2011-01-01

Collaborative And Distributed Algorithms For Localization In Wireless Sensor Networks Based On The Solution Of Spatially Constrained Local And Sub-Local Problems

Juan De Dios Cota
University of Texas at El Paso, jcota2@miners.utep.edu

COLLABORATIVE AND DISTRIBUTED ALGORITHMS FOR LOCALIZATION IN WIRELESS SENSOR NETWORKS BASED ON THE SOLUTION OF SPATIALLY CONSTRAINED LOCAL AND SUB-LOCAL PROBLEMS

JUAN COTA

Department of Electrical and Computer Engineering

APPROVED:

Jose Gerardo Rosiles, Ph.D., Chair

Sergio D. Cabrera, Ph.D.

John A. Moya, PhD.

Rafael Gutierrez, Ph.D.

Patricia D. Witherspoon, Ph.D.
Dean of the Graduate School

to my beautiful daughter

HANIA VALENTINA

COLLABORATIVE AND DISTRIBUTED ALGORITHMS FOR LOCALIZATION IN WIRELESS SENSOR NETWORKS BASED ON THE SOLUTION OF SPATIALLY CONSTRAINED LOCAL AND SUB-LOCAL PROBLEMS

by

JUAN COTA

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at El Paso

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

Department of Electrical and Computer Engineering

THE UNIVERSITY OF TEXAS AT EL PASO

May 2011

Acknowledgements

First of all, I would like to acknowledge to Dr. Jose Gerardo Rosiles. This research project would not have been possible without his advice, support, and guidance (thank you Dr. Rosiles for your patience). Also, I wish to express my love and gratitude to my family members: my beautiful daughter, Hannia Valentina, and my wife, Vanessa. To my parents, Juan and Maria. To my brothers, Carlos Armando, Abdiel Keni, and my sister, Milleli Vianey: you are always in my mind. Special thanks also to my super cool friend Dr. Pablo Rivas Perea for his valuable assistance during the last semesters. I would also like to convey thanks to the members of my dissertation committee for their valuable suggestions. Finally, I acknowledge PROMEP and the Chihuahua Scholarship for their financial support for this research. Thank you God for everything.

Abstract

In this research we present algorithms for the distributed and collaborative localization of nodes for applications in wireless sensor networks. The algorithms are distributed in the sense that each node can estimate its own position using only range information and position estimates from neighboring nodes. The algorithms aim at achieving good accuracy with low computational complexity and low energy consumption. We consider the full localization process consisting of an initialization stage followed by a refinement stage.

For initialization, we propose a *bilateration* algorithm where each node uses a set of anchors and their respective ranges to solve a set of circle intersection problems. These problems are solved through a purely geometric formulation with low computational complexity. The resulting circle intersections are processed to pick those that cluster together and then take their average to produce an initial node location. For the refinement stage, we develop an iterative collaborative variation of multilateration where all sensors solve a spatially-constrained optimization program to find position updates which are used on the next refinement iteration. We introduce two types of objective functions: local and sub-local. In the local case, each sensor tries to minimize the mean absolute range error with all its neighbors simultaneously. In the sub-local case, a node sets a separate objective function with each of it neighbors and produces a set of solutions which are averaged to produce a final update. Our distributed algorithms are characterized by a spatial constraint that limits the solution space to some region around the current position estimate. This constraint allows all the nodes to update their position simultaneously while achieving convergence. In general, simulations show the local objective function performs better than the sub-local case. We note that the proposed approach has computational characteristics that allow its deployment on real mote hardware.

Furthermore, we introduce a simple stopping criterion based on a local threshold τ that relaxes the absolute distance change between position updates. We also introduce a realistic energy model that models consumption at the processor cycle and bit transmission levels. The model

characterizes energy consumption for the localization process over the complete network. Combining τ with the energy model, experimental results show that we can determine the best tradeoff between energy and accuracy performance for a given energy budget. We conclude that there is a strong dependance on the initialization scheme, and that the use of local objective functions provides a better accuracy-energy tradeoff in our simulations. As far we know, there is not a research in WSN localization that presents such a detailed analysis of energy consumption in the localization process.

Finally, we develop schemes for location refinement based on the observation that nodes inside the convex hull formed by the anchors tend to provide position estimates with smaller error. This work leads to a novel refinement step that applies to all the nodes in the network, and achieves an excellent tradeoff between accuracy and energy consumption. The refinement step solves another multilateration optimization problem using the Levenberg Marquardt algorithm. Evidence shows that even for large values of τ the algorithm finds a solution that provides significant improvements on accuracy with minimal energy costs.

Table of Contents

List of Tables

List of Figures

List of Symbols

Symbol	Description
S	Set of Nodes
$\mathbf{p}_i$	Position estimate for the sensor s_i
$\mathbf{z}_i$	True location for the sensor s_i
M	Number of anchors in the network
$\mathbf{A}$	Set of M anchors
$\mathbf{q}_i$	True location for the anchor a_i
N	Number of nodes in the network
R	Range of coverage in sensor nodes
R_{ik}	Range estimate between the anchor a_k and the node s_i
d_{ik}	True distance between the anchor a_k and the node s_i
r_{ij}	Range estimate between two nodes s_i and s_j
$\mathbf{L}$	Set of position estimates
$\mathbf{A}_{jk}$	Subset of two different anchors (a_j, a_k)
Q	Total set of two-anchor subsets given by $\mathrm{Q} = \binom{M}{2}$
$\mathbf{g}_i^{jk}$	Circle intersection obtained by the node s_i using the anchor subset (a_j, a_k)
$\overline{\mathbf{g}}_i^{jk}$	Image of $\mathbf{g}_i^{jk}$
$\mathbf{p}_i^0$	Initial estimates of the sensor s_i
$\mathbf{p}_i^\ell$	Position estimate of the sensor s_i at the iteration ℓ
α_{ij}^ℓ	Range error between the sensor $s_i$and the sensor s_j at the iteration ℓ
β_{ik}^ℓ	Range error between the sensor s_i and the anchor a_k at the iteration ℓ
Ω_i	Constrained search region centered at the location $\mathbf{p}_i^\ell$
$p_{ij}^{\ell+1}$	Local candidate position found by s_i with respect to s_j
η_p	Path-loss exponent

Symbol	Description
δ	Parameter used to constraint the search region
Ω_{ij}	Constrained search region centered at $\mathbf{p}_i^{\ell}$ used to minimize the distance to the node s_j
C	Multiplicative Factor to eliminate the bias of the RSS technique
$\overline{P}_{ij}$	Expected power measurement received by the node s_i from the node s_j
N_i	Set of neighboring nodes of the sensor s_i
$C(\cdot)$	Global cost function
$\|\cdot\|$	Norm 2
s.t.	Subject to
min	Minimize
$\varepsilon\{\cdot\}$	Expected value
σ_{SH}	Noise in decibels
τ	Threshold for the stopping criterion $\left\|\mathbf{p}_i^{\ell} - \mathbf{p}_i^{\ell+1}\right\|$
$\mathbf{R}(\mathbf{p}_k)$	Residual error vector
$\mathbf{I}$	Identity matrix
$\mathbf{J}(\mathbf{p}_k)$	Jacobian of $\mathbf{R}(\mathbf{p}_k)$ at the iteration k
$M_f(\mathbf{p}_k)$	Merit function
$M_f^{'}(\mathbf{p}_k)$	Derivative of the merit function
$\triangle_{LM}$	Levenberg-Marquardt direction
J_i	Total wireless transmissions in the sensor s_i
B_i	Total wireless receptions in the sensor s_i

List of Acronyms

Acronym	Description
CCI	Circle-Circle Intersections
dB	Decibels
DSCL-L	Distributed Spatially Constrained Localization-Local
DSCL-SL	Distributed Spatially Constrained Localization-Sub-Local
dwMDs	Distributed Weighted Multidimensional Scaling
GSLS	Global Spherical Least Squares
GOF	Global Optimization Function
GPS	Global Positioning System
KS	Kolmogorov-Smirnov
LM	Levenberg-Marquardt
LS	Least Squares
ML	Maximum Likelihood
NLLS	Non Linear Least Squares
ND	Neighboring Density
QoT	Quality of Trilateration
QP	Quadratic Programming
RF	Radio Frequency
RMSE	Root Mean Square Error
RoC	Rate of Convergence
RSS	Receive Signal Strength
RSSI	Receive Signal Strength Indicator
RV	Random Variable
Rx	Reception

Acronym	Description
SDP	Semidefinite Programming
SQP	Sequential Quadratic Programming
TDoA	Time Difference of Arrival
ToA	Time of Arrival
ToF	Time of Flight
TRR	Trust Region Reflective
Tx	Transmission
WLS	Weighted Least Squares
WSN	Wireless Sensor Network

Chapter 1

Introduction

1.1 Wireless Sensor Network Localization

Wireless sensor networks (WSNs) are making inroads into the most varied applications. They offer the ability to acquire information at spatial and time scales which were difficult, expensive, or impossible to achieve previously. The low cost of the nodes and the savings on infrastructure allows deployments of tens, hundreds, or even more devices equipped with application specific sensors. WSNs are a technological breakthrough that is changing the social landscape as they are integrated into different aspects of our lives like health care, homeland security, infrastructure monitoring, and transportation to name a few.

On a WSN it is a common assumption to deploy sensors over a region with limited to non-existent control on the position of the motes (i.e., sensor nodes). This situation has given rise to a significant amount of work on self localization schemes for WSNs [74, 57, 64, 1]. The problem of WSN localization is preceded by work on target and source localization for military and communication applications using geometrical (e.g., lateration) and signal processing (e.g., beamforming) techniques [14]. Localization has been an active area of research in WSNs from the beginning. Knowledge of the node positions is crucial to establish the network topology, track objects, monitor an event, determine the quality of coverage, move data through the network, and to determine spatial/geographical relationships for data mining and signal analysis. Moreover, since many of phenomena under analysis may exhibit locality, then it is also desirable to provide proximity relationships among the nodes for in-network analysis and decision making.

