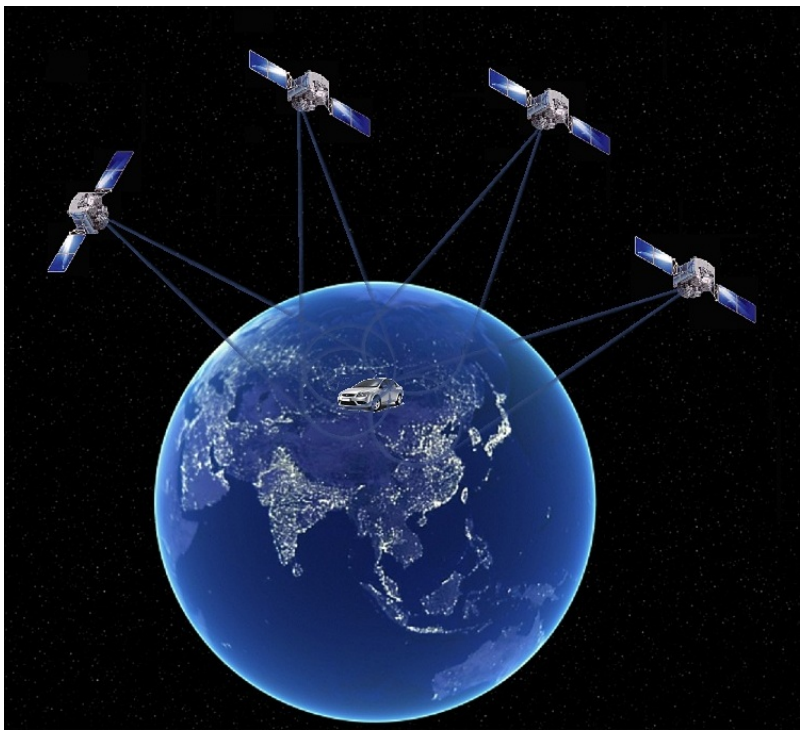


Figure 1.1: Global positioning system.

1.2 Techniques

Every positioning technique relies on reference points for localization. In a network, we have two types of nodes: anchors, which know their positions, and agents, whose positions have to be estimated. Anchor nodes act as reference points for positioning. Agents can either estimate the distance/angle between the anchor and itself by some ranging protocol or from the received signal strength. After having these measurements, the agent can determine its own position by trilateration/triangulation or by some similar methods. A simple positioning technique using trilateration is presented in Figure 1.2. There are many types of positioning techniques which

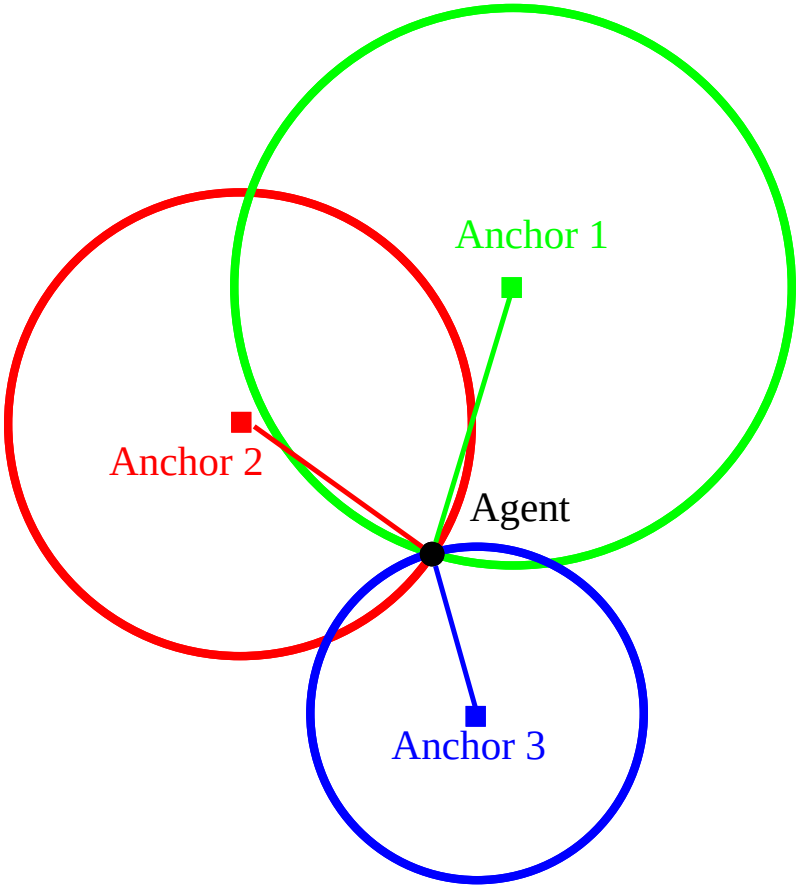


Figure 1.2: Positioning by trilateration.

can be divided into two major categories:

1. Selfish Positioning: In this technique, the agents only rely on anchor nodes for positioning.

2. Cooperative Positioning: In addition to communicate with anchors, here the agents share positional information and do ranging between themselves for positioning. More details about cooperative positioning are presented in next section.

1.3 Cooperative Positioning

The performance of range based positioning depends on the distance estimation accuracy and coverage by reference nodes. Conventional range based positioning may fail in some applications such as indoor positioning due to the lack of connectivity with sufficient reference nodes. In these cases cooperative positioning can improve the positioning performance by exchanging full statistical information between devices [3]. In Figure 1.3 such a scenario has been presented. Both of the agents 1 and 2 have connectivity from only two anchors A, B and C, D respectively, so the agents can not be localized by trilateration. However by cooperation between agents, this problem can be solved.

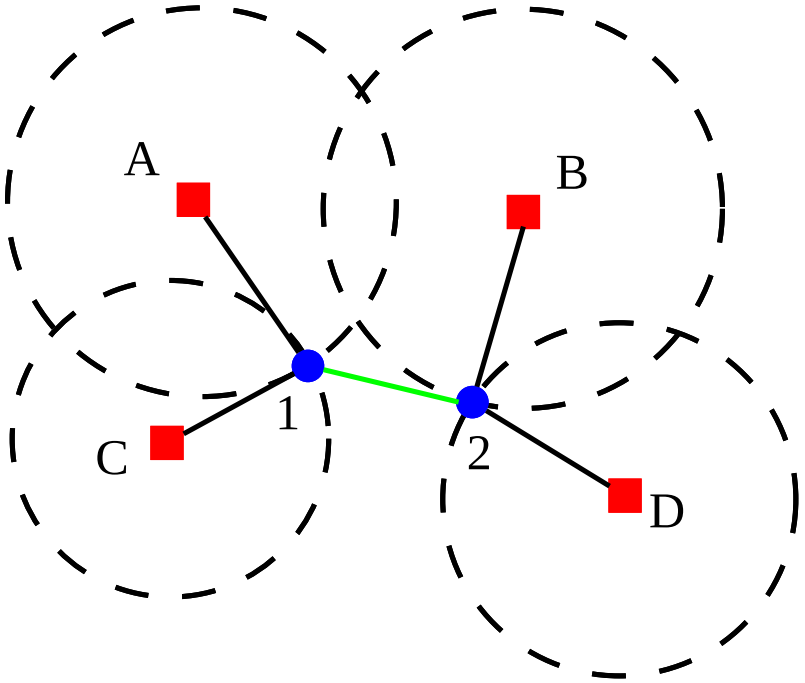


Figure 1.3: Benefit of cooperative positioning.

1.4 Censoring in Cooperative Positioning

Despite improved performance, cooperative positioning has high computational complexity due to huge amount of incoming information from the neighboring devices. The network traffic is also high in cooperative positioning in comparison to non-cooperative positioning for the same reason. It may be possible to have good position estimate by using only reliable information from the neighbors without degrading the performance of positioning significantly. Now the important question is how to select these so-called best links.

Insight-

1. The link selection process should have low overhead.

2. The overall process should be robust.

1.5 Related Works

The authors of [4] proposed to use the closest reference nodes from the position of node in consideration. The closest reference nodes may not be the best for positioning as the geometric location of these nodes also effects the performance of positioning. These problem has partially been solved in [5, 6], where Cramér Rao bound has been used to choose the best reference nodes. In [7, 8] the authors apply geometric dilution of precision (GDOP) to select the best four satellites for GPS receiver which is comparable to use of Cramér Rao bound. [1]

1.6 Goal

In typical non-cooperative positioning any node can be localized if it has distance estimates from three reference nodes (considering perfect synchronization between all nodes). In case of cooperative positioning the number of available links for update may be in the order of 10-30 considering 20 m communication range. Most of the nodes do not have good estimate in the beginning of the iterative algorithm but after some iterations they can be localized well. None of the link selection methods addressed above are suitable for cooperative positioning as any agent may have uncertain estimate in certain iteration but its geometric location may be suitable for another agent.

In this thesis, we show that by censoring we can reduce the computation complexity and network traffic without performance degradation. This work was published in IEEE International Workshop on Advances in Positioning and Location- Enabled Communications (in conjunction with PIMRC'10), Istanbul, Turkey [9]. We also show that censoring schemes can be more meaningfully used to reduce the complexity of Bayesian positioning algorithm, where the nodes share full statistical positional information.

The remainder of this thesis is arranged as follows. In Chapter 2, we describe our model and assumptions. In Chapter 3 we describe two different types of positioning algorithms, the criterion and methodology of censoring have been presented in Chapter 4. Results from simulations are presented in Chapter 5. Finally in Chapter 6, we draw our conclusions with possible future extensions of this thesis.

[1] A GPS receiver may get signal from more than four satellites and at least four reference nodes are necessary to localize as the receiver is not synchronized with the satellites.

Chapter 2

Problem formulation

At first consider a wireless network with N nodes. In our model there are two types of nodes: anchors, which know their positions, and agents, whose positions have to be estimated. In distributed networks, agents iteratively update their position estimates. The anchor nodes act as reference points for positioning. In a cooperative environment the update phase of the agents depends on range-measurements between agents and anchors as well as on agent-to-agent measurements.

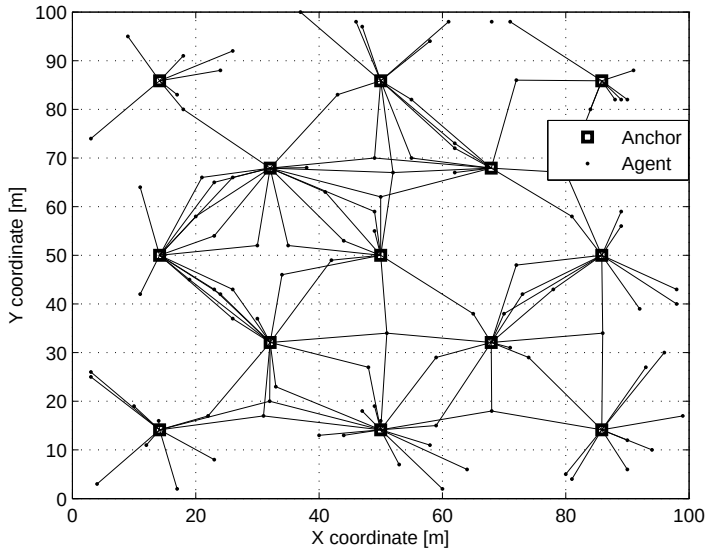


Figure 2.1: A typical non-cooperative network with 100 agents and 13 anchors; average connectivity = 1.55.

We denote by $\mathbf{x}_i$ the position of node i in the network and by $\hat{\mathbf{x}}_i$ the corresponding estimated position. $S_{\to i}$ is the set of nodes from which node i can receive signals. Based on a ranging protocol (e.g., time of arrival (TOA), time difference of arrival (TDOA), receive signal strength (RSS) etc.) [3] with node $j \in S_{\to i}$, node i can estimate the distance between itself and node j, $\hat{d}_{j\to i} = \left\|\mathbf{x}_i - \mathbf{x}_j\right\| + n_{j\to i}$, where $n_{j\to i}$ is the ranging noise. We assume $n_{j\to i} \sim \mathcal{N}\left(0, \sigma^2_{j\to i}\right)$

[3]. The goal of node i is to estimate its own position. Ideally, the positioning process should require low complexity and communication overhead per node as well as low latency.

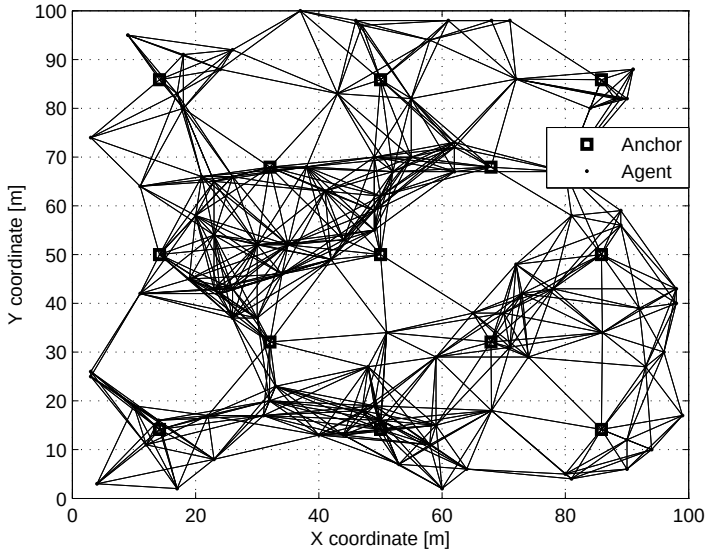


Figure 2.2: A typical cooperative network with 100 agents and 13 anchors; average connectivity = 13.67.

In our network, we consider 13 anchors and 100 agents, with a communication range of 20 m. In a non-cooperative environment very few agents can have connectivity with enough number of anchors to be localized. Most of the agents are connected to only one or two anchors which are shown in Figure 2.1. If the agents can communicate with each other with in the communication range i.e., in a cooperative environment, the network connectivity will be high as in Figure 2.2. It turns out that there is roughly ten-fold increase of usable links in the cooperative network, making cooperative positioning promising, but challenging to implement [9].

Chapter 3

Positioning algorithm

As mentioned earlier, we will only discuss cooperative positioning algorithms since a non-cooperative algorithm can be interpreted as a first iteration of a cooperative algorithm. The cooperative positioning algorithms can be grouped into two classes:

1. Non-Bayesian: The non-Bayesian algorithms don't use any prior location information during the update phase. E.g., cooperative least squares (LS), cooperative maximum likelihood (ML), etc.,

2. Bayesian: In Bayesian algorithms the agents share full statistical information (i.e. prior location information). E.g., SPAWN.

Now we will briefly describe the cooperative LS algorithm (non-Bayesian) and SPAWN (Bayesian) here which are also used in our simulation.

3.1 Cooperative Least Square Algorithm

We briefly describe the cooperative LS positioning Algorithm 3.1 using the notation introduced in Chapter 2. The LS estimator minimizes the following cost function with respect to $\mathbf{x} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]$,

$$\mathscr{C}_{\mathrm{LS}}(\mathbf{x}) = \sum_{i=1}^{N} \sum_{j \in S_{\to i}} \left\| \hat{d}_{j \to i} - \left\| \mathbf{x}_i - \mathbf{x}_j \right\| \right\|^2 .$$

The update phase of cooperative LS algorithm becomes (for a detailed derivation, see [3])

$$\hat{\mathbf{x}}_i^{(l)} = \hat{\mathbf{x}}_i^{(l-1)} + \delta_i^{(l)} \sum_{j \in S_{\to i}} (\hat{d}_{j \to i} - \tilde{d}_{j \to i}^{(l-1)})\, \tilde{\mathbf{q}}_{ij}^{(l-1)}, \tag{3.1}$$

where l is the iteration index, and $0 < \delta_i^{(l)} \ll 1$ is the step size corresponding to node i at iteration l,

$$\tilde{d}_{j \to i}^{(l-1)} = \left\| \hat{\mathbf{x}}_i^{(l-1)} - \hat{\mathbf{x}}_j^{(l-1)} \right\|,$$

and

$$\tilde{\mathbf{q}}_{ij}^{(l-1)} = \frac{\left(\hat{\mathbf{x}}_i^{(l-1)} - \hat{\mathbf{x}}_j^{(l-1)}\right)}{\left\|\hat{\mathbf{x}}_i^{(l-1)} - \hat{\mathbf{x}}_j^{(l-1)}\right\|}. \tag{3.2}$$

Algorithm 3.1 Cooperative least square positioning (at an arbitrary time slot).

given $\hat{\mathbf{x}}_i$, $\forall i$
for $l = 1$ to N_{iter} **do** {iteration index}
  nodes $i = 1$ to N **in parallel**
    perform *transmit censoring* [using Algorithm 4.1]
    broadcast current location estimate $\hat{\mathbf{x}}_i^{(l-1)}$
    receive positional information from neighbors $\hat{\mathbf{x}}_j^{(l-1)}$, $j \in S_{\to i}$
    select the best links by *receive censoring* [using Algorithm 4.2]
    update location estimate {only for agents}

$$\hat{\mathbf{x}}_i^{(l)} = \hat{\mathbf{x}}_i^{(l-1)} + \delta_i^{(l)} \sum_{j \in S_{\to i}} (\hat{d}_{j\to i} - \tilde{d}_{j\to i}^{(l-1)})\, \tilde{\mathbf{q}}_{ij}^{(l-1)}$$

  end parallel
end for

3.2 Sum Product Algorithm over a Wireless Network

Before describing the SPAWN algorithm we will introduce the basic idea of factor graphs and sum-product algorithm.

3.2.1 Factor Graphs

Factor graphs are a graphical representation of factorization of a function [10, 11]. Suppose a function $f(X_1, X_2, X_3, X_4)$ can be factorized as $f(X_1, X_2, X_3, X_4) = f_A(X_1) f_B(X_1, X_2) f_c(X_1, X_3, X_4)$, where X_1, X_2, X_3, X_4 are variables and f_A, f_B, f_c are the factors. We can represent this factorization as the factor graph shown in Figure 3.1.

3.2.2 Sum Product Algorithm

Sum product algorithm is an algorithm which is used to compute marginals of a function by message passing on its factor graph. Figure 3.2 shows a fragment of a factor graph where variable nodes are indicated by circles and factor nodes are indicated by squares. Messages are passing along the edges between variables and factors in both directions.

Here we will briefly describe the computational methods of the sum-product algorithm [10, 11]. Let us denote by $m_{A\to B}(X)$ the message sent from vertex A to vertex B. The message

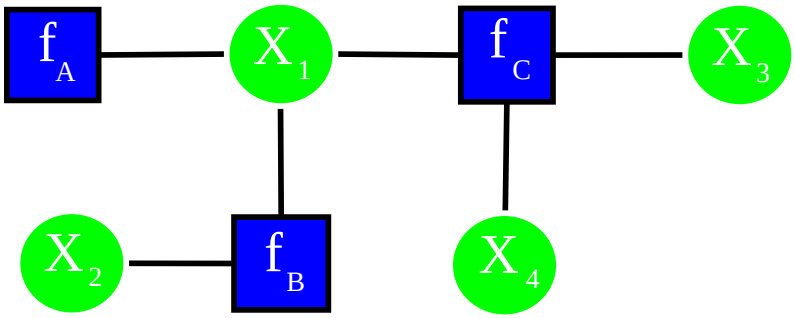


Figure 3.1: Factor graph of $f_A(X_1)f_B(X_1,X_2)f_c(X_1,X_3,X_4)$.

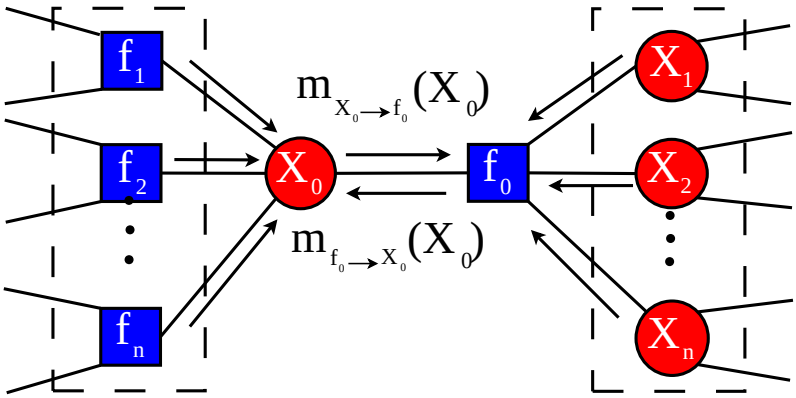


Figure 3.2: Message passing.

computation can be expressed as follows:
A message from a factor node to a variable node:

$$m_{f_0\to X_0}(X_0) = \sum_{X_1}\cdots\sum_{X_n} f(X_0,X_{1,},\cdots,X_n)\,.\prod_{i=1}^{n} m_{X_i\to f_0}(X_i). \tag{3.3}$$

A message from a variable node to a factor node:

$$m_{X_0\to f_0}(X_0) = \prod_{i=1}^{n} m_{f_i\to X_0}(X_0). \tag{3.4}$$

The marginal distribution of node X_0 is given by,

$$g(X_0) = m_{f_0\to X_0}(X_0).m_{X_0\to f_0}(X_0) = \sum_{X_1}\cdots\sum_{X_n} f(X_0,X_{1,},\cdots,X_n)\,. \tag{3.5}$$

The operation in Equation 3.5 is known as message filtering and the operation in Equation 3.4 is called message multiplication.

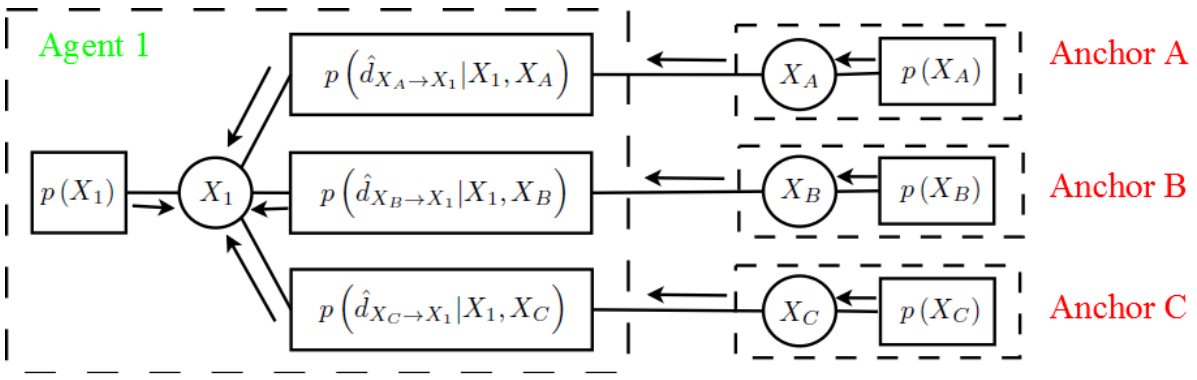


Figure 3.3: Net factor graph and its message passing.

3.2.3 Distributed Positioning Algorithm

We briefly describe here the sum product algorithm over a wireless network (Algorithm 3.2, SPAWN). SPAWN maps a factor graph onto the network topology and develops the message passing scheme over the network factor graph [3]. To create the factor graph, we need first to formulate the relationship between all variables in the network and factorize it. Then we create a net factor graph based on this factor graph by associating each node with its local information. We implement the sum-product algorithm over the net factor graph, therefore, for each agent in the positioning algorithm SPAWN, the updating of its own belief is based on the above mentioned rules in (3.4-3.5).

To demonstrate the algorithm, let us consider Figure 3.3 where node X_1 is connected to three anchors X_A, X_B, X_c. After knowing the distance estimates (from node X_1 to these three anchors) $\hat{d}_{X_A\to X_1}, \hat{d}_{X_B\to X_1}, \hat{d}_{X_C\to X_1}$ by range measurements we can factorize the posterior distribution as,

$$p\left(X_1,X_A,X_B,X_C|\hat{d}_{X_A\to X_1},\hat{d}_{X_B\to X_1},\hat{d}_{X_C\to X_1}\right) \propto p(X_1)\,p(X_A)\,p(X_B)\,p(X_C)\,.p\left(\hat{d}_{X_A\to X_1}|X_1,X_A\right) .p\left(\hat{d}_{X_A\to X_1}|X_1,X_B\right).p\left(\hat{d}_{X_A\to X_1}|X_1,X_C\right). \quad (3.6)$$

Each variable with the factors that contains its local information, resulting a net factor graph depicted in Figure 3.3, where arrows indicate the flow of messages.