1.1.1 Range Estimations

Range-based techniques estimate the true distance between two sensor nodes using timing, power, and/or angle measurements. Timing is considered the most accurate technique to estimate the distance d_{ij} between a sensor s_i and a sensor s_j. It is based on the time of flight (ToF) of a signal (e.g., acoustic or RF), which basically presents two modalities: The time of arrival (ToA) and the time difference of arrival ($TDoA$). These techniques require additional sensor hardware to detect signals (radio frequency or acoustic) and make accurate timing measurements. Knowing the ToF from sensor s_i to sensor s_j and the velocity of propagation of a signal v_p, the distance between the sensors s_i and s_j can be primarily formulated as $d_{ij} = v_p \cdot ToF$. However, this first approximation does not take into account many factors like external environments (e.g., additive noise and multipath signals) and internal time delays, τ_D, on the nodes. Thus, the $TDoA$ technique is used to overcome the time delay problem.

Consider a sensor s_i that should estimate its distance d_{ij} to a sensor s_j. The sensor s_i sends two $starting-signals$ to the sensor s_j via RF and the acoustic medium as shown in Figure 1.1. The RF signal immediately starts a timer in the sensor s_j at time t_s and after a little time the acoustic signal arrives to the sensor s_j stopping the timer at time t_p; consequently, the estimated distance between the two sensors can be found as $d_{ij} \cong (t_s - t_p) \cdot v_s$, where v_s is the speed of the acoustic signal. This technique provides a good approximation to the true distance, but it requires extra hardware on the sensor node.

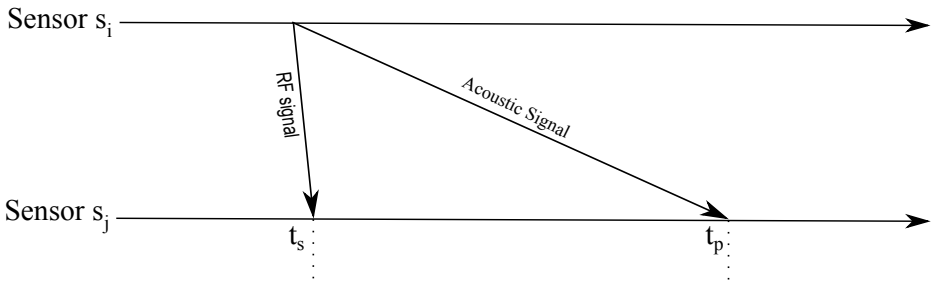


Figure 1.1: $TDoA$ between two sensors

One variation for the TDoA ranging technique consists on synchronizing all anchors a_i ($i = 1, ..., M$ using the technique described in [29]. Once synchronized, multiple signals are broadcast

at the same time by the anchors as shown in Figure 1.2.

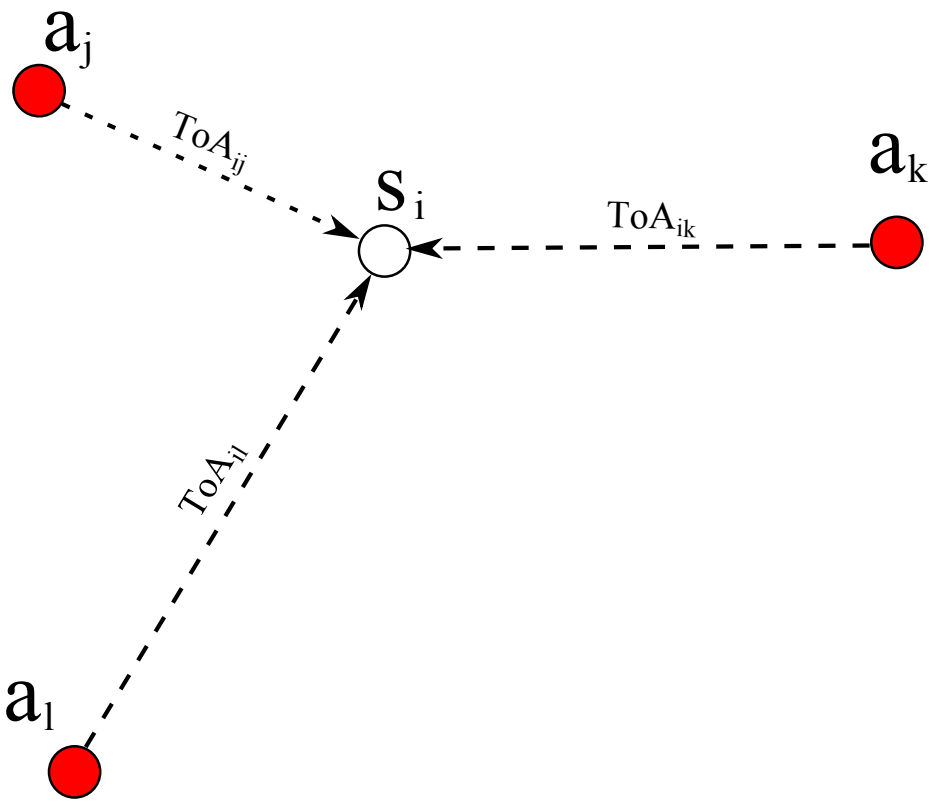


Figure 1.2: TDoA Method

Next, sensor s_i measures the TDoA using the ToAs of anchors and solves for ToF. The process is as follows. Let us to consider the term ToA_{ij} as the time of arrival measured by sensor s_i from a signal that comes from the anchor a_j and so on. Next, assume that each ToA is composed by a τ_D and a ToF. The analysis for three anchors (a_j, a_k, a_l) can be represented in the following manner:

$$\begin{aligned} ToA_{ij} &= \tau_{D_{ij}} + ToF_{ij} \\ ToA_{ik} &= \tau_{D_{ik}} + ToF_{ik} \\ ToA_{il} &= \tau_{D_{il}} + ToF_{il}. \end{aligned} \tag{1.1}$$

Hence, the $TDoA$ for every pair-anchor combination can be calculated as follows:

$$\begin{aligned} TDoA_{jk} &= ToA_{ij} - ToA_{ik} = \left(\tau_{D_{ij}} + ToF_{ij}\right) - \left(\tau_{D_{ik}} - ToF_{ik}\right) \\ TDoA_{jl} &= ToA_{ij} - ToA_{il} = \left(\tau_{D_{ij}} + ToF_{ij}\right) - \left(\tau_{D_{il}} - ToF_{il}\right) \\ TDoA_{kl} &= ToA_{ik} - ToA_{il} = \left(\tau_{D_{ik}} + ToF_{ik}\right) - \left(\tau_{D_{il}} - ToF_{il}\right). \end{aligned} \tag{1.2}$$

Now, considering that every anchor has the same time delay transmission $\tau_{D_{ij}} = \tau_{D_{ik}} = \tau_{D_{il}}$, the equation (1.2) can be rewritten as

$$\begin{aligned} TDoA_{jk} &= ToA_{ij} - ToA_{ik} = ToF_{ij} - ToF_{ik} \\ TDoA_{jl} &= ToA_{ij} - ToA_{il} = ToF_{ij} - ToF_{il} \\ TDoA_{kl} &= ToA_{ik} - ToA_{il} = ToF_{ik} - ToF_{il}. \end{aligned} \tag{1.3}$$

Consequently, using (1.3) sensor s_i solves for the ToF on the anchors. As a result, the estimated distances to each anchor can be easily calculated. Therefore, the $TDoA$ scheme eliminates the τ_D problem introduced by the transmitter. Another way to use the $TDoA$ technique consists of using the cross-correlation between two signals (one known by the receiver) where the largest cross-correlation value between the received and the known signal corresponds to τ_D [10].

Received signal strength (RSS) is another ranging technique based on the measurement of the signal power. This technique is popular because sensor nodes do not require special hardware support to estimate distances. As a first approximation, considering the free space path loss model, the distance d_{ij} between two sensors s_i and s_j can be estimated based on assuming that the power signal decreases in a way that is inversely proportional to the square of the distance $\left(\frac{1}{d_{ij}^2}\right)$. However, in real environments the signal power is attenuated by a factor $d^{-\eta_p}$. The path-loss factor η_p is closely related to geometrical and environmental factors, and it varies from 2 to 4 for practical situations [52]. Also, in realistic environments the signal power is affected by the noise or shadowing effect. An empirical model that integrates the mentioned factors is given by [56]

$$P(d) = P_0(d_0) - 10\eta_p log_{10}\left(\frac{d}{d_0}\right) + \rho_\sigma, \tag{1.4}$$

where $P_0(d_0)$ represents the measured power in dB at the reference distance d_0 from the transmitter and $P(d)$ is the measured power at the distance d. The ρ_σ represents a zero mean Gaussian random variable with standard deviation σ in decibels (dB). This random variable takes into account the shadowing effect. The last two models (ToA and the RSS) will be described in detail in section 4.1.

1.1.2 Localization Techniques for WSNs

The simplest way to determine location on WSNs consists of integrating a global positioning system (GPS) in each sensor. This option has the advantage of geographical precision, but increases the cost, energy-consumption, and the size of a mote (three parameters that always should be minimized in a WSN). Also, GPS requires line-of-sight between the sensor and the satellites [58] which is not possible on indoor environments. A more efficient scheme from the *cost-energy-precision* point of view consists of equipping only a small fraction of the motes with GPS capabilities. These motes serve as anchors or reference nodes which can be used by the remaining sensors to estimate their own positions using the estimate ranges (i.e., distances) to them. Methods to estimate the range among sensors can be the time of arrival (*ToA*), received signal strength (*RSS*), or angle of arrival (*AoA*) as described in last section [24].

Commonly a distributed WSN localization scheme consists of two stages: initialization and refinement. In the first stage each unknown sensor should be able to estimate its initial position by using a set of reference positions (i.e., anchors) and the corresponding range measurements. On the second stage, each sensor uses an iterative algorithm to re-estimate its own position based not only on the anchor information, but also on the current positions and the corresponding range estimates for each available (i.e.,wireless connected) neighbor sensor.

The initialization step requires the derivation of a range estimate $\tilde{d}_{ij}$ between an unlocalized sensor s_i (for $i = 1,2,...,N$) and each of the available anchor nodes a_j (for $j = 1,2,,...M$). These

estimates are affected by environmental conditions and, in cases like RSS, they are more severe as the separation between nodes is increased [2]. Hence, an initial position $\mathbf{x_i^*}$ (in Cartesian coordinates) for s_i is typically set as the solution to an estimation or an optimization problem. For instance, a least-squares problem can be defined as

$$\mathbf{x_i^*} = \arg\min_{\mathbf{x}} \sum_{\mathbf{j=1}}^{\mathbf{M}} (\|\mathbf{x} - \mathbf{q_j}\| - \tilde{\mathbf{d}}_{\mathbf{i,j}})^{\mathbf{2}}, \tag{1.5}$$

where each $\mathbf{q}_j$ corresponds to the *known* coordinates of a_j , and the norm $\|\cdot\|$ is the Euclidean distance. Hence, the initial location $\mathbf{x_i^*}$ is selected as the one that provides the best compromise for all $\tilde{d_{ij}}$. Given the possibility of having large range errors (particularly if the anchors are far from the sensor) and a small number of anchors, the solutions $\mathbf{x_i^*}$ should only be considered a first approximation. Further processing should be done to improve or refine the locations for all s_i.

Next, on the refining process the localization problem becomes more complex because we need to consider additional information which can be obtained from all neighboring sensors s_j. This information includes position estimates and the ranging information between each s_i and all its neighbors using the ranging techniques mentioned above. Hence, the information available to each s_i is highly unreliable.

In summary, the WSN localization problem is commonly modeled as non-convex with non-linear constraints where a non-linear optimization algorithm should be applied to obtain the best solutions. A variety of methods have been proposed: least-squares optimization (described before), quadratic programming, penalty and augmented Lagrangian methods, sequential quadratic programming (SQP), and interior point methods [48]. A more detailed information about the state of the art in WSN localization is described in the next subsection.

1.1.3 Related Work

If a WSN localization problem is formulated as in (1.5), non-linear least-squares (NLLS) optimization algorithms can be used to solve the localization problem; this non-linear optimization problem can be solved using either Gauss-Newton or Quasi-Newton methods [48, 44]. However,

it is known that NLLS methods are suboptimal (do not reach global convergence) in general [11]. Some variations of this method have been proposed to improve the accuracy; such is the case of the Weighted Least Square approach (WLS) [11]. Unfortunately, this option is also subject to local minima and requires a sufficiently close initial guess to the true locations to ensure global convergence [43]. In [44], an iterative descent procedure (i.e., Gauss-Newton method) is used in a centralized way to solve the NLLS problem. Also, if the pair-wise error measurements are considered Gaussian distributed, the Maximum Likelihood (ML) centralized method proposed in [49] can be stated as NLLS problem; thus, the problem can be solved iteratively [11]. Another interesting centralized scheme was proposed in [20] where the WSN localization problem is modeled as linear or semidefinite program (SDP), and a convex optimization is used to solve problem. The centralized algorithms solve the localization problem in a base station or processing center, where all pair-wise distances are gathered via wireless transmission. Thus, energy-conservation and robustness of centralized schemes are affected by the network size, topology, and mote range [67, 74].

On the other hand, if the mathematical model for the WSN localization problem can be stated and solved on a distributed form (i.e., each sensor being able to estimate its own position), the amount of wireless communications among sensors could be greatly reduced. Moreover, the whole WSN can be tolerant to node failures. Recently, many distributed algorithms have been proposed coming from different perspectives. In [31], a novel distributed weighted multidimensional scaling (dwMDS) that corresponds to the WLS approach is proposed. This algorithm is a variation of classical centralized MDS. In a similar way, [13] using the NLLS approach divides a global optimization function, subject to non linear geometrical constraints, into local optimization functions that are solved in a distributed way through the Gauss-Newton method. In [40], a robust least squares scheme (RLS) for multi-hop node localization is proposed. This approach reduces the effects of error propagation by introducing a regularization parameter in the covariance matrix. However, the computational cost to mitigate the adverse effects of error propagation is too high at energy-constrained nodes. Similarly, the authors of [71] propose a Quality of Trilateration method (QoT) for node localization. This approach provides a quantitative evaluation

of different geometric forms of trilaterations. However, it seems to be that the main idea of this methodology depends on the quality or resolution of geometric forms (i.e., like image processing) which is impractical to be implemented in resource-constrained devices with limited memory and processing capabilities (i.e., nodes). Also, a divide-and-conquer approach was proposed in [26], where graph theory and a penalty function are used to minimize a global optimization function. Other localization algorithms have formulated the WSN localization problem like a SDP problem where interior point methods are successfully applied [61, 48]. The main idea behind the SDP formulation is to linearize quadratic constraints (i.e., non-convex) using relaxation [15]. Thus, the resulting convex optimization problem can be solved efficiently in polynomial time [6, 36, 61]. Using a different approach, a push-pull physical model is applied in [38] to design a distributed localization algorithm, where force vectors are used to iteratively re-estimate the positions of each unknown sensor until convergence is achieved.

1.1.4 Optimization Concepts (Local and Global Solutions)

According to [32], the mathematical formulation for a local unconstrained minimization function consists of finding the minimum of a real-valued objective function $f(\mathbf{x})$ of N variables, $\mathbf{x} \in R^N$, where the independent variable $\mathbf{x}$ is not subject to any constraints, and the function f is defined for all $\mathbf{x}$. A general formulation can be stated as follows:

$$\arg\min_{\mathbf{x}} f(\mathbf{x}), \tag{1.6}$$

where the solution implies to find a point $\mathbf{x}^*$ such that $f(\mathbf{x}^*) \leq f(\mathbf{x})$ for all $\mathbf{x}$ close to $\mathbf{x}^*$. On the other hand, if the objective function $f(\mathbf{x})$ is defined over a set of points $\mathbf{x}$ belonging to Ω, which should satisfy certain constraints, the local constrained minimization function can be stated as

$$\arg\min_{\mathbf{x}\in\Omega} f(\mathbf{x}), \tag{1.7}$$

where the solution implies to find a point $\mathbf{x}^*$ such that $f(\mathbf{x}^*) \leq f(\mathbf{x})$ for all $\mathbf{x} \in \Omega$ close to $\mathbf{x}^*$.

Global Solutions are much harder to find than local solutions. Global optimization methods

have the harder task of determining the smallest local minimum (i.e., the global minimum) of a finite set of local minima. Formally, global optimization consists of minimizing a real-valued function $f(\mathbf{x})$ through independent variables $\mathbf{x} \in R^N$. It is common to express this kind of global unconstrained minimization functions like

$$\arg\min_{\mathbf{x}} f(\mathbf{x}) \quad for\ all\ \mathbf{x}. \tag{1.8}$$

In the case of global constrained optimization, the variable $\mathbf{x}$ is constrained to a certain set of solutions. Thus, the global optimization function can be stated as follows:

$$\arg\min_{\mathbf{x}\in\Omega} f(\mathbf{x}), \tag{1.9}$$

where the solution implies to find a point $\mathbf{x}^*$ such that $f(\mathbf{x}^*) \leq f(\mathbf{x})$ for all $\mathbf{x} \in \Omega$.

1.2 Problem Description

Geometric concepts are strongly involved in the mathematical modeling for solving WSN localization problems. The mathematical model for realistic problems commonly results in a no convex optimization problem that tends to be computationally complex with a finite number of local minima (generated by the noisy range estimations) which do not guarantee to find globally minimal solutions [9]. Let us describe this mathematical model as follows: consider a set of N sensor nodes $\mathbf{S} = \{s_1, s_2, \ldots, s_N\}$, randomly deployed over a 2-D region whose locations are unknown. We represent these unknown locations with $\mathbf{p}_i = [x_i, y_i]^T$ and their true locations with $\mathbf{z}_i = [x_i, y_i]^T$. Further, we assume the presence of M anchor nodes equipped with GPS or a similar scheme to self-localize $\mathbf{A} = \{a_1, a_2, \ldots, a_M\}$, also randomly deployed over the same 2-D region with known locations $\mathbf{q}_k = [x_k, y_k]^T$. Also, for practical situations we are considering $M \ll N$ with $M > 2$ in a 2-D scenario, but it can be easily generalized to the 3-D case with the condition $M > 3$. To keep our description simple, we identify the anchor nodes simply as *anchors* and the non-localized motes simply as *sensor nodes*, *nodes*, or *sensors*.

Moreover, we assume that ideally any sensor can estimate pairwise ranges with its neighbor sensors using time-of-arrival (ToA) or received signal strenght (RSS) techniques [24], if these neighboring nodes are within a radio range R[1]. For example, the range estimate between the anchor a_k and node s_i denoted as

$$R_{ik} = R_{ki} = d_{ik} + e_{ik} \tag{1.10}$$

is available for the ith node if $d_{ik} = \|\mathbf{z}_i - \mathbf{q}_k\| < R$. The variable e_{ik} represents an error introduced by the environmental noise, propagation distortion, and the ranging technique. In a similar way, the range estimate between two sensor nodes s_i and s_j denoted as

$$r_{ij} = r_{ji} = d_{ij} + e_{ij} \tag{1.11}$$

is available for the ith node if $d_{ij} = \left\|\mathbf{z}_i - \mathbf{z}_j\right\| < R$.