Observe that there are two types of messages: the ones within the dashed-blocks that are computed within the nodes and the ones between the blocks that are passing over the link. The former are from factor to variable therefore we apply Equation 3.5 to compute them, while the latter messages are from variable to factor that need to obey the computation rule in Equation 3.4. Any representation of messages must enable efficient computation of these two operations. More specifically, given initialization of node i's prior as $p_i(X_i)$ and incoming messages

$m_{j\to i}(X_i)$ from the set of neighboring nodes $S_{\to i}$, the belief is

$$b_{X_i}(X_i) \propto p_i(X_i) \prod_{i \in S_{\to i}} m_{j\to i}(X_i), \tag{3.7}$$

which is simply multiplying every incoming messages with its previous prior.

Algorithm 3.2 SPAWN (at an arbitrary time slot).

1: **nodes** $i = 1$ to N **in parallel**
2: initialize $b_{X_i}^{(0)}(\cdot)$ from ranging information
3: **end parallel**
4: **for** $l = 1$ to N_{iter} **do** {iteration index}
5: **nodes** $i = 1$ to N **in parallel**
6: broadcast $b_{X_i}^{(l-1)}(\cdot)$
7: receive $b_{X_j}^{(l-1)}(\cdot)$ from neighbors $j \in S_{\to i}$
8: convert $b_{X_j}^{(l-1)}(\cdot)$ to a distribution over X_i using (3.4)

$$m_{j\to i}(x_i) \propto \int p(\hat{d}_{j\to i} | x_i, x_j) b_{X_j}^{(l-1)}(x_j) dx_j$$

9: select the best set of links by receive censoring (using algorithm 4.4)
10: compute new message using (3.4)

$$b_{X_i}^{(l)}(x_i) \propto p(x_i) \prod_{j \in S_{\to i}} m_{j\to i}(x_i)$$

11: choose which nodes should broadcast by transmit censoring (using Algorithm 4.3)
12: **end parallel**
13: **end for**

3.2.4 Message Representation

We use the parametric representation of messages where the distributions of true messages can be characterized from a family of some particular parametrized distributions (e.g., Gaussian distribution). We represent the message by three types of distributions as described in the following sub-sections.

3.2.4.1 A Single Donut Distribution

When an agent only talks to one anchor, (3.7) will return with a single donut distribution which is shown in the Figure 3.4. The distribution can be mathematically represented by [12]:

$$\mathscr{D}(\mathbf{x}; a, b, \sigma^2, \rho) = \frac{1}{C(\sigma^2, \rho)} \exp\left\{ -\frac{1}{2\sigma^2} \left[\sqrt{(x_1 - a)_2 + (x_2 - b)^2} - \rho \right]^2 \right\}, \tag{3.8}$$

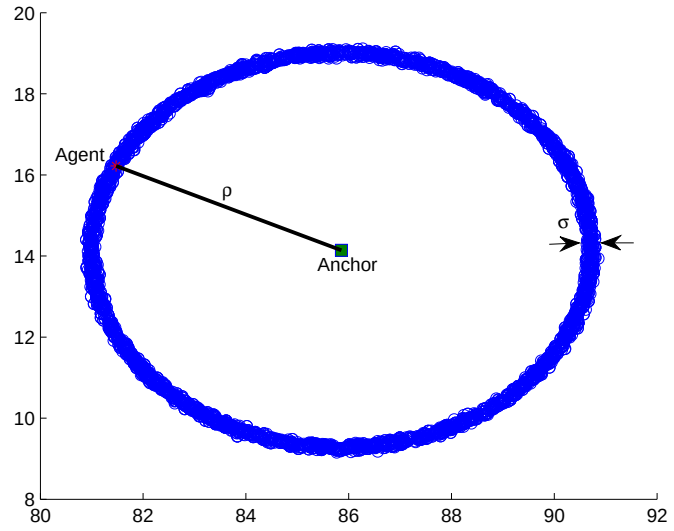


Figure 3.4: Single donut distribution.

where $[a,b]$ is the middle point of the distribution, (anchor's position in this case), ρ is the radius (measured distance between anchor and agent), σ^2 is the variance (measurement noise variance), and $C\left(\sigma^2,\rho\right)$ is a normalization constant.

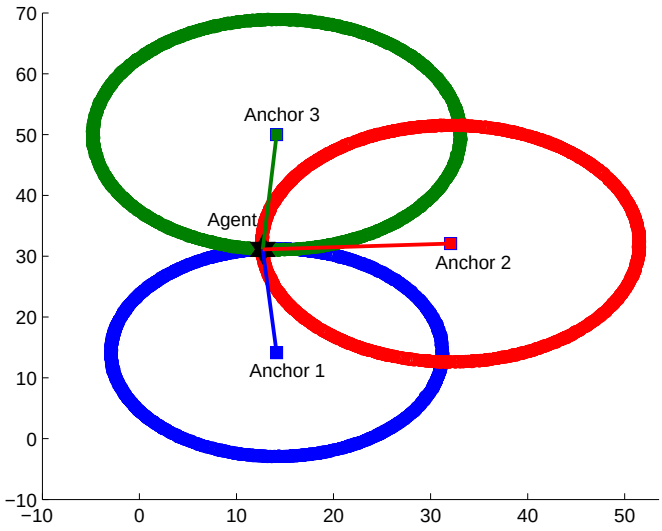


Figure 3.5: Single Gaussian distribution.

3.2.4.2 Single Gaussian Distribution

When an agent talks to three or more anchors, the distribution of the message can be approximated by a Gaussian distribution which is shown in Figure 3.5. We can represent a Gaussian distribution $\mathcal{N}\left([a,b],\sigma^2\right)$ using three parameters. $[a,b]$ can be estimated as the mean of the

donuts' intersections and σ^2 is the corresponding variance.

3.2.4.3 Mixture of two Gaussian Distribution

When an agent talks to two anchors, it will have a distribution like a mixture of two Gaussians which is shown in Figure 3.6. This distribution can be mathematically presented by

$$p_{GM}(x,y) = 0.5 \times \mathcal{N}\left([a_1,b_1],\sigma^2\right) + 0.5 \times \mathcal{N}\left([a_2,b_2],\sigma^2\right),$$

where $[a_1,b_1]$and $[a_2,b_2]$ are the midpoints of the two Gaussian distributions.

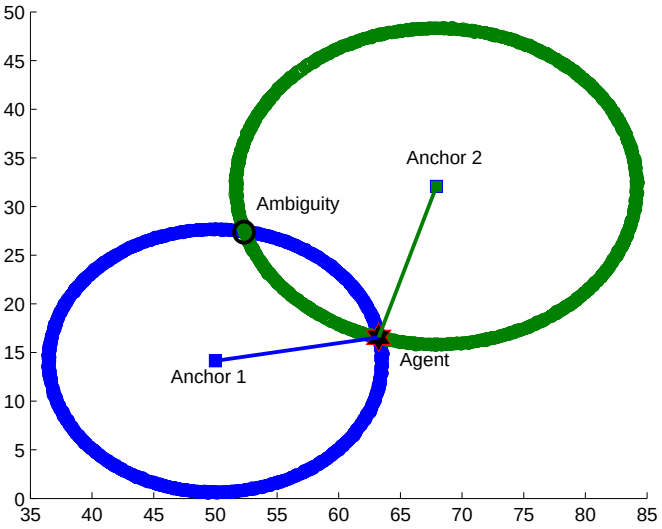


Figure 3.6: Mixture of two Gaussian distribution.

3.3 Results and Discussion

3.3.1 Cooperative Least Square

We have developed the cooperative least square (LS) positioning algorithm by using Equations 3.1-3.2. To simulate the results in a 100 m$\times$100 m map, we take 100 agents cooperating each other with in 20 m range and 13 systematically placed anchors. The results are shown in Figure 3.7, where the blue stars are the positions of the anchors and the green squares represent the actual position of the agents. The red dots are the estimated positions of the agents at different iterations. If you follow a trail of red dots, you can have an idea how the cooperative LS update its estimate. Estimates of 50 iterations have been plotted in Figure 3.7. The blue lines show the error between actual positions and corresponding final estimated positions (after 50 iterations) of the agents.

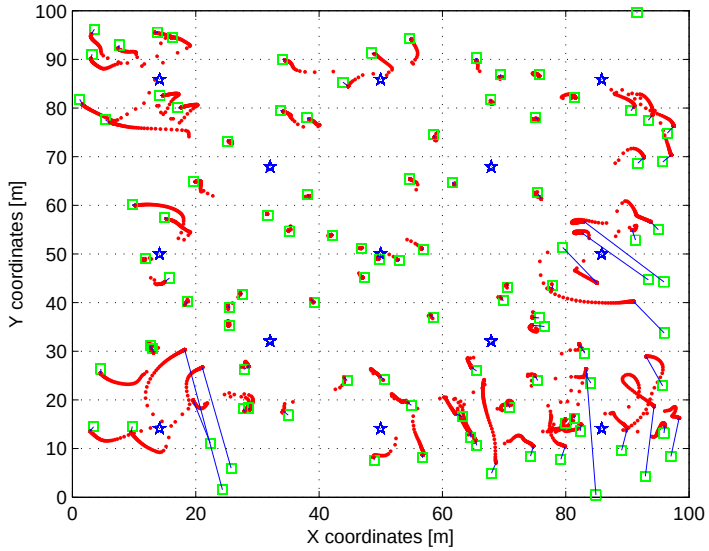


Figure 3.7: Cooperative Least Square positioning updates.

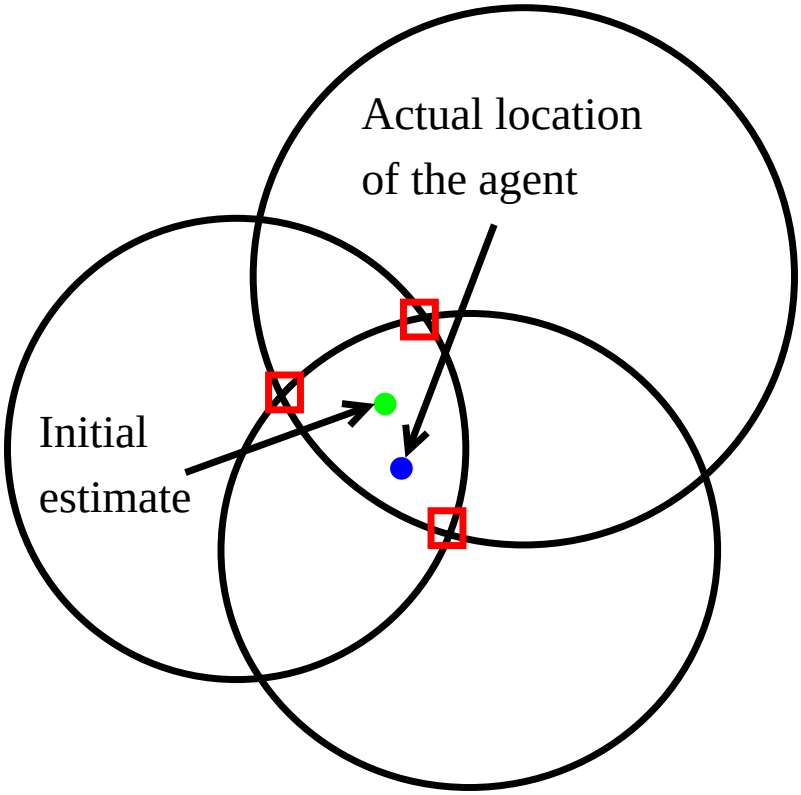


Figure 3.8: Initial estimate calculation in practical trilateration.

To initialize the cooperative LS algorithm we take several reasonable actions. The agents those can receive information from 3 or more anchors have good initial estimates. In error free ranging environment this will be the same as trilateration but in practice all the ranging estimates have errors. So we get 3 intersection points like in Figure 3.8 instead of 1 point considering 3 anchors. In this case our initial estimate will be the center of gravity of the

triangle generated by the 3 intersection points.[1] The agents which can not find at least 3 anchors will not initialize in the first iteration. These agents may get some information from their neighboring agents after some iterations. These agents will wait up to 20 iterations without initializing in a hope to get such information. When the number of reference points is enough these agents will initialize using the previous method. The agents which can not initialize after 20 iterations are in a condition that even cooperation can not help them to have good estimates. The agents those have information from two reference points (in this point both of these may be anchors or agents; or the combination of anchor and agent) have two intersection points. We will take the mid-point of these two points as the initial estimate of the agents. The position of the neighboring reference point will be the initial estimate of the agents who have only one neighbor after 20 iterations.