If we consider that each node is able to estimate its distance (e.g., using DV-hop [45] in multi-hop environments) to at least three non collinear anchors, each node will be able to estimate its own initial location using lateration techniques [35]. However, most lateration algorithms spend too much time and energy-consumption to find raw initial estimates, which in practical situations have a significant impact on the robustness and convergence of the algorithm [48].

Basically the problem in WSN localization consists of estimating the unknown position for each sensor s_i such that the norms of these positions minimize the residuals with the corresponding ranges r_{ij} and R_{ik}. The solution to this problem is one of the most challenging problems in WSNs. The problem can be mathematically formulated as the following Global Optimization Function (GOF)

[1]It is assumed that all sensor nodes have the same circular coverage range given by a value R

$$\arg\min_{\mathrm{L}} \sum_{(i,j)\in N_R} \left| \left\| \mathbf{p}_i - \mathbf{p}_j \right\| - r_{ij} \right| + \sum_{(i,k)\in M_R} \left| \left\| \mathbf{p}_i - \mathbf{q}_k \right\| - R_{ik} \right|$$

$$\begin{aligned} &\text{such that} \\ &N_R = \left\{ (i,j) \,|\, i < j, \left\| \mathbf{z}_i - \mathbf{z}_j \right\| < R \right\} \\ &M_R = \left\{ (i,k) \,|, \left\| \mathbf{z}_i - \mathbf{q}_k \right\| < R \right\} \end{aligned} \tag{1.12}$$

where the set of locations L = p1,p2, . . . ,pN is the solution to the global optimization function.

As described before, the localization problem results in a non-linear, non-convex, and NP-hard problem; and our challenge consists of solving (1.12) with an efficient algorithm. By efficient we mean a distributed algorithm which can be able to provide high localization accuracy and low energy consumption when compared with well known state-of-the-art localization algorithms.

1.3 Scope of Research

In recent years, the use of WSNs has had a wide range of potential applications in environmental monitoring, reporting, prediction, health monitoring, industrial processes, target tracking, detection of natural disasters, smart homes, and military applications. Here, the location accuracy is a fundamental issue that affects the performance of such mentioned applications. Node localization algorithms have been extensively studied, and the way to solve the location problem comes from different perspectives [23].

In this research, we propose localization algorithms which aim at achieving good accuracy with low computational complexity and low energy consumption. To achieve this objective we consider the full localization process. From our point of view this process consists of an initialization stage followed by a refinement stage.

For initialization, we propose a bilateration algorithm where each node uses a set of anchors and their respective ranges to solve a set of circle intersection problems using a geometric for-

mulation. The solutions from these geometry problems are processed to pick those that cluster around the location estimate and then take their average to produce an initial node location.

For the refinement stage, we will present an algorithm that consists of a variation of multilateration. The algorithm is an iterative collaborative process where all sensors solve a spatially-constrained optimization program to determine position updates which are broadcasted to continue position refinement over the next iteration. We introduced two types of objective functions: local and sub-local. In the local case, each sensor tries to minimize the mean absolute range error with all its neighbors simultaneously. In the sub-local case, the objective function is actually a set of objective functions were the range error is satisfied between a node an each one of its neighbors separately. A final solution is found be averaging the sub-local solutions.

Our iterative algorithm is characterized by a spatial constraint that limits the solution space some region around the current position estimate. This constraint allows all the nodes to update their position simultaneously while achieving convergence. In general, simulations show the local objective function performs better than the sub-local case. We note that the proposed approach has computational characteristics that allow its deployment on real mote hardware. Furthermore, we introduce a simple stop criteria based on a threshold τ that relaxes the absolute distance change between position updates. Each node stops updating its position once τ is exceeded. We performed an extensive evaluation of the tradeoff between localization accuracy and energy consumption. This is a key metric for an iterative scheme as we want to minimize the number of wireless transmissions/receptions (the most energy expensive operations on a WSN) while providing the best localization accuracy. Obviously these are conflicting goals; as τ increases, the cost of wireless communication decreases while the localization RMSE increases. We found that there is a strong dependence on the initialization scheme and that the local objective function provides a better accuracy-energy tradeoff.

We introduce a realistic energy model that models consumption at the processor cycle and bit transmission levels. The model characterizes energy consumption for the localization process over the complete network. Experimental results show that, under certain network conditions, it is easy to determine the best tradeoff between energy and accuracy performance for a given

energy budget. As far we know, there is not a research in WSN localization that presents such a detailed analysis of the energy consumption in the WSN localization process.

Finally, we developed schemes for location refinement based on the observation that nodes inside the convex hull formed by the anchors tend to provide position estimates with smaller error. This work leads to a novel refinement step that applies to all the nodes in the network, and achieves an excellent tradeoff between accuracy and energy consumption. The refinement step solves a new multi-lateration optimization problem using the Levenberg Marquardt algorithm. Evidence shows that even for large values of τ (the stop criteria threshold), the algorithm finds a solution that provides significant improvements on accuracy with minimal computational and energy costs.

Chapter 2

Anchor-based Algorithms for Initial Position Estimation

2.1 Problem Definition for Single-Hop Localization

Consider a set of N wireless sensor nodes $\mathbf{S} = \{s_1, s_2, \ldots, s_N\}$, randomly distributed over a 2-D region whose locations are unknown. We represent these unknown locations with vectors $\mathbf{z}_i = [x_i, y_i]^T$. Further, we assume the presence of a set $\mathbf{A} = \{a_1, a_2, \ldots, a_M\}$ of M reference or anchor nodes with known position $\mathbf{q}_j = \left[x_j, y_j\right]^T$. Anchor nodes, a_i, are equipped with GPS or a similar scheme to self localize. Also, for practical situations $M \ll N$ with $M > 2$. We develop our discussion assuming a 2-D scenario, but it can be easily generalized to the 3-D case.

Moreover, we assume that any sensor can estimate pairwise ranges with its neighbors using time-of-arrival (ToA) or radio signal strength (RSS) techniques [24]. Denote the range estimate between the node s_i and anchor a_j as

$$R_{ij} = d_{ij} + e_{ij} \tag{2.1}$$

where d_{ij} is the true distance between a_j and s_i, and e_{ij} represents the error introduced by environmental noise, propagation distortion, and the ranging technique. Then the solution to the one-hop localization problem for a node s_i consists of minimizing the sum of certain weighted error-distance function $e_w(\cdot)$ as follows:

$$\mathcal{F} = \underset{\mathbf{p}_i}{arg\,min} \sum_{j=1}^{M} e_w \left(\left\| \mathbf{q}_j - \mathbf{p}_i \right\| - R_{ij} \right), \tag{2.2}$$

where $\mathbf{p}_i = [x_i, y_i]^T$ represents the most likely position for the sensor s_i that minimizes $\mathcal{F}$, $\|\cdot\|$ represents the Euclidean norm, and $e_w(x)$ represents a function that provides an specific weight to the argument x (i.e., error distance). For example, $e_1(x) = |x|$ or $e_2(x) = (x)^2$ are the most used functions due their tractability and efficiency in both mathematical and computational analysis.

2.2 Least-Squares Multilateration Localization Algorithms

In this section, we describe well known multilateration schemes that provide solutions to the Least-Square (LS) problem for location estimates using noisy ranging information derived from ToA and RSS ranging techniques. We describe methods based on closed-form solutions and an iterative method using the Levenberg-Marquardt algorithm. Both methods rely only on anchor information/measurements to individuality estimate their own position. The estimates can be used as an initialization point to distributed algorithms that collaboratively refine locations.

2.2.1 Closed-Form LS Multilateration

Closed-Form methods have the advantage of fast time processing useful for constrained devices (i.e., motes) where the energy conservation represents one of the major concerns. However, this approach is also subject to inaccurate estimates due to noisy ranging measurements so in most cases this approach is not a suitable option in real WSN scenarios where current ranging techniques are not able to provide the required accuracy on the ranging measurements. For example, Spherical Intersection (SX), Spherical Interpolation (SI), and Global Spherical Least Squares (GSLS) [29] can solve a non linear set of equations using direct formulas. These approaches provide good accuracy in the estimated positions under conditions like small biases and small standard deviations, but they also provide meaningless estimates under noisy environments [28]. A more robust closed-form scheme consists of using the classical LS multilateration discussed next [12, 40, 66].

Consider that a sensor s_i with cartesian position $\mathbf{p}_i = [x_i, y_i]^T$ has already estimated its range R_{ij} to M anchors. For each anchor a_j with position $\mathbf{q}_j = [x_j, y_j]^T$, an equation $\left\|\mathbf{q}_j - \mathbf{p}_i\right\|^2 = R_{ij}^2$ is generated as shown the next formulas:

$$\begin{array}{ccc} \left\|\mathbf{q}_1-\mathbf{p}_j\right\|^2=R_{i1}^2 & & (x_1-x_i)^2+(y_1-y_i)^2=R_{i1}^2 \\ \left\|\mathbf{q}_2-\mathbf{p}_j\right\|^2=R_{i2}^2 & \Leftrightarrow & (x_2-x_i)^2+(y_1-y_i)^2=R_{i2}^2 \\ \vdots & & \vdots \\ \left\|\mathbf{q}_M-\mathbf{p}_j\right\|^2=R_{iM}^2 & & (x_M-x_i)^2+(y_M-y_i)^2=R_{iM}^2 \end{array} \tag{2.3}$$

The system of equations (2.3) can be linearized by subtracting the first equation $(j=1)$ from the last $M-1$ equations as follows:

$$\begin{array}{c} (x_2-x_i)^2+(y_2-y_i)^2-R_{i2}^2-(x_1-x_i)^2-(y_1-y_i)^2+R_{i1}^2=0 \\ (x_3-x_i)^2+(y_3-y_i)^2-R_{i3}^2-(x_1-x_i)^2-(y_1-y_i)^2+R_{i1}^2=0 \\ \vdots \\ (x_M-x_i)^2+(y_M-y_i)^2-R_{iM}^2-(x_1-x_i)^2-(y_1-y_i)^2+R_{i1}^2=0 \end{array} \tag{2.4}$$

After some manipulations we arrive to the following:

$$\begin{array}{c} -2x_2x_i+2y_2y_i+2x_1x_i+2y_1y_i=R_{i2}^2-R_{i1}^2+x_1^2+y_1^2-x_2^2-y_2^2 \\ -2x_3x_i+2y_3y_i+2x_1x_i+2y_1y_i=R_{i3}^2-R_{i1}^2+x_1^2+y_1^2-x_3^2-y_3^2 \\ \vdots \\ -2x_Mx_i+2y_My_i+2x_1x_i+2y_1y_i=R_{iM}^2-R_{i1}^2+x_1^2+y_1^2-x_M^2-y_M^2, \end{array} \tag{2.5}$$

factorizing we have

$$\begin{array}{c} -2x_i(x_2-x1)-2y_i(y_2-y_1)=R_{i2}^2-R_{i1}^2+x_1^2+y_1^2-x_2^2-y_2^2 \\ -2x_i(x_3-x1)-2y_i(y_3-y_1)=R_{i3}^2-R_{i1}^2+x_1^2+y_1^2-x_3^2-y_3^2 \\ \vdots \\ -2x_i(x_M-x1)-2y_i(y_M-y_1)=R_{iM}^2-R_{i1}^2+x_1^2+y_1^2-x_M^2-y_M^2, \end{array} \tag{2.6}$$

then we arrive to a linear system that can be represented in a matrix form as

$$\mathbf{A}\mathbf{p}_i=\mathbf{b}, \tag{2.7}$$

where

$$\mathbf{A} = -2 \begin{bmatrix} x_2 - x_1 & y_2 - y_1 \\ x_3 - x_1 & y_3 - y_1 \\ \vdots & \vdots \\ x_M - x_1 & y_M - y_1 \end{bmatrix}_{(M-1)x2} \tag{2.8}$$

$$\mathbf{b} = \begin{bmatrix} R_{i2}^2 - R_{i1}^2 + x_1^2 + y_1^2 - x_2^2 - y_2^2 \\ R_{i3}^2 - R_{i1}^2 + x_1^2 + y_1^2 - x_3^2 - y_3^2 \\ \vdots \\ R_{iM}^2 - R_{i1}^2 + x_1^2 + y_1^2 - x_M^2 - y_M^2 \end{bmatrix}_{(M-1)x1} \tag{2.9}$$

Now the least square solution to equation (2.7) is to determine an estimate for $\mathbf{p}_i$ that minimizes

$$\begin{aligned} f(\mathbf{p}_i) &= \min_{\mathbf{p}_i} \left\{ \tfrac{1}{2} \|\mathbf{A}\mathbf{p}_i - \mathbf{b}\|^2 \right\} \\ &= \min_{\mathbf{p}_i} \left\{ \tfrac{1}{2} (\mathbf{A}\mathbf{p}_i - \mathbf{b})^T (\mathbf{A}\mathbf{p}_i - \mathbf{b}) \right\}. \end{aligned} \tag{2.10}$$

After some manipulations we obtain the following:

$$f(\mathbf{p}_i) = \min_{\mathbf{p}_i} \left\{ \frac{1}{2} \mathbf{p}_i^T \mathbf{A}^T \mathbf{A} \mathbf{p}_i - \mathbf{p}_i^T \mathbf{A}^T \mathbf{b} + \frac{1}{2} \mathbf{b}^T \mathbf{b}, \right\} \tag{2.11}$$

and the gradient of f at $\mathbf{p}_i$ is

$$\bigtriangledown f(\mathbf{p}_i) = \mathbf{A}^T \mathbf{A} \mathbf{p}_i - \mathbf{A}^T \mathbf{b} = 0, \tag{2.12}$$

which provides the estimate (i.e., normal equations) to equation (2.7):

$$\hat{\mathbf{p}}_i = (\mathbf{A}^T \mathbf{A})^{-1} \mathbf{A}^T \mathbf{b} \tag{2.13}$$

Solving for (2.13) may not work properly if $\mathbf{A}^T \mathbf{A}$ is close singular, so a recommended approach is to use a Tikhonov regularization as follows:

For $\mu > 0$ (e.g., close to zero)

$$\begin{aligned} f_\mu(\mathbf{p}_i) &= \min_{\mathbf{p}_i} \left\{ \tfrac{1}{2} \|\mathbf{A}\mathbf{p}_i - \mathbf{b}\|^2 + \tfrac{\mu}{2} \|\mathbf{p}_i\| \right\} \\ &= \min_{\mathbf{p}_i} \left\{ \tfrac{1}{2} \mathbf{p}_i^T \mathbf{A}^T \mathbf{A}\mathbf{p}_i - \mathbf{p}_i^T \mathbf{A}^T \mathbf{b} + \tfrac{1}{2} \mathbf{b}^T \mathbf{b} + \tfrac{\mu}{2} p_i^T \mathbf{p}_i \right\}. \end{aligned} \tag{2.14}$$

Then the gradient of f_μ at $\mathbf{p}_i$ is

$$\bigtriangledown f_\mu(\mathbf{p}_i) = \mathbf{A}^T \mathbf{A}\mathbf{p}_i - \mathbf{A}^T \mathbf{b} + \mu \mathbf{p}_i = 0. \tag{2.15}$$

Factorizing we arrive to a robust estimate for the LS problem where the idea is to modify Eigen-values to avoid working with zero Eigen-values [18, 40].

$$\hat{\mathbf{p}}_i = (\mathbf{A}^T \mathbf{A} + \mu \mathbf{I})^{-1} \mathbf{A}^T \mathbf{b}. \tag{2.16}$$

2.2.2 Iterative LS Algorithms

Iterative methods are usually employed either when large-data set of information need to be processed or when not exact solution to a certain problem is feasible e.g., non-linear systems of equations [48]. Optimization techniques represent a good alternative to solve such non-linear equations using an iterative procedure. Optimization algorithms that solve Non-Linear Least-Square (NLLS) problems (i.e., the WSN localization problem) have been extensively proposed where the Newton or Quasi-Newton methods are iteratively used to minimizing some residuals [13, 31, 44]. The next paragraphs describe two well known iterative algorithms that are used to solve the NLLS problem: the Levenberg-Marquardt (LM) and the Trust-Region-Reflective (TRR).

Assuming that a node denoted $\mathbf{s}_i$, with Cartesian position $\mathbf{p}_i = [x_i, y_i]^T$, estimates its distance R_{ij} to M anchors denoted a_j, with positions $\mathbf{q}_j = [x_j, y_j]^T$, with $j = 1, \ldots, M$. Consider the following residual error vector:

$$\mathbf{R}(\mathbf{p}_i) = \begin{bmatrix} R_{i1} - \|\mathbf{p}_i - \mathbf{q}_1\| \\ R_{i2} - \|\mathbf{p}_i - \mathbf{q}_2\| \\ \vdots \\ R_{iM} - \|\mathbf{p}_i - \mathbf{q}_M\|. \end{bmatrix} \tag{2.17}$$

Therefore, to find the more likely position of $\mathbf{p}_i$, the program

$$\min_{\mathbf{p}_i} f(\mathbf{p}_i) = \min_{\mathbf{p}_i} \left(\frac{1}{2} \mathbf{R}(\mathbf{p_i})^T \mathbf{R}(\mathbf{p_i}) \right) \tag{2.18}$$

is solved, which is the least square problem.

To solve equation (6.3) we employ the TRR algorithm and the LM algorithm. The TRR algorithm uses a sub-space trust-region method to minimize a function $f(x)$. Here, approximations to f inside of a trust-region are iteratively required. The three main concerns in this algorithm are how to choose and compute the approximation to the function, how to choose and modify the trust region, and, finally, how to minimize over the sub-space trust-region. Even though the TRR algorithm provides an accurate solution for the WSN initial estimates, it is expensive (computationally speaking) for constrained sensor nodes [17].

On the other hand, the LM algorithm uses the search direction approach (a mix between the Gauss-Newton direction and the steepest descent direction) to find the solution to (6.3). This algorithm is faster than the typical gradient descent methodology, and also it avoids dangerous operations with singular matrices as the pure Newton method does, so this methodology represents a good algorithm for comparison due its robustness, speed, and accuracy [72]. Following the procedure presented in [18], (6.3) can be solved by the Line Search Levenberg-Marquardt methodology as shown the algorithm presented in Table 1, where $\|\cdot\|$ is the ℓ-2 norm, $\mathbf{I}$ is the identity matrix, R_{ij} is the estimated distance between the mote s_i and the anchor a_j, $\mathbf{J}(\mathbf{p}_k)$ represents the Jacobian of $\mathbf{R}(\mathbf{p}_k)$ at the iteration k, and $\mathbf{M}_f(\mathbf{p}_k)$ is the merit function given by

$$\frac{1}{2} \mathbf{R}^T(\mathbf{p}_k) \mathbf{R}(\mathbf{p}_k). \tag{2.19}$$

The derivative of the merit function at the iteration k is

$$\mathbf{M}_f^{'}(\mathbf{p}_k) = \mathbf{J}^T(\mathbf{p}_k)\mathbf{R}(\mathbf{p}_k), \tag{2.20}$$

Δ_{LM} is the Levenberg-Marquardt direction,

$$\mu_k = \rho \left\| \mathbf{J}^T(\mathbf{p}_k)\mathbf{R}(\mathbf{p}_k) \right\|, \tag{2.21}$$

where $\rho \in (0,1)$, and finally

$$\mathbf{p}_0 = \frac{1}{M}\sum_{j=1}^{M} \mathbf{q}_j \tag{2.22}$$

provides the initial guess required for the TRR and LM iterative algorithms.