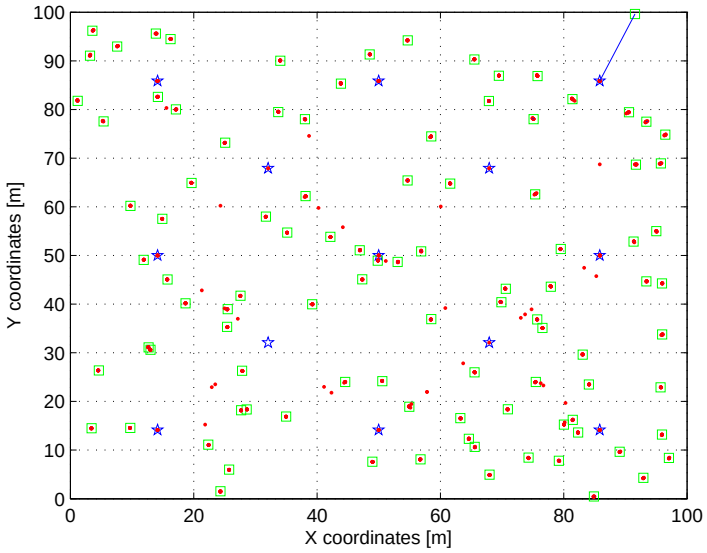


Figure 3.9: SPAWN positioning updates.

3.3.2 Sum Product Algorithm over a Wireless Network (SPAWN)

In Figure 3.9, the simulation results from same network as cooperative LS has been presented. The same legends like Figure 3.7 are also applicable here. We show the estimates for 10 iterations here. It is very interesting to see that in this typical network only 1 agent could not satisfactory converged due to it very bad geometrical placement. The performance of this algorithm is much better than cooperative LS although the complexity of the message multiplication phase is very high. To get an idea about the complexity of the SPAWN we can compare the simulation time of SPAWN over cooperative LS. We observe that a typical network with SPAWN takes about 5 hours to converge (10 iterations), while cooperative LS only takes 15 minutes

[1] Although there are 6 intersection points in Figure 3.8, we are interested on those points which are inside the three circles. These are marked with red square boxes.

to converge (50 iterations) in a same computing platform. This high computation complexity makes SPAWN challenging for practical implementation.

As the agents share full statistical positional information, we don't have to determine initial estimate as cooperative LS. If we know the ranging noise variance we can represent the initial distribution very well even if some agents have ambiguity (those can be represented by a mixture of two Gaussian distribution).

Chapter 4

Censoring

In a dense network, agents may receive information from many neighbors. Not all of those links are useful and by censoring the bad links, we may achieve reduced complexity and traffic, at little or no performance loss.

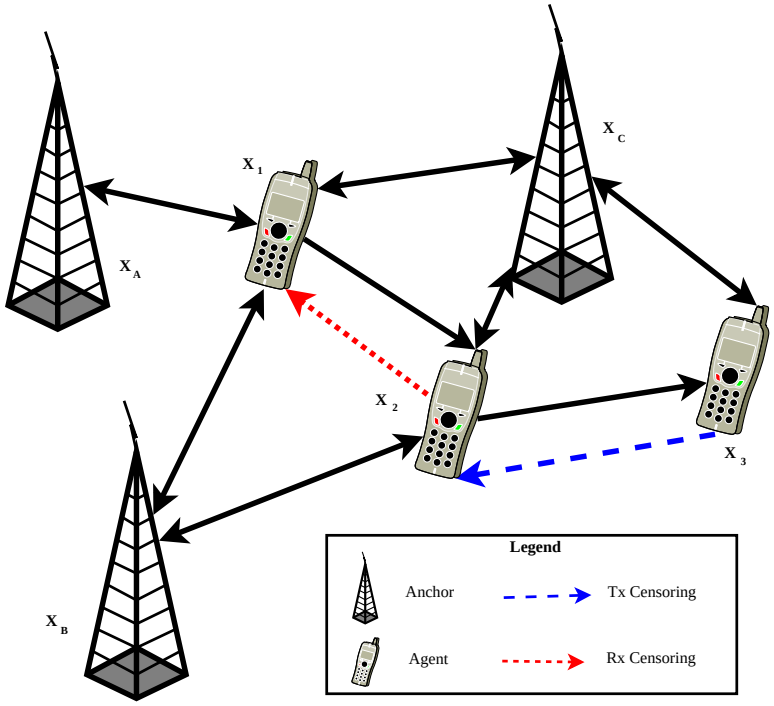


Figure 4.1: Transmit and receive censoring schemes in a cooperative network, with 3 agents (X_1, X_2, X_3) and 3 anchors (X_A, X_B, X_C).

In Figure 4.1, two censoring schemes are shown. The three agents have connectivity between them and with fixed anchors. Agent 3 is connected to only one anchor, so initially it has limited knowledge about its position. Hence, this agent cannot help its neighbor nodes to be localized and should not broadcast its positional information. We define this as *transmit censoring*. Note that agent 2 is connected to two anchors which gives it an ambiguity but this information may be useful for other agents. So agent 2 should broadcast its positional informa-

tion. Agent 1 can get information from three anchors and also from agent 2. By discarding the information from agent 2, its positioning accuracy may be almost unaffected. We define this as *receive censoring*.

4.1 Censoring: Criterion

Clearly, censoring requires a criterion based on which agents decide whether or not to censor. Different measures such as entropy or the Cramér-Rao bound (CRB) can be used to quantify the uncertainty. The calculation of entropy has almost same or more computational complexity as the positioning algorithms. So the use of entropy is not meaningful in our case. We propose to use the Cramér-Rao bound (CRB), because of its rigorous foundation, and its wide applicability to cooperative positioning algorithms. The Cramér-Rao bound (CRB) of any agents estimate is also dependent on geometrical configuration of its neighbors estimates.

4.1.1 Entropy

Entropy is a measure of the uncertainty associated with a random variable. If $\mathbf{x}_i$ is the position of agent i, the entropy can be calculated as

$$h_i = -\int_{-\infty}^{\infty} \Pr(\mathbf{x}_i) \log(\Pr(\mathbf{x}_i))\, d\mathbf{x}_i. \tag{4.1}$$

The multiplication of the pdf of $\mathbf{x}_i$ and log-likelihood function makes the computation of entropy tedious.

4.1.2 Cramér-Rao bound

The Cramér-Rao bound (CRB) is a lower bound on the performance of any unbiased estimator. This bound can act as a guideline about the performance of any estimator in comparison to the best possible estimator. It is calculated by taking inverse of the Fisher Information matrix (FIM) [13, 14]. Considering the position $\mathbf{x}_i$ of agent i, then the FIM is given by

$$\mathbf{F}(\mathbf{x}_i) = -\mathbb{E}_{n_{j\to i}} \left\{ \frac{\partial^2 \Lambda(\mathbf{x}_i)}{\partial \mathbf{x}_i^2} \right\}, \tag{4.2}$$

where

$$\Lambda(\mathbf{x}_i) = \log \left[\Pr\left(\hat{\mathbf{d}} \mid \mathbf{x}_i, \left\{ \mathbf{x}_j \right\}_{j\in S_{\to i}} \right) \right] \tag{4.3}$$

is the log-likelihood function, and the expectation occurs over the ranging noise. Here $\mathbf{x}_i$ is deterministic unknown and $\left\{ \mathbf{x}_j \right\}_{j\in S_{\to i}}$ are the neighbors' position of agent j, which are assumed to be known. It can be shown that the FIM of $\mathbf{x}_i$ will be of the form [15]

$$\mathbf{F}(\mathbf{x}_i) = \sum_{j \in S_{\to i}} \frac{1}{\sigma^2_{j \to i}} \mathbf{q}_{ij} \mathbf{q}_{ij}^T, \tag{4.4}$$

where

$$\mathbf{q}_{ij} = \frac{\mathbf{x}_i - \mathbf{x}_j}{\left\| \mathbf{x}_i - \mathbf{x}_j \right\|}.$$

Finally, the CRB can be calculated as

$$\mathrm{CRB}(\mathbf{x}_i) = \mathrm{trace}\left(\left[\mathbf{F}(\mathbf{x}_i) \right]^{-1} \right). \tag{4.5}$$

4.1.2.1 Cramér-Rao bound with nuisance parameters

In estimation theory we often have some parameters beyond our interest, these parameters are know as nuisance parameters. The calculation of Cramér-Rao bound (CRB) in presence of nuisance parameters can be found by

$$\mathbf{NF}(\mathbf{x}_i) = -\mathbb{E}_{n_{j \to i}} \left\{ \frac{\partial^2 \chi(\mathbf{x}_i)}{\partial \mathbf{x}_i^2} \right\}, \tag{4.6}$$

where

$$\chi(\mathbf{x}_i) = \log \left[\Pr\left(\hat{\mathbf{d}} \mid \mathbf{x}_i \right) \right] \tag{4.7}$$

and

$$\Pr\left(\hat{\mathbf{d}} \mid \mathbf{x}_i \right) = \int \Pr\left(\left\{ \mathbf{x}_j \right\}_{j \in S_{\to i}}, \hat{\mathbf{d}} \mid \mathbf{x}_i \right) \mathrm{d} \left\{ \mathbf{x}_j \right\}_{j \in S_{\to i}}, \tag{4.8}$$

here $\mathbf{x}_i$ is a deterministic unknown and $\left\{ \mathbf{x}_j \right\}_{j \in S_{\to i}}$ are random variables. The integration to determine the marginal distribution $\Pr\left(\hat{\mathbf{d}} \mid \mathbf{x}_i \right)$ makes the calculation of Cramér-Rao bound (CRB) quite tedious. In this case calculation of Modified Cramér-Rao bound (MCRB) is much easier. Suppose we know the prior distributions of $\left\{ \mathbf{x}_j \right\}_{j \in S_{\to i}}$s (neighbors of i), then the Modified Fisher Information matrix (MFIM) [16] can be calculated as

$$\mathbf{MF}(\mathbf{x}_i) = -\mathbb{E}_{\left\{ \mathbf{x}_j \right\}_{j \in S_{\to i}}} \left[\mathbb{E}_{n_{j \to i}} \left\{ \frac{\partial^2 \Lambda(\mathbf{x}_i)}{\partial \mathbf{x}_i^2} \right\} \right], \tag{4.9}$$

here the expectation occurs over the ranging noise and random variables $\left\{ \mathbf{x}_j \right\}_{j \in S_{\to i}}$. Note that $\Lambda(\mathbf{x}_i)$ was introduced in (4.3). This can be further decomposed as

$$\mathbf{MF}(\mathbf{x}_i) = \mathbb{E}_{\left\{ \mathbf{x}_j \right\}_{j \in S_{\to i}}} \left[\sum_{j \in S_{\to i}} \frac{1}{\sigma^2_{j \to i}} \mathbf{q}_{ij} \mathbf{q}_{ij}^T \right], \tag{4.10}$$

and the MCRB can be found by

$$\mathrm{MCRB}(\mathbf{x}_i) = \mathrm{trace}\left(\left[\mathbf{MF}(\mathbf{x}_i) \right]^{-1} \right). \tag{4.11}$$

4.1.2.2 Bayesian Cramér-Rao bound with nuisance parameters

The ordinary CRB is not valid to compare the performance of a Bayesian estimator. The performance of any estimator in Bayesian scenario can be bounded by, Bayesian Cramér-Rao bound (BCRB). The Bayesian Fisher Information Matrix(BFIM) [17]can be achieved by using following equation

$$\mathbf{BF}_i = -\mathbb{E}_{n_{j\to i},\mathbf{x}_i}\left\{\frac{\partial^2 \chi(\mathbf{x}_i)}{\partial \mathbf{x}_i^2}\right\} - \mathbb{E}_{\mathbf{x}_i}\left\{\frac{\partial^2 \log\left[p(\mathbf{x}_i)\right]}{\partial \mathbf{x}_i^2}\right\}, \tag{4.12}$$

where

$$\chi(\mathbf{x}_i) = \log\left[\Pr\left(\hat{\mathbf{d}} \mid \mathbf{x}_i\right)\right],$$

and $p(\mathbf{x}_i)$ is the prior distribution of agent i. $\Pr\left(\hat{\mathbf{d}} \mid \mathbf{x}_i\right)$ is the marginal distribution, which is introduced in (4.8), is very hard to calculate. In this case both $\mathbf{x}_i$ and $\left\{\mathbf{x}_j\right\}_{j\in S_{\to i}}$ are random variables.