Algorithm 1 Levenberg-Marquardt methodology.

Require: an initial position $\mathbf{p}_0$
Ensure: a solution $\mathbf{p}_{k+1}$
1: **Initialize:** k=0,τ=Threshold, $\rho = 0.05$
2: **do**
3: Solve: $(\mathbf{J}^T(\mathbf{p}_k)\mathbf{J}(\mathbf{p}_k) + \mu_k \mathbf{I})\Delta_{LM} = -\mathbf{J}^T(\mathbf{p}_k)\mathbf{R}(\mathbf{p}_k)$
4: Find the sufficient decrease (Armijo's condition):
5: such that $\alpha_k = (\frac{1}{2})^t \, for \; t = 0, 1, \ldots$
6: satisfies $\mathbf{M}_f(\mathbf{p}_k + \alpha_k \Delta_{LM}) \leq \mathbf{M}_f(\mathbf{p}_k) + 10^{-4}\alpha_k \mathbf{M}_f^{'}(\mathbf{p})^T \Delta_{LM}$
7: Update position: $\mathbf{p}_{k+1} = \mathbf{p} + \alpha_k \Delta_{LM}$
8: Update μ_k: $\mu_k = \rho \|\mathbf{J}^T(\mathbf{p}_k)\mathbf{R}(\mathbf{p}_k)\|$
9: **while**$(\|\mathbf{p}_{k+1} - \mathbf{p}_k\| \leq \tau$ or $k \leq 100)$

2.3 A Bilateration Localization Method

In this section we introduce a bilateration method for WSN localization which can be used as the initialization step for iterative localization schemes. This algorithm avoids iterative procedures, gradient calculations, and matrix operations that increase the internal processing in a constrained device. This part of the research was done independently of the work presented in [39]. Even though both schemes share the same idea (i.e., bilateration), the procedure and the scope of both works are different. We show that it is possible to obtain a position estimate by solving a set of bilateration problems between a sensor node and its neighboring anchors, and then fusing the solutions according to the geometrical relationships among the nodes. Our aim is to find a scheme that can be deployed on a computationally constrained node. We argue that bilateration is an attractive option as the localization problem is divided on smaller sub-problems which can be efficiently solved on a mote. Next we start our development by introducing the typical assumptions and definitions considered in a WSN localization problem.

Let us define anchor subsets $\mathbf{A}_{jk} \subset \mathbf{A}$ such that $\mathbf{A}_{jk} = \{a_j, a_k\}$ with $j \neq k$. Hence, there is a total of $Q = \binom{M}{2}$ anchor subsets. Without loss of generality, consider the case for one node s_i that receives from a subset A_{jk} the anchor positions $\mathbf{q}_j$ and $\mathbf{q}_k$, and computes the respective ranges R_{ij} and R_{ik} using RSS or ToA measurements. A possible geometrical scenario for this configuration is shown in Figure 2.1. We can appreciate from this example that the range estimates R_{ij} is larger than d_{ij} and R_{ik} is shorter than d_{ik}. Now, consider the two range circles shown in the figure; one with its origin at $\mathbf{q}_j$ and radius R_{ij}, and the second with center in $\mathbf{q}_k$ and radius R_{ik}. Next, define the two circle intersection points as $\mathbf{g}_i^{jk}$ and $\overline{\mathbf{g}}_i^{jk}$, where $\overline{\mathbf{g}}_i^{jk}$ is the reflection of $\mathbf{g}_i^{jk}$ with respect to the (imaginary) line that connects $\mathbf{q}_j$ and $\mathbf{q}_k$. In this case, the superscript jk represents the anchor subset A_{jk}. To simplify our discussion, we drop the superscripts, and only used them when more than one anchor subset is involved in our discussion.

In our approach, node s_i determines the circle-circle intersections (*CCI*) $\mathbf{g}_i$ and $\overline{\mathbf{g}}_i$ by solving the closed-form expresion reported in [8]. For instance, consider the two right triangles formed by

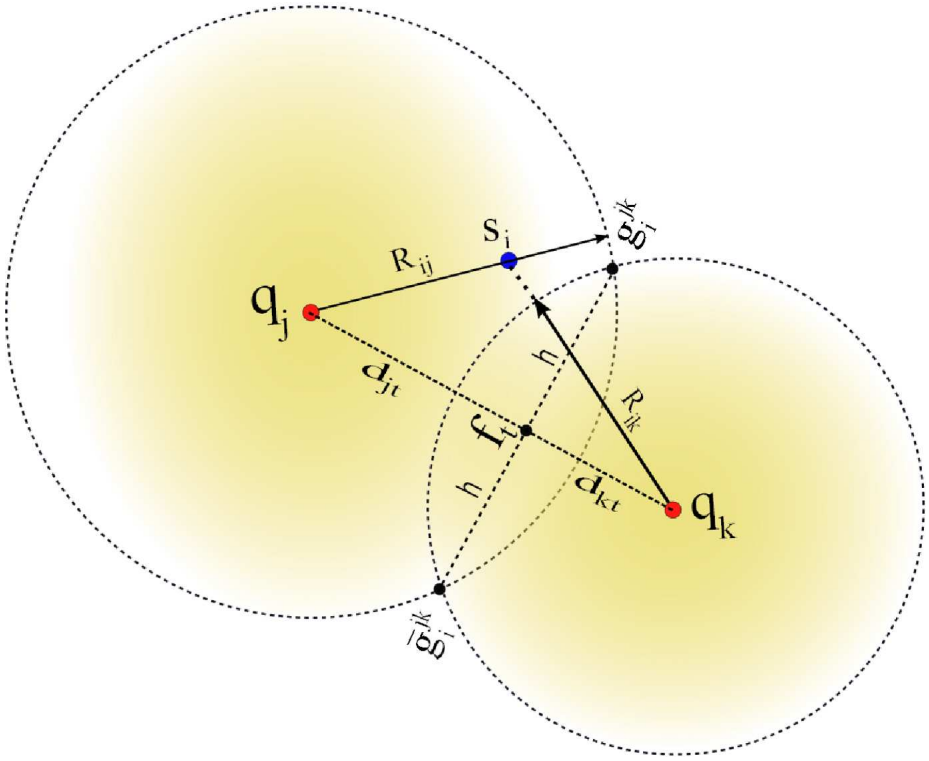


Figure 2.1: Sensor s_i finding its two feasible solutions $(\mathbf{g}_i, \overline{\mathbf{g}}_i)$ based on the anchors locations $(\mathbf{q}_k, \mathbf{q}_j)$ and their respective anchor range measurements (R_{ij}, R_{ik}).

the coordinates $(\mathbf{q}_j, \mathbf{g}_i, \mathbf{f}_t)$ and $(\mathbf{q}_k, \mathbf{g}_i, \mathbf{f}_t)$ in Figure 2.1, which satisfy the following relationships:

$$d_{jt}^2 + h^2 = R_{ij}^2 \tag{2.23a}$$

$$d_{kt}^2 + h^2 = R_{ik}^2, \tag{2.23b}$$

respectively. The distance d_{jt} can be obtained by solving for h^2 in (2.23a) (2.23b):

$$R_{ij}^2 - d_{jt}^2 = R_{ik}^2 - d_{kt}^2, \tag{2.24}$$

and letting $d = \left\|\mathbf{q}_j - \mathbf{q}_k\right\| = d_{jt} + d_{kt}$ resulting in

$$R_{ij}^2 - d_{jt}^2 = R_{ik}^2 - (d - d_{jt})^2 \tag{2.25a}$$

$$R_{ij}^2 = R_{ik}^2 - d^2 + 2 \cdot d \cdot d_{jt} \tag{2.25b}$$

$$d_{jt} = \frac{R_{ij}^2 - R_{ik}^2 \cdot d^2}{2 \cdot d} \tag{2.25c}$$

where the position $\mathbf{f}_t = [x_t, y_t]^T$ is obtained as follows:

$$\mathbf{f}_t = \mathbf{q}_j + \frac{d_{jt}}{d} \left(\mathbf{q}_k - \mathbf{q}_j \right). \tag{2.26}$$

Finally, the circle intersection $\mathbf{g}_i = [x_i, y_i]^T$ is computed as

$$x_i = x_t \pm \frac{h}{d}(y_k - y_j) \tag{2.27a}$$

$$y_i = y_t \mp \frac{h}{d}(x_k - x_j), \tag{2.27b}$$

where $\mathbf{q}_j = [x_j, y_j]^T, \mathbf{q}_k = [x_k, y_k]^T$, and h is easily obtained from equation (2.23). The complementary signs of equations (2.27a) and (2.27b) are used to obtain the solution for $\overline{\mathbf{g}}_i$.