In presence of nuisance parameters the modified Bayesian Cramér-Rao bound (MBCRB) [16, 18] is easier to calculate than BCRB. The modified Bayesian Fisher Information matrix (MFIM) can be obtained by

$$\mathbf{MBF}_i = -\mathbb{E}_{n_{j\to i},\mathbf{x}_i,\left\{\mathbf{x}_j\right\}_{j\in S_{\to i}}}\left\{\frac{\partial^2 \Lambda(\mathbf{x}_i)}{\partial \mathbf{x}_i^2}\right\} - \mathbb{E}_{\mathbf{x}_i}\left\{\frac{\partial^2 \log\left[p(\mathbf{x}_i)\right]}{\partial \mathbf{x}_i^2}\right\}, \tag{4.13}$$

here the expectation occurs over the ranging noise and random variables $\mathbf{x}_i$ and $\left\{\mathbf{x}_j\right\}_{j\in S_{\to i}}$.This can also be written as

$$\mathbf{MBF}_i = \mathbb{E}_{\mathbf{x}_i}\left[\mathbb{E}_{\left\{\mathbf{x}_j\right\}_{j\in S_{\to i}}}\left\{\mathbf{F}(\mathbf{x}_i)\right\}\right] - \mathbb{E}_{\mathbf{x}_i}\left\{\frac{\partial^2 \log\left[p(\mathbf{x}_i)\right]}{\partial \mathbf{x}_i^2}\right\},$$

where $p(\mathbf{x}_i)$ is the prior distribution of $\mathbf{x}_i$. We approximate this distribution as a Gaussian and

$$p(\mathbf{x}_i) = \frac{1}{2\pi\sigma_i^2}\exp\left(-\frac{1}{2\sigma_i^2}\left(\mathbf{x}_i - \mathbf{a}_i\right)^2\right),$$

where σ_i^2 is the variance of the distribution and $\mathbf{a}_i$ is the mean, so that $\mathbf{MBF}_i$ becomes

$$\mathbf{MBF}_i = \mathbb{E}_{\mathbf{x}_i}\left[\mathbb{E}_{\left\{\mathbf{x}_j\right\}_{j\in S_{\to i}}}\left[\sum_{j\in S_{\to i}}\frac{1}{\sigma_{j\to i}^2}\mathbf{q}_{ij}\mathbf{q}_{ij}^T\right]\right] + \frac{1}{\sigma_i^2}. \tag{4.14}$$

Finally, the modified Bayesian Cramér-Rao bound (MBCRB) can be calculated as,

$$\mathrm{MBCRB}_i = \mathrm{trace}\left(\left[\mathbf{MBF}_i\right]^{-1}\right). \tag{4.15}$$

4.1.3 Reason of choosing Cramér-Rao bound as a censoring criterion

To get an intuition about the reason of using Cramér-Rao bound, we can consider Figure 4.2, where an agent X is trying to select the best 3 neighbors from a set of 5 neighbors for positioning. Node 1, 2, 3 can be the best ones if agent X will consider the links individually. But in a

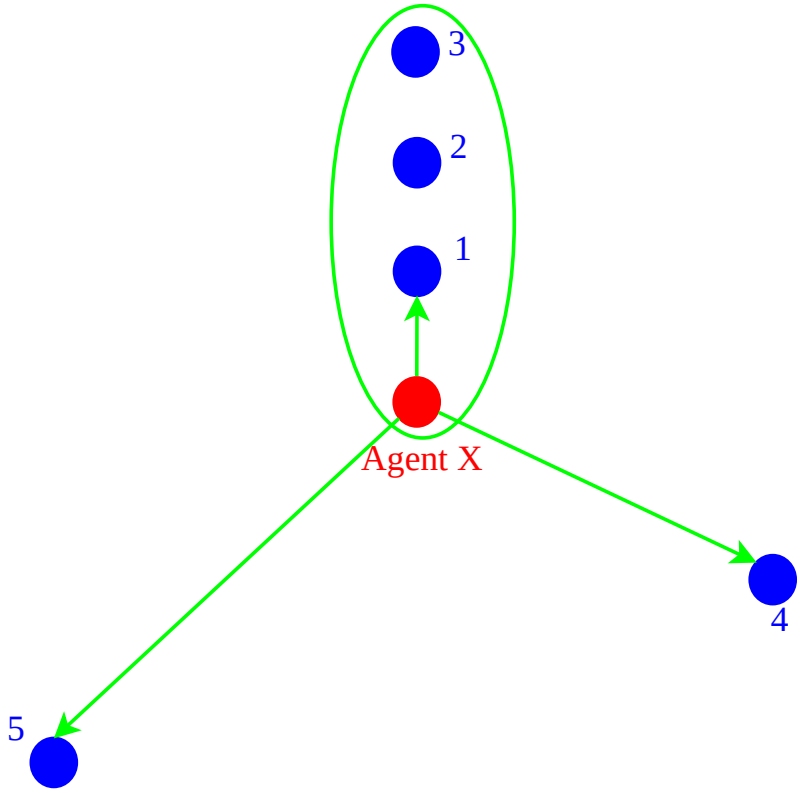


Figure 4.2: Reason of using Cramér-Rao bound as a censoring criterion.

group node 2 and 3 can not provide extra information than that is provided by node 1 to agent X because all three nodes give information in one direction. As we already discuss the positioning performance also dependent on the geometry of an agent and its neighbors. Cramér-Rao bound cares about this geometry and it can select the best possible combination. In Figure 4.2, if we employ Cramér-Rao bound to select the best 3 neighbors it will select node 1, 4, 5, the connections with agent X are shown with green arrows.

4.2 Censoring: Overview

Here we present the high level presentation of transmit and receive censoring schemes in the form of a tutorial. Let us consider a network with some anchors and agents. In Figure 4.3(a) the towers represents the position of anchors and the phones represents the agents. The agents can have range estimate with the anchors in range. The connections between agents and corresponding anchors have been shown in Figure 4.3(b). The agents which get information from enough number of anchors have good initial estimate and they will broadcast their positional

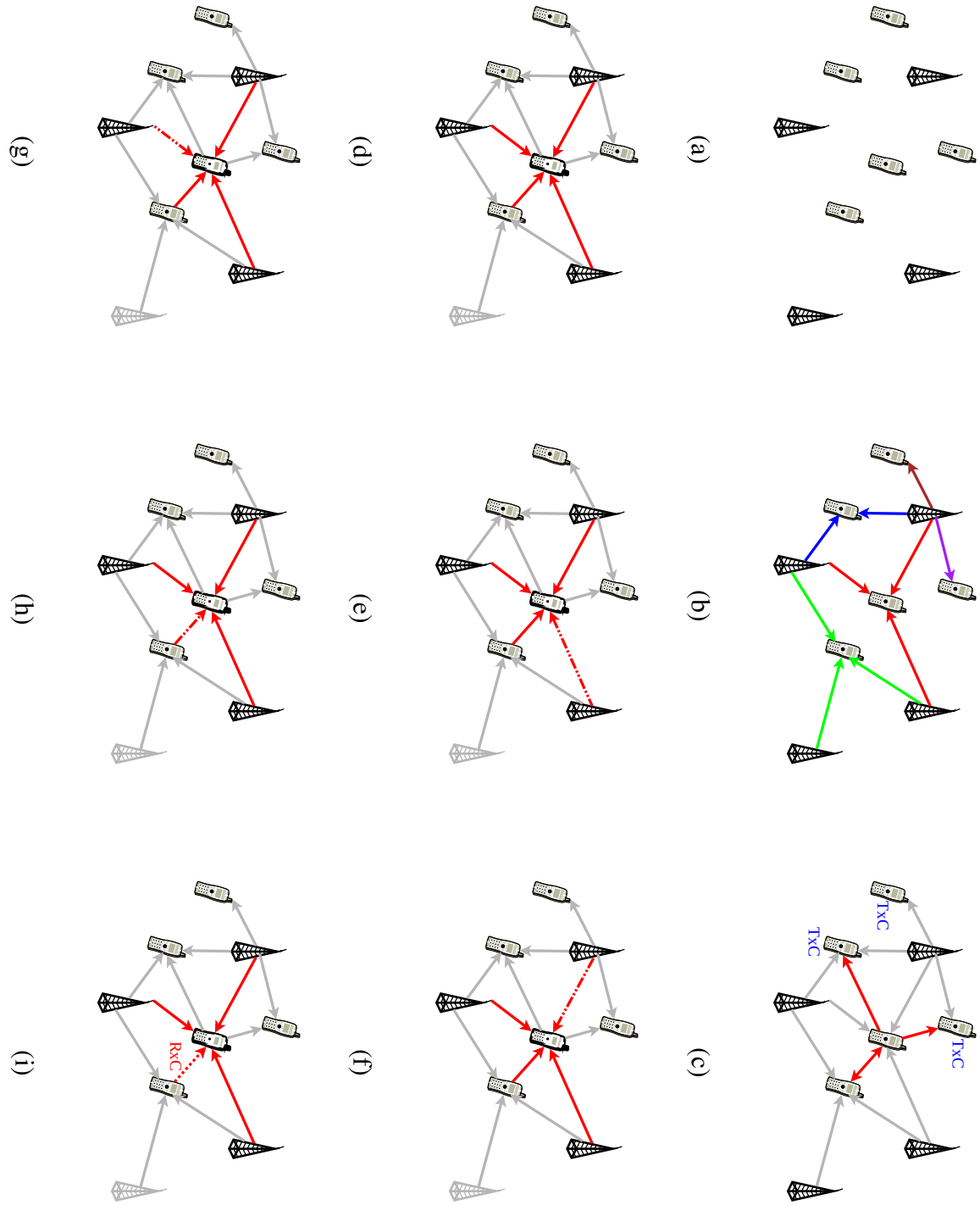


Figure 4.3: Censoring overview.

information to their neighbors. On the other hand, the agents which are in a bad geometrical position to see enough anchors can not have good initial estimate and they should not broadcast their position estimates to their corresponding neighbors. From our previous discussions we propose to use Cramér-Rao bound to differentiate these two types of agents and block the broadcast the information from the so-called *unreliable agents*, we call this transmit censoring (shown in Figure 4.3(c)). Now some of the agents can get information from some good neighboring agents as well as from the anchors in range (Figure 4.3(d)). Not all of these information sources may be required to localize precisely. Again Cramér-Rao bound can be used to remove the *uninformative* links or to select *the best set* of links for the next iterative update. To do this at first an agent will compute its Cramér-Rao bound using all the available links and store it internally. Then it will recalculate the Cramér-Rao bound after removing an agent at a time and by comparing it with previously stored value the agent can determine the worst link. These activities can be visualized in Figures 4.3(e)-(h). After determining the worst link we can remove the so-called *uninformative* link, which we define as receive censoring (Figure 4.3(i)). This link discarding will continue until reaching a certain receive censoring threshold and the process will be occur in a distributive manner.

4.3 Censoring for Cooperative Least Square Algorithm

4.3.1 Transmit Censoring

In transmit censoring Algorithm 4.1, an agent will decide to broadcast or censor its positional information based on its CRB. Since the true position of the agent, nor of its neighbors, is known, the CRB will be calculated by using the *estimated* positions. Hence, an agent will transmit-censor when

$$\mathrm{CRB}\left(\hat{\mathbf{x}}_i\right) = \mathrm{trace}\left(\left[\tilde{\mathbf{F}}\left(\hat{\mathbf{x}}_i\right)\right]^{-1}\right) > \gamma_{\mathrm{TX}}, \tag{4.16}$$

where

$$\tilde{\mathbf{F}}\left(\hat{\mathbf{x}}_i\right) = \sum_{j \in S_{\rightarrow i}} \frac{1}{\sigma^2_{j\rightarrow i}} \tilde{\mathbf{q}}_{ij} \tilde{\mathbf{q}}_{ij}^T, \tag{4.17}$$

in which

$$\tilde{\mathbf{q}}_{ij} = \frac{\left(\hat{\mathbf{x}}_i - \hat{\mathbf{x}}_j\right)}{\left\|\hat{\mathbf{x}}_i - \hat{\mathbf{x}}_j\right\|},$$

and γ_{TX} is a threshold. The value of this threshold depends on the ranging model and the performance requirements. Initially, the set of neighbors $S_{\rightarrow i}$ contains very few elements (e.g., anchors within range). After some iterations, the number of elements in the set of neighbors $S_{\rightarrow i}$ will increase. This helps the agent to attain lower CRB and meet the bound of sharing

information, at which point the agent will start broadcasting.