Each node s_i applies the *CCI* procedure using all Q subsets $\mathbf{A}_{jk}$. For instance, $\mathbf{g}_i^{jk}$ and $\overline{\mathbf{g}}_i^{jk}$ are obtained from the subset $\mathbf{A}_{jk}$, $\mathbf{g}_i^{j\ell}$ and $\overline{\mathbf{g}}_i^{j\ell}$ are obtained from the subset $\mathbf{A}_{j\ell}$, and so on. Hence, a sensor node will have $2Q$ possible initial position estimates where half are considered mirror solutions which should be eliminated through the selection process described next. Geometrically, we expect that the true location will be located around the region where solutions form a cluster (i.e., half of the circle intersections should ideally intersect at the solution). Let us to consider the example shown in Figure 2.2. There are three anchors named a_j, a_k and a_ℓ and a node s_i that needs to be localized. The range estimate R_{ij} is larger than d_{ij}, the range estimate R_{ik} is shorter than d_{ik}, and the range estimate $R_{i\ell}$ is shorter than $d_{i\ell}$. Hence, s_i computes a set of of six location candidates given by $\{\mathbf{g}_i^{jk}, \overline{\mathbf{g}}_i^{jk}, \mathbf{g}_i^{j\ell}, \overline{\mathbf{g}}_i^{j\ell}, \mathbf{g}_i^{k\ell}, \overline{\mathbf{g}}_i^{k\ell}\}$. As seen in the figure, all the mirror circle intersection estimates will tend to be isolated while the correct circle intersections will tend to cluster around the node location. For example, to decide between $\mathbf{g}_i^{jk}$ and $\overline{\mathbf{g}}_i^{jk}$ candidate positions, generated using the anchors $\left(a_j, a_k\right)$, the sensor s_i obtains the minimum Square Euclidean sum

from the location $\mathbf{g}_i^{jk}$ to each pair of cantidate positions as follows:

$$\psi = \min\left(\left\|\mathbf{g}_i^{jk} - \mathbf{g}_i^{j\ell}\right\|^2, \left\|\mathbf{g}_i^{jk} - \overline{\mathbf{g}}_i^{j\ell}\right\|^2\right) + \min\left(\left\|\mathbf{g}_i^{jk} - \mathbf{g}_i^{k\ell}\right\|^2, \left\|\mathbf{g}_i^{jk} - \overline{\mathbf{g}}_i^{k\ell}\right\|^2\right). \tag{2.28}$$

On the other hand, the sensor s_i also obtains the minimum Square Euclidean sum from the location $\overline{\mathbf{g}}_i^{jk}$ to each pair of candidate positions as follows:

$$\varphi = \min\left(\left\|\overline{\mathbf{g}}_i^{jk} - \mathbf{g}_i^{j\ell}\right\|^2, \left\|\overline{\mathbf{g}}_i^{jk} - \overline{\mathbf{g}}_i^{j\ell}\right\|^2\right) + \min\left(\left\|\overline{\mathbf{g}}_i^{jk} - \mathbf{g}_i^{k\ell}\right\|^2, \left\|\overline{\mathbf{g}}_i^{jk} - \overline{\mathbf{g}}_i^{k\ell}\right\|^2\right). \tag{2.29}$$

Finally, the lowest value of ψ and φ helps to decide between choosing $\mathbf{g}_i^{jk}$ or $\overline{\mathbf{g}}_i^{jk}$. The process is repeated for all Q solution pairs to generate a set of disambiguated locations.

Referring to our example, once node s_i removes the mirror locations, then an estimate of the node position can be formed by taking the *average* of the disambiguated set $\mathbf{G} = \{\mathbf{g}_i^{jk}, \mathbf{g}_i^{j\ell}, \mathbf{g}_i^{k\ell}\}$. For future discussion, we denote this initial localization estimate for s_i as $\mathbf{p}_i^0$; more generally

$$\mathbf{p}_i^0 = \frac{1}{Q}\sum_{\mathbf{g}\in\mathbf{G}} \mathbf{g}_i^{jk}. \tag{2.30}$$

The complete bilateration scheme is described in Algorithm 2. This is a distributed localization algorithm in the sense that each node can implement Algorithm 2 and determine its position estimate, given the anchor positions and the range estimates R_{ij} between each node and all the anchors. Given the limitation of using only anchor measurement, this algorithm can be used as an initialization step to generate a set of positions that can be used with algorithms that integrate more information from all (anchor and non-anchor nodes). An iterative distributed algorithm that benefits from this scheme will be presented in the next chapter.

There are some anomalous cases which should be considered in the proposed bilateration algorithm. In order to get its initial estimation $\mathbf{p}_i^0$, it is essential that every sensor s_i gets the two location estimations from each one of the Q subsets even if the solutions are not feasible. For example, assume the two special cases shown in Figure 2.3. If we consider the left-side case

Algorithm 2 General code used by every sensor s_i to get its initial position estimate $\mathbf{p}_i^0$

(The algorithm omits the special cases where there are no circle intersections. The procedure for these instances is described further in the text).

Require: $\mathbf{q}_k, R_{ik}, with \{k \leftarrow 1,\ldots,M\}$, and $Q \leftarrow \binom{M}{2}$
Ensure: $\mathbf{p}_i^0$
1: **Initialize:** $\mathbf{T} \leftarrow [0,0]^T$
2: **for** each subset $\mathbf{A}_{jk} \in \binom{M}{2}$ two-anchor subsets **do**
3: $\psi \leftarrow 0$
4: $\varphi \leftarrow 0$
5: $\left(\mathbf{g}_i^{jk}, \overline{\mathbf{g}}_i^{jk}\right) \leftarrow CCI\left(\mathbf{q}_j, \mathbf{q}_k, R_{ij}, R_{ik}\right)$ {Return the two circle intersections}
6: **for** each subset $\mathbf{A}_{\ell m} \neq \mathbf{A}_{jk} \in \binom{M}{2}$ two-anchor subsets **do**
7: $\left(\mathbf{g}_i^{\ell m}, \overline{\mathbf{g}}_i^{\ell m}\right) \leftarrow CCI\left(\mathbf{q}_\ell, \mathbf{q}_m, R_{i\ell}, R_{im}\right)$ {Return the two circle intersections}
8: $v_1 \leftarrow \left\|\mathbf{g}_i^{jk} - \mathbf{g}_i^{\ell m}\right\|^2$
9: $v_2 \leftarrow \left\|\mathbf{g}_i^{jk} - \overline{\mathbf{g}}_i^{\ell m}\right\|^2$
10: $\psi \leftarrow \psi + \min\left(v_1, v_2\right)$ {Return the minimum between v_1 and v_2}
11: $w_1 \leftarrow \left\|\overline{\mathbf{g}}_i^{jk} - \mathbf{g}_i^{\ell m}\right\|^2$
12: $w_2 \leftarrow \left\|\overline{\mathbf{g}}_i^{jk} - \overline{\mathbf{g}}_i^{\ell m}\right\|^2$
13: $\varphi \leftarrow \varphi + \min\left(w_1, w_2\right)$ {Return the minimum between w_1 and w_2}
14: **end for**
15: **if** $(\psi < \varphi)$ **then**
16: $\mathbf{T} \leftarrow \mathbf{T} + \mathbf{g}_i^{jk}$
17: **else**
18: $\mathbf{T} \leftarrow \mathbf{T} + \overline{\mathbf{g}}_i^{jk}$
19: **end if**
20: **end for**
21: $\mathbf{p}_i^0 \leftarrow \frac{\mathbf{T}}{Q}$

on the figure, R_{ik} is shorter than d_{ik}, and R_{ij} is shorter than d_{ij}, we can realize that the triangle inequalities are not satisfied:

$$\begin{aligned} R_{ij} + R_{ik} &> \left\|\mathbf{q}_j - \mathbf{q}_k\right\| \\ R_{ij} + \left\|\mathbf{q}_j - \mathbf{q}_k\right\| &> R_{ik} \\ R_{ik} + \left\|\mathbf{q}_j - \mathbf{q}_k\right\| &> R_{ij} \end{aligned} \tag{2.31}$$