If we lower the transmit censoring threshold γ_{TX}, more agents will block their broadcast. This will reduce the network traffic but we may loose the positioning performance. On the other hand, increasing the value of γ_{TX}results in high traffic.

4.3.2 Receive Censoring

In receive censoring Algorithm 4.2, an agent will receive positional information from its neighbors and it can calculate its present Cramér Rao bound, $\text{CRB}(\hat{\mathbf{x}}_i)$. The agent will then remove links, as long as $\text{CRB}(\hat{\mathbf{x}}_i) < \gamma_{\text{RX}}$. Here, the threshold γ_{RX} again depends on the model and the desired performance. During the worst link selection the agent will calculate

$$\text{CRB}_k(\hat{\mathbf{x}}_i) = \text{trace}\left(\left[\tilde{\mathbf{F}}_k(\hat{\mathbf{x}}_i)\right]^{-1}\right), \tag{4.18}$$

where

$$\tilde{\mathbf{F}}_k(\hat{\mathbf{x}}_i) = \sum_{j \in S^k_{\to i}} \frac{1}{\sigma^2_{j\to i}} \tilde{\mathbf{q}}_{ij} \tilde{\mathbf{q}}^T_{ij}. \tag{4.19}$$

Algorithm 4.1 Transmit censoring for cooperative LS positioning (at an arbitrary iteration for an agent)

```
if i is an agent then
    calculate CRB(x̂_i)
    if CRB(x̂_i) < γ_TX then
        broadcast current positional information
    end if
end if
```

4.3.3 Combination of Transmit and Receive Censoring

We can merge both transmit and receive censoring schemes as shown in Algorithm 4.1 and Algorithm 4.2 respectively. This combined algorithm can remove harmful links and also select the best links for update.

4.4 Censoring for SPAWN

4.4.1 Transmit Censoring

In transmit censoring Algorithm 4.3, an agent will decide to broadcast or censor its positional information based on the uncertainty of its belief. After calculating the belief of any agent at a certain iteration we can know the standard deviation of the belief, i.e., how tight the belief is. Hence, an agent will transmit-censor when

Algorithm 4.2 Receive censoring for cooperative LS positioning (at an arbitrary iteration for an agent)

while $\mathrm{CRB}(\hat{\mathbf{x}}_i) < \gamma_{\mathrm{RX}}$ **do**
  while $|S_{\to i}| > 3$ **do**
    for $k = 1, \cdots, |S_{\to i}|$ **do**
      make the new set of neighbors $S^k_{\to i}$ by removing the kth element from $S_{\to i}$
      calculate $\mathrm{CRB}_k(\hat{\mathbf{x}}_i)$ using (4.18)
    end for
    determine the worst link,

$$\hat{k} = \arg\min_k \mathrm{CRB}_k(\hat{\mathbf{x}}_i)$$

    if $\mathrm{CRB}_{\hat{k}}(\hat{\mathbf{x}}_i) < \gamma_{\mathrm{RX}}$ **then**
      remove link $\hat{k}$ from $S_{\to i}$
    else
      break
    end if
  end while
end while
use the current set of neighbors $S_{\to i}$ for update

$$2(\rho + \sigma)^2 > \gamma_{TX}, \tag{4.20}$$

which is the MBCRB as in (4.15) where there is no measurements, i.e., (4.13) has only the second term in the right hand side. This is also comparable to transmit censoring schemes [9]. γ_{TX} is a threshold. Here the Fisher Information Matrix is an measure of uncertainty of the belief, we can write it as,

$$\tilde{\mathbf{F}}(\hat{\mathbf{x}}_i) = \begin{matrix} \frac{1}{(\rho+\sigma)^2} & 0 \\ 0 & \frac{1}{(\rho+\sigma)^2} \end{matrix}. \tag{4.21}$$

The value of this threshold depends on the ranging model and the performance requirements. During first iteration (non-cooperative phase) the set of neighbors $S_{\to i}$ contains very few elements (e.g., anchors within range). As a result most of the beliefs have high standard deviation. From the second iteration, the number of elements in the set of neighbors $S_{\to i}$ will increase as the agents start sharing their positional information. This helps the agents to lower the uncertainty of their beliefs and meet the bound of sharing information, at which point the agents will start broadcasting.

Algorithm 4.3 Transmit censoring for SPAWN (at an arbitrary iteration for an agent).

check $(\rho + \sigma)$ of belief of agent i {only for agents}
if $2(\rho + \sigma)^2 < \gamma_{TX}$ **then**
  broadcast current positional information
end if

4.4.2 Receive Censoring

In receive censoring, an agent will receive positional information from its neighbors. At first the agent will check the standard deviation, $(\rho + \sigma)$ of its belief of the agent in previous iteration which is locally stored in the device. If $2(\rho + \sigma)^2 < \gamma_{RX}$, it will check the number of components in the belief. If the belief has one components it will censor all the incoming links, i.e., it will stop updating; else it will try to wipe off the ambiguity. As Cramér Rao bound does not account for ambiguity, so we need to deal with that separately using the steps mentioned in section 4.4.2.1. Again if $2(\rho + \sigma)^2 > \gamma_{RX}$, it will check the number of components. If the belief has one components it will follow the steps mentioned in link selection section 4.4.2.2. In case of $2(\rho + \sigma)^2 > \gamma_{RX}$ and outgoing message has two components the agent will at first try to remove its ambiguity using 4.4.2.1 and then choose the set of links using 4.4.2.2.

Algorithm 4.4 Receive censoring for SPAWN (at an arbitrary iteration for an agent).

1: receive positional information from neighbors, $j \in S_{\rightarrow i}$
2: check the number of components in outgoing message of agent i from the previous iteration
3: **if** $number\,of\,components = 1$ **then**
4: check $(\rho + \sigma)$ in outgoing message of agent i from the previous iteration {only for agents}
5: **if** $2(\rho + \sigma)^2 < \gamma_{RX}$ **then**
6: remove all neighbors from $S_{\rightarrow i}$
7: **else**
8: select best 3 neighbors from $S_{\rightarrow i}$ (using Section 4.4.2.2)
9: **end if**
10: **else**
11: try to remove ambiguity in outgoing message of agent i locally (using Section 4.4.2.1)
12: go back to line 4
13: **end if**
14: use the current set of neighbors $S_{\rightarrow i}$ for update

4.4.2.1 Ambiguity removal algorithm

1. Organize the elements in $S_{\rightarrow i}$ accenting order according to corresponding $(\rho + \sigma)$.
2. Agent i will try to remove its ambiguity geometrically (if possible) with the help of agent j where $j \in S_{\rightarrow i}$. For this purpose j having one component in its belief will be prioritize.
3. As soon as one neighbor agent j has been found which can remove the ambiguity of i, the algorithm stops and select the correct component of i internally.

4.4.2.2 Link selection algorithm

In link selection we restrict the maximum allowable neighbors to three. The link selection algorithm will only start if the number of incoming links is greater than three. It will select the

best three neighbors based on modified Bayesian Cramér Rao bound, MBCRB_i using following steps:

1. Let $S_{\to i}^{(k,m,n)}$ be the set of neighbors obtained by taking the kth, mth and nth element from $S_{\to i}$, where $k > m > n$. Calculate

$$\text{MBCRB}_i^{(k,m,n)} = \text{trace}\left(\left[\tilde{\mathbf{MBF}}_i^{(k,m,n)}\right]^{-1}\right), \tag{4.22}$$

where

$$\tilde{\mathbf{MBF}}_i^{(k,m,n)} = \mathbb{E}_{\hat{\mathbf{x}}_i}\left[\mathbb{E}_{\left\{\hat{\mathbf{x}}_j\right\}_{j\in S_{\to i}}}\left[\sum_{j\in S_{\to i}^{k,m,n}} \frac{1}{\sigma_{j\to i}^2}\tilde{\mathbf{q}}_{ij}\tilde{\mathbf{q}}_{ij}^T\right]\right] + \frac{1}{\sigma_i^2}. \tag{4.23}$$

Here $\text{MBCRB}_i^{(k,m,n)}$ is the modified Bayesian Cramér Rao bound of agent i using the kth, mth and nth element from its neighbors set $S_{\to i}$. Similar explanation is also valid for $\tilde{\mathbf{MBF}}_i^{(k,m,n)}$.

2. Select the best set of links:

$$\left[\hat{k},\hat{m},\hat{n}\right] = \arg\min_{k,m,n} \text{MBCRB}_i^{(k,m,n)}. \tag{4.24}$$

3. Set $S_{\to i}$ to $S_{\to i}^{\left(\hat{k},\hat{m},\hat{n}\right)}$.

4.4.3 Combination of Transmit and Receive Censoring

We can merge both transmit and receive censoring schemes as shown in Algorithms 4.3 and 4.4. This combined scheme can remove harmful links and also select the best links for update.

Chapter 5

Numerical Results

5.1 Simulation Setup

In our simulation, we consider a wireless sensor network with 100 randomly placed agents with 20 m communication limit, and 13 anchors placed in a organized way in a 100 m $\times$ 100 m map (see Figure). The standard deviation of range measurement noise is 10 cm (standard for indoor UWB measurements) [3].

5.2 Results for Cooperative Least Squares Algorithm

5.2.1 Simulation Parameters

We fix the value of $\delta_i^{(l)} = 0.075 \, \forall i, l$ as it gives a good positional accuracy and convergence trade-off. The second term in the right hand side of (3.1) is the *correction* from all of the neighbors. When the correction falls below a threshold (depending on the ranging model and quality requirements) we can stop updating the positional information of that agent. We call the threshold *stoplimit*. We first set several parameters: the stoplimit, γ_{TX} and γ_{RX}. When we relax our expectation of positioning accuracy (i.e., increase the stoplimit) most of the agents will converge after few iterations. On the other hand, if we tighten our accuracy requirement (i.e., reduce the stoplimit), cooperative LS may need more iterations to converge. For rest of the simulations we fix the stoplimit such that agents converge in less than 100 iterations, leading to a stoplimit 0.015 m.

We now fix the censoring thresholds γ_{TX} and γ_{RX}. Our goal is to maintain a performance similar to normal cooperative LS, with reduced packet exchanges and complexity. Censoring can be conservative (i.e., less censoring) or aggressive (i.e., more censoring). The smaller γ_{TX}, the more aggressive we perform transmit censoring. Similarly, the larger γ_{RX}, the more aggressive we perform receive censoring. We set the thresholds based on the percentiles of

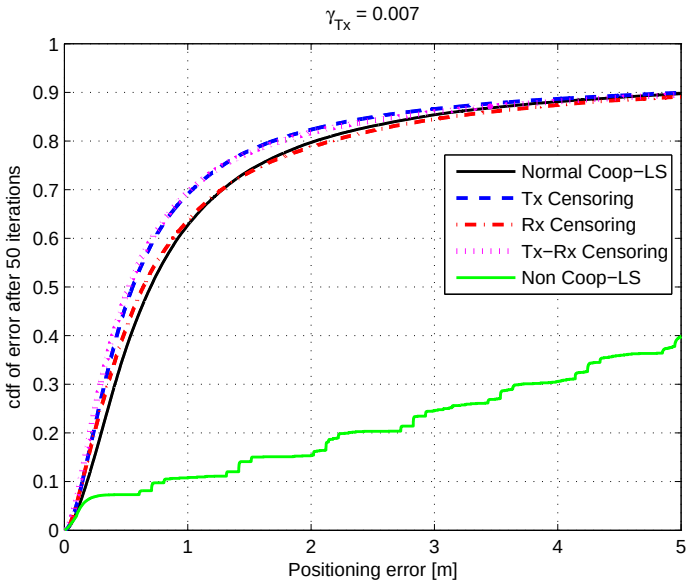


Figure 5.1: Performance comparison of different censoring schemes in conservative approach.

the expected CRB.[1] We consider aggressive receive censoring by setting $\gamma_{\text{RX}} = (8\text{cm})^2$, which is the 90th percentile of the CRB. This means that roughly 90% of the agents will perform receive censoring. For transmit censoring, we consider two types of censoring: conservative and aggressive, corresponding to the 60th and 90th percentile of the CRB, respectively. This leads to, respectively, $\gamma_{\text{TX}} = (8\text{cm})^2$ (conservative) and $\gamma_{\text{TX}} = (6\,\text{cm})^2$ (aggressive). This means that roughly 40% of the agents will perform transmit censoring under the aggressive approach.

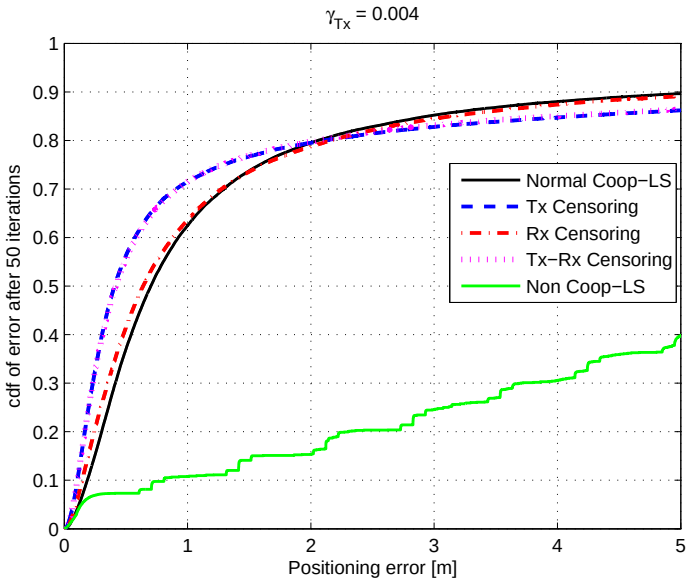


Figure 5.2: Performance comparison of different censoring schemes in aggressive approach.

[1]These percentiles can be determined through test simulations. In practice, the percentiles can be determined from the network topology and the performance requirements.

5.2.2 Simulation Discussion

We now investigate the performance of the different schemes in previously mentioned censoring approaches, showing results after 50 iterations, after which the algorithms have converged most of the time. The cumulative distribution function (cdf) of the positioning error is shown in Figures 5.1-5.2, corresponding to conservative and aggressive transmit censoring, respectively. In addition to the normal cooperative and non-cooperative LS, we show the performance of transmit (Tx) censoring, receive (Rx) censoring, and combined (Tx-Rx) censoring.

From Figure 5.1 (conservative approach), we can distinguish three error regimes: the low error regime (errors less than 1 meter), the medium error regime (errors between 1 and 5 meters), and the high error regime (errors above 5 meters). Note that the high error regime occurs for around 10% of the agents (60% for non-cooperative LS). In the low error regime, we see that all censoring schemes outperform normal cooperative LS. This observation is congruent with our expectation, since the CRB criterion is most meaningful in the low error regime. We observe that transmit censoring (with or without receive censoring) yields the best performance, while receive censoring by itself is the least effective of all the censoring schemes. In the medium error regime, conservative transmit censoring (with or without receive censoring) is beneficial, while receive censoring leads to slightly worse performance compared to normal cooperative LS. Transmit censoring is beneficial in this regime, since agents with poor positional information can censor themselves and not mislead their neighbors. Receive censoring is detrimental since agents with poor position estimates should not discard information from neighbors. The performance of aggressive transmit censoring is shown in Figure 5.2. We observe that this approach can achieve better performance in the low error regime but not in medium error regime. The reason is that too much information is discarded in the medium error regime.

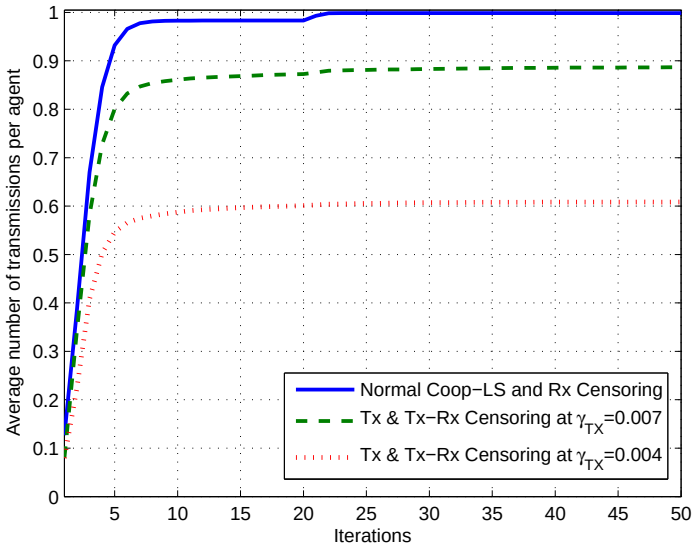


Figure 5.3: Normalized number of packet transmissions as a function of iterations for different censoring schemes.

In addition to the positioning performance, it is also important to evaluate other aspects of censoring algorithms, such as the complexity and number of packets exchanged. Figure 5.3 shows the average number of packets transmitted per node per iteration, as a function of the iteration index. Initially, only agents that can communicate with at least three anchors have an initial estimate, so those are the only ones that will broadcast their position estimates. Hence, the number of packets per agent is very low. As iterations progress, more of the agents acquire estimates through cooperation and will start broadcasting. When transmit censoring is activated, we achieve approximately 10% and 40% reduction in total data traffic with the conservative and aggressive approaches, respectively. These values are directly related to the transmit censoring threshold γ_{TX}. In receive censoring, the number of packets is the same as that of normal cooperative LS.

Table 5.1: Comparison of average links used for update phase.

	Normal Coop-LS	Tx Censoring	Rx Censoring	Tx-Rx Censoring
conservative Tx	11.76	11.14	6.00	5.73
aggressive Tx	11.76	9.91	6.00	5.01

Finally, in Table 5.1, we compare the average number of used links per agent during the update of LS positioning. We observe that less than half of the links are used in receive censoring compared to normal cooperative LS. Even fewer links are used in combined transmit-receive censoring, in particular with aggressive transmit censoring. The overall reduction in used links makes little impact on the computational complexity for LS positioning, but will be important when more sophisticated algorithms are considered such as SPAWN.

5.3 Results for SPAWN

5.3.1 Simulation Parameters

In our simulation we have to set several parameters. The receive censoring threshold γ_{RX} is directly related to positioning accuracy. To achieve high accuracy we have to allow more iterations. On the other hand if we relax our accuracy requirements most of the agents will converge after 5-6 iterations and the overall process will be faster. After the first iteration of the positioning phase (non-cooperative) very few agents (those who have connections with three or more anchors) can achieve low estimation uncertainty and the others have high estimation uncertainty. The transmit censoring threshold γ_{TX} should be chosen such a way that only the agents who have low estimation uncertainty should broadcast their positional information. Both thresholds are dependent on the network geometry. These thresholds can be determined by some test simulations.

For the final results we set $\gamma_{RX} = 0.02$ m and $\gamma_{TX} = 0.03$ m. Note that γ_{TX} should be always greater or equal to γ_{RX} and the positioning performance will be degraded if we set γ_{RX} below or very near to the bound of accuracy. We observe that change in γ_{TX} does not affect in positioning accuracy or average transmission significantly.

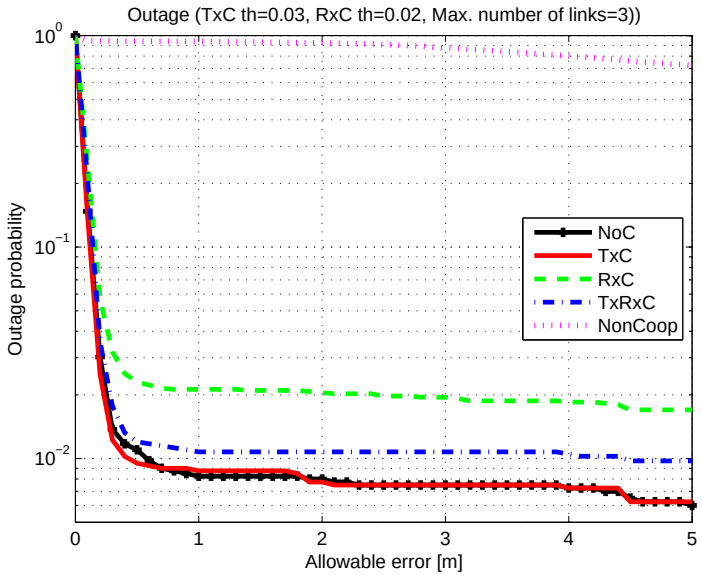


Figure 5.4: Outage probability comparison with and without censoring.

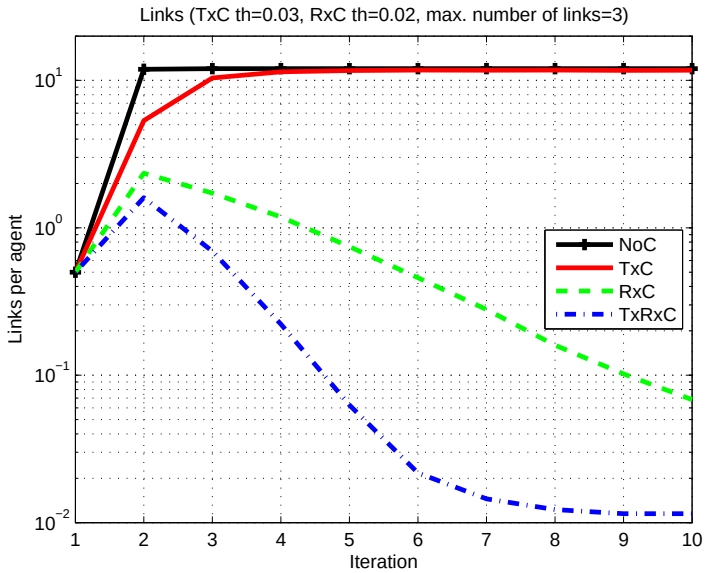


Figure 5.5: Comparison of number of used links.

5.3.2 Simulation Discussion

We now investigate the performance of the different schemes in previously mentioned censoring approaches, showing results after 10 iterations, after which the algorithms have converged

most of the time. To evaluate positioning performance, we consider the *outage probability* criterion [3]. We can easily get the outage probability by plotting 1-(cdf of errors) like in co-operative LS. The outage probability in different censoring are shown in Figure 5.4. It can be

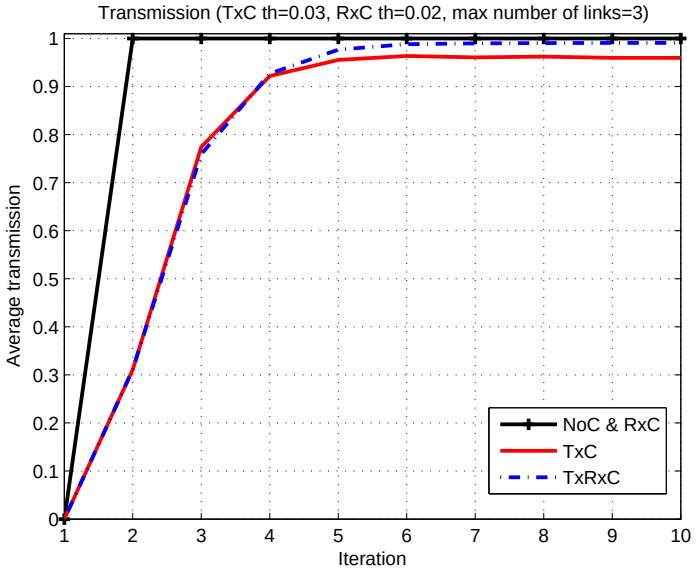


Figure 5.6: Comparison of average transmission.

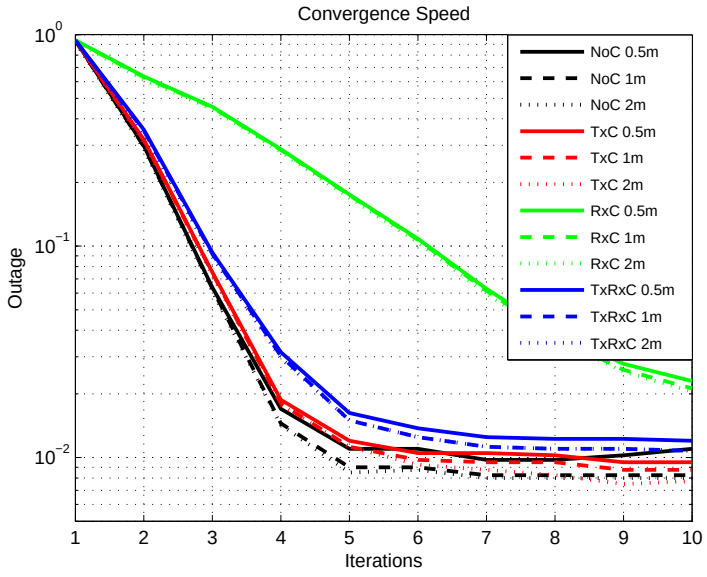


Figure 5.7: Convergence speed of different censoring schemes.

clearly seen from the Figure 5.4 that SPAWN [3] can improve the positioning quality over non-cooperative positioning. We observe that the outage probability curve of transmit censoring scheme follows the curve of SPAWN without any censoring. Receive censoring can not reach the SPAWN without any censoring in quality as we always force the agents to limit the number of links in update phase to 3. We get interesting result when we use transmit-receive combined

censoring scheme. In this case the outage probability curve almost follow the no censoring curve.

Table 5.2: Comparison of simulation time requirements.

	Normal SPAWN	Tx Censoring	Rx Censoring	Tx-Rx Censoring
Simulation time (ratio)	19.5	18.7	1.2	1

The main benefit of censoring schemes is to reduce the computational complexity of message multiplication. By using receive censoring we can select the best links which are really informative. This helps to reduce unnecessary multiplications which make the update phase faster. However, we always loose positioning performance with this scheme. We can see from Figure 5.5 that with receive censoring the average number of used links can be significantly reduced. To give an idea about the complexity reduction capability of SPAWN we would like to mention the simulation time requirement to converge the network (10 iterations) with and without censoring schemes (Table 5.2). We observe that with the transmit-receive combined censoring scheme SPAWN can converge about 20 times faster than previous. Again the transmit censoring blocks the broadcast the positional information of unreliable nodes. This significantly reduces the overall packet transmission in the network. The average transmission in the

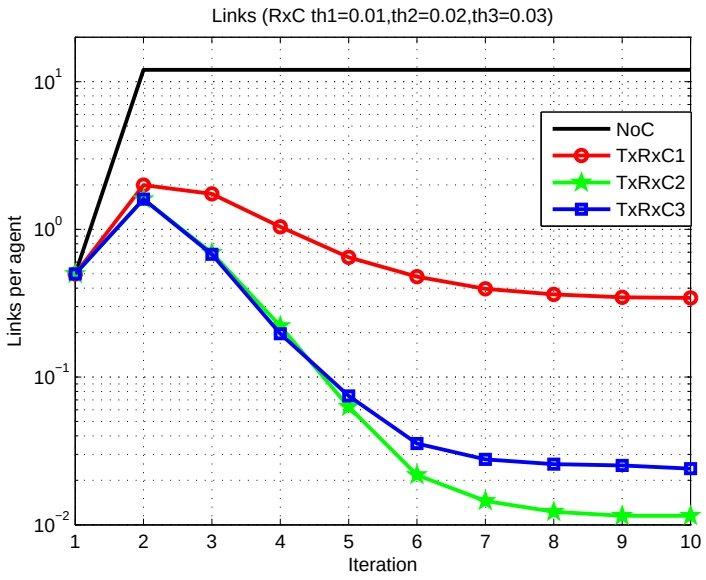


Figure 5.8: Link comparison with different receive censoring thresholds.

network has been shown in Figure 5.6. Here average transmission 1 means 100% nodes broadcasting their positional information in that certain iteration. In the beginning of the iterative algorithm very few nodes have good estimate, as a result the transmission is minimum when transmit censoring has been applied. After few iterations when most of the nodes have been lo-

calized, transmit censoring can not reduce the traffic as the scheme is based on the uncertainty information of the estimates.

From Figure 5.7, we can compare the convergence speed of different censoring schemes in different error expectations over no censoring case. This figure can be used to decide how many iterations should we allow for update. It is clear from the figure that the receive censoring has the worst performance while the transmit censoring and the combined censoring have acceptable speed in comparison to the no censoring case. It is also interesting to observe that after 5-6 iterations the outage vs. iterations curves for transmit censoring and combined censoring become flat. So at that stage we can stop the update of the algorithm. However we have allowed

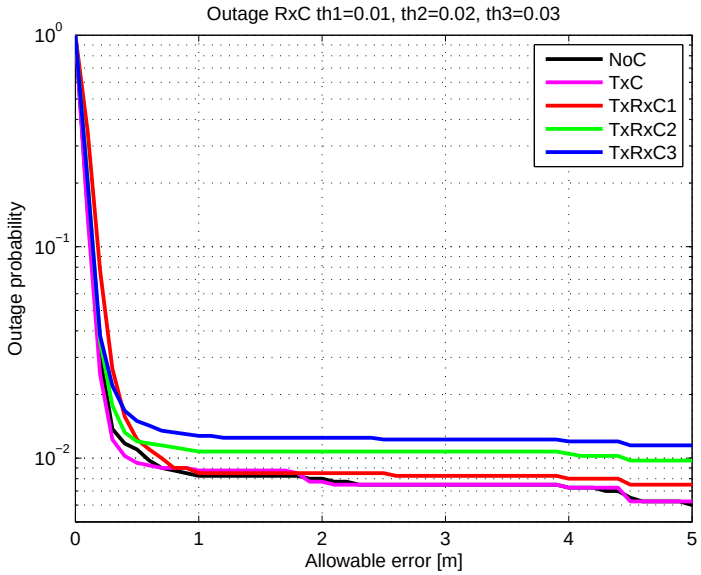


Figure 5.9: Outage comparison for receive censoring with different threshold values.

10 iterations for fair comparison as the receive censoring scheme can not achieve comparable performance before that.

Before setting the final threshold values for transmit and receive we have changed with several values. We tested our algorithm using three threshold values ($\gamma_{TX} = 0.01, 0.02, 0.03$) for transmit censoring. With these three values we do not find significant changes in average network traffic and also in outage. So we select the most aggressive transmit censoring $\gamma_{TX} = 0.03$ and for the combined censoring case we always use this value. In Figure 5.8, the average number of used links for combined censoring schemes with different receive censoring thresholds have been shown. We get the best performance for $\gamma_{RX} = 0.02$ however we get slightly worse performance for $\gamma_{RX} = 0.03$ during iterations 5-10. The reason is with this aggressive receive censoring some nodes will still continue to update after 5-6 iterations. Although from Figure 5.9, it is clear that with $\gamma_{RX} = 0.01$ we can have best outage, finally we select $\gamma_{RX} = 0.02$ as it gives us lowest average links with acceptable outage.

Chapter 6

Conclusions and Future Work

In this thesis we have evaluated different censoring schemes for cooperative positioning in dense networks, in order to reduce complexity, and packet transmissions while still maintaining excellent performance. All censoring decisions are distributed and based on a Cramér Rao bound criterion.

We have applied these censoring schemes to a standard cooperative least squares positioning algorithm, and found that: (i) receive censoring, which was proposed previously, can improve positioning performance, while at the same time considering less information from neighbors; (ii) two new censoring schemes (based on transmit censoring) can improve positioning performance even further, with drastic reduction in network traffic.

We also show that censoring schemes are more meaningful in Bayesian positioning algorithms such as SPAWN. By applying our proposed censoring schemes to SPAWN we have found that: (i) receive censoring can reduce the number of links that are using for the update but at a cost in positioning performance which helps SPAWN to converge about 16 times faster than previous; (ii) transmit censoring (with or without receive censoring) can reduce the network traffic specially in the beginning of the iteration without any significant performance loss.

These advantages of censoring schemes (distributed nature, reduced complexity and network traffic while maintain the positioning performance) make them promising for large-scale dense networks. Future work includes extending the proposed censoring schemes to account for non-line of sight (NLOS) propagation and testbed implementation.

Bibliography

[1] T. Rappaport, J. Reed, and B. Woerner, "Position location using wireless communications on highways of the future," *Communications Magazine, IEEE*, vol. 34, no. 10, pp. 33 –41, oct. 1996.

[2] N. Patwari, J. Ash, S. Kyperountas, A. Hero III, R. Moses, and N. Correal, "Locating the nodes: cooperative localization in wireless sensor networks," *IEEE Signal Process. Mag.*, vol. 22, no. 4, pp. 54 – 69, Jul. 2005.

[3] H. Wymeersch, J. Lien, and M. Z. Win, "Cooperative localization in wireless networks," *Proc. of the IEEE*, vol. 97, no. 2, pp. 427 –450, Feb. 2009.

[4] V. Tam, K. Cheng, and K. Lui, "Using micro-genetic algorithms to improve localization in wireless sensor networks," *Journal of Communications*, vol. 1, no. 4, p. 1, 2006.

[5] D. Lieckfeldt, J. You, and D. Timmermann, "Distributed selection of references for localization in wireless sensor networks," in *Proc. of the 5th Workshop on Positioning, Navigation and Communication (WPNC)*, Mar. 2008, pp. 31 –36.

[6] ——, "An algorithm for distributed beacon selection," in *Proc. of the Sixth Annual IEEE Int. Conf. on Pervasive Computing and Communications (PerCom)*, Mar. 2008, pp. 318 –323.

[7] J. Li, A. Ndili, L. Ward, and S. Buchman, "GPS receiver satellite/antenna selection algorithm for the Stanford gravity probe B relativity mission," in *National Technical Meeting'Vision 2010: Present and Future, Institute of Navigation, San Diego, CA*, 1999, pp. 541–550.

[8] C. Park, "Precise relative navigation using augmented CDGPS," Ph.D. dissertation, Stanford University, Jun. 2001.

[9] K. Das and H. Wymeersch, "Censored cooperative positioning for dense wireless networks," in *IEEE International Workshop on Advances in Positioning and Location-Enabled Communications (in conjunction with PIMRC'10) (APLEC 2010)*, Istanbul, Turkey, 2010.

[10] H. Wymeersch, *Iterative receiver design*. Cambridge University Press, 2007.

[11] H.-A. Loeliger, "An introduction to factor graphs," *Signal Processing Magazine, IEEE*, vol. 21, no. 1, pp. 28 – 41, jan. 2004.

[12] L. Wenjing, "Message representation and updates for cooperative positioning," Master's thesis, Department of Signals and Systems, Chalmers University of Technology, 2010.

[13] H. Van Trees, *Detection, estimation, and modulation theory*. Wiley-Interscience, 2001.

[14] S. Kay, *Fundamentals of statistical signal processing: estimation theory*, A. V. Oppenheim, Ed. Prentice Hall PTR, 1993.

[15] N. Patwari, A. Hero III, M. Perkins, N. Correal, and R. O'Dea, "Relative location estimation in wireless sensor networks," *IEEE Trans. on Signal Process.*, vol. 51, no. 8, pp. 2137 – 2148, Aug. 2003.

[16] M. Moeneclaey, "On the true and the modified cramer-rao bounds for the estimation of a scalar parameter in the presence of nuisance parameters," *IEEE Trans. on Commun.*, vol. 46, no. 11, pp. 1536 –1544, Nov. 1998.

[17] H. Van Trees, K. Bell, and S. Dosso, Eds., *Bayesian Bounds for Parameter Estimation and Nonlinear Filtering/Tracking*. John Wiley & Sons, Inc., 2007.

[18] S. Bay, C. Herzet, J.-M. Brossier, J.-P. Barbot, and B. Geller, "Analytic and asymptotic analysis of Bayesian cramer rao bound for dynamical phase offset estimation," *Signal Processing, IEEE Transactions on*, vol. 56, no. 1, pp. 61 –70, jan. 2008.