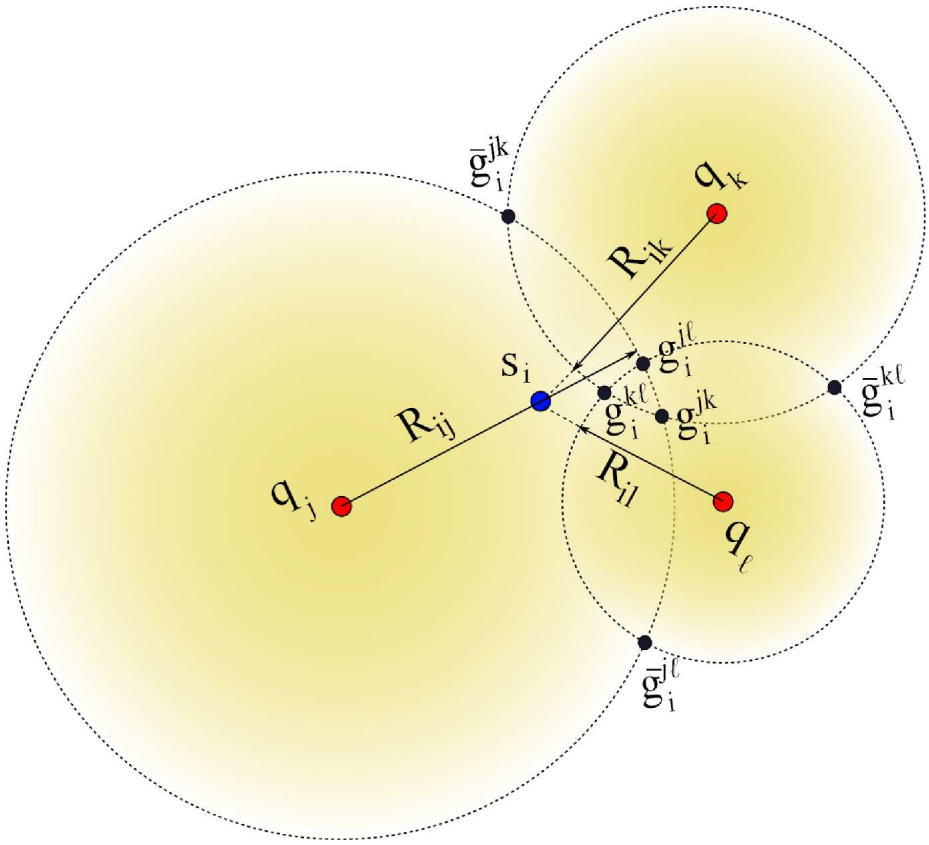
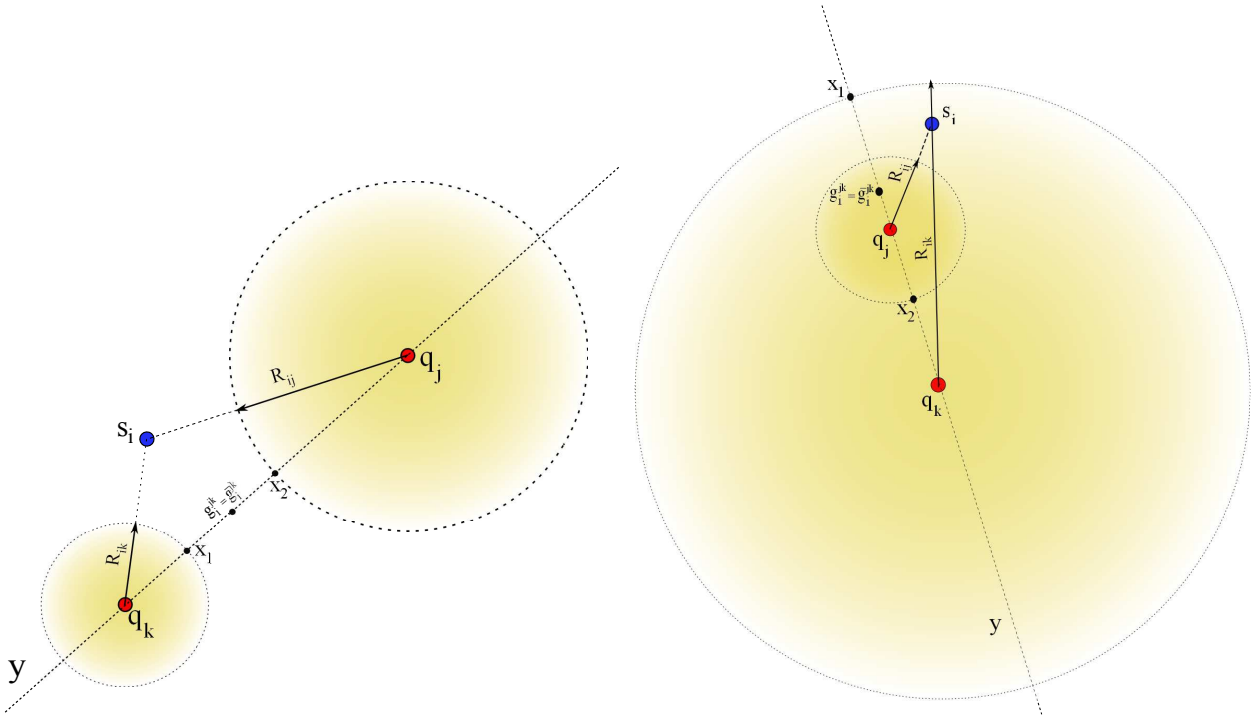


Figure 2.2: Sensor s_i getting its initial estimation P_i^0 from three anchors $\left(a_j, a_k, a_\ell\right)$.

As a consequence, the sensor s_i will not be able to find any solution.

In other words, if the two circles are not in touch, it will not be feasible to find the circle intersections $\mathbf{g}_i$ and $\overline{\mathbf{g}}_i$. Therefore, a relaxed estimation should be generated as described next. Considering that $\left\|\mathbf{q}_j - \mathbf{q}_k\right\|$ is a constant distance between the anchors in set $\mathbf{A}_{jk}$, the node s_i takes two steps to estimate the locations $\mathbf{g}_i^{jk}$ and $\overline{\mathbf{g}}_i^{jk}$. First, a location $\mathbf{x}_1$ is obtained by fixing R_{ik} and making $R_{ij} = \left|\left\|\mathbf{q}_j - \mathbf{q}_k\right\| - R_{ik}\right|$ to satisfy the triangle inequality. Next, the sensor s_i should use the *CCI* procedure to solve for $\mathbf{x}_1$. Similarly, a second location estimate $\mathbf{x}_2$ is obtained by fixing R_{ij}, choosing $R_{ik} = \left|\left\|\mathbf{q}_j - \mathbf{q}_k\right\| - R_{ij}\right|$ to satisfy the triangle inequality and solving the problem through the *CCI* procedure. Finally, both $\mathbf{g}_i^{jk}$ and $\overline{\mathbf{g}}_i^{jk}$ are generated as the average $\frac{\mathbf{x}_1+\mathbf{x}_2}{2}$ implying that when the triangle inequality is not satisfied, there will be a single solution that fall over the line y. A similar procedure can be derived for the second case as depicted in Figure 2.3.



(a) Case A.

(b) Case B

Figure 2.3: Possible cases where the triangle inequality is not satisfied.

2.3.1 Comparison with Previous Bilateration Scheme

As described before, the research reported in [39] is focused on a distributed bilateration scheme that finds initial estimates. Using two anchors at time each sensor node s_i finds two possible candidates (i.e., circle intersections). If sufficient anchors are available, the sensor node s_i averages the cloud of candidates which tend to be close to each other. The average of such candidates provides the initial estimate.

As can be seen the general idea for this approach is quite similar to our proposed approach. However, there are remarkable differences between two the schemes that should be taken into account. These differences make that our bilateration approach be more robust and less expensive in wireless transmissions than the scheme proposed in [39]. For instance, one of the differences between our bilateration algorithm and the proposed in [39] is that the last one does not take into account special cases when a sensor s_i is not able to compute circle intersections of two anchors (i.e., the circles are not in touch) as shown Figure 2.3. Therefore, under this perspective this scheme is limited to naive scenarios in which estimated distances between sensors and anchors should have good accuracy. Thus, RSS measurements, commonly used in realistic scenarios, may not provide useful information for this scheme given the noisy measurements. Hence, if a sensor s_i is not able to find sufficient circle intersections from two neighboring pair of anchors at time, the localization process will fail. In our case, our proposed bilateration scheme is able to obtain initial estimates in spite of the most severe scenarios (i.e., not circle intersections).

Another important aspect to consider in [39] is the use of a threshold δ which is used to reduce the number of possible candidate positions, making this approach more selective. However, the value of δ value is hard to determine in practice, and also it does not guarantee good results in noisy environments. Finally, each sensor node s_i should create a table of its neighboring anchors. All anchors have a specific position inside of the table, and they are weighted by the sensor s_i according to the candidate positions that they generate. The value of δ is used to select a certain group of candidate positions. The anchors are weighted according to the candidates that they generated. Finally, all tables are broadcast by sensors. Once all sensors have received the

anchor tables of its neighbors, they run a post-processing stage to determine which anchors are more reliable than others. These anchors are used to obtain initial estimates. As can be seen, the drawback of this approach are extra wireless transmissions required to share anchor tables among sensors. In our case we avoid any kind of wireless transmission with the goal to save energy. Finally, we should remark that we are using a sorting algorithm to determine initial positions. Analysis results shown in next section demonstrate that our initialization algorithm is competitive in comparison with well known accurate and efficient algorithms based on least-squares methodologies.

2.4 Accuracy Performance Between Closed-formulas and Iterative Procedures in the WSN Localization Problem

In this section we analyze the accuracy performance of both methodologies described before. Even though the strength of a closed-formula for solving the WSN localization problem is its low complexity compared with an iterative algorithm, closed-formulas can present large errors on the initial estimates in presence of inaccurate ranging measurements. However, in many cases is desirable to sacrifice accuracy to save energy (i.e., increase battery lifetime). On the other hand, the weakness for direct mehods (i.e., noise sensitivity) represents the strong point for iterative methods and viceversa so the goal of both methodologies seems to be in opposite directions. However, the main effort in WSN localization research is focused on developing an strategy that can join the strength of both methodologies to create an efficient algorithm that can save energy providing the best accuracy in the estimated positions.

Next we present an evaluation of accuracy between closed-formulas and iterative methodologies. For the former methods we are considering the classical LS Multilateration and our proposed bilateration algorithm. For iterative methodologies we are also considering two algorithms to solve the NLLS: the LM and the TRR algorithms.

For the simulations that follow, we consider 20 different sensor networks where each one is composed by $N = 100$ sensors, randomly distributed, in a 100m by 100m area. Also, we select four non-collinear anchors with full-connectivity on every realization. For each network, we generate a set of noisy ranges between anchors and nodes using the log-distance path loss model (1.4). Finally, the estimated distances are simulated using $\sigma_{dB} = 6$ and $\eta_P = 2.6$, typical parameters for the propagation models on outdoors scenarios. These propagation models are detailed on section 4.1.

To compare the accuracy performance between both methodologies it was necessary to use the same set of range measurements for each direct method and iterative algorithm. Figure 2.4 summarizes the initial estimates obtained by both methodologies using the RMSE metric

as shown the next equation:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N} \left\|\mathbf{p}_i^0 - \mathbf{z}_i\right\|}, \tag{2.32}$$

where $\mathbf{p}_i^0$ represents an initial position estimate for a sensor s_i and $\mathbf{z}_i$ its true position. As can be seen the closed-form LS approach provides the least accurate initial estimates (mean=22.7m and standard deviation=2.22m) compared with iterative algorithms as expected due to the noisy ranging measurements. Also, we can appreciate that both iterative algorithms, the LM and the TRR, provide practically the best and similar results for initial estimates (mean=12.54m and standard deviation=.69m) as expected, and finally our proposed bilateration algorithm presents very acceptable initial estimates compared with the last two algorithms (mean=12.96m standard deviation=.84m). However, we should consider that the computational complexity for the LM and the TRR algorithms is significantly larger than the bilateration algorithm. This will be expanded on the next section.

Also, we tested the GSLS algorithm [29] using the same set of networks. The estimated positions presented large errors under this scheme as indicated by [28]. Then, these results were disregarded in our analysis.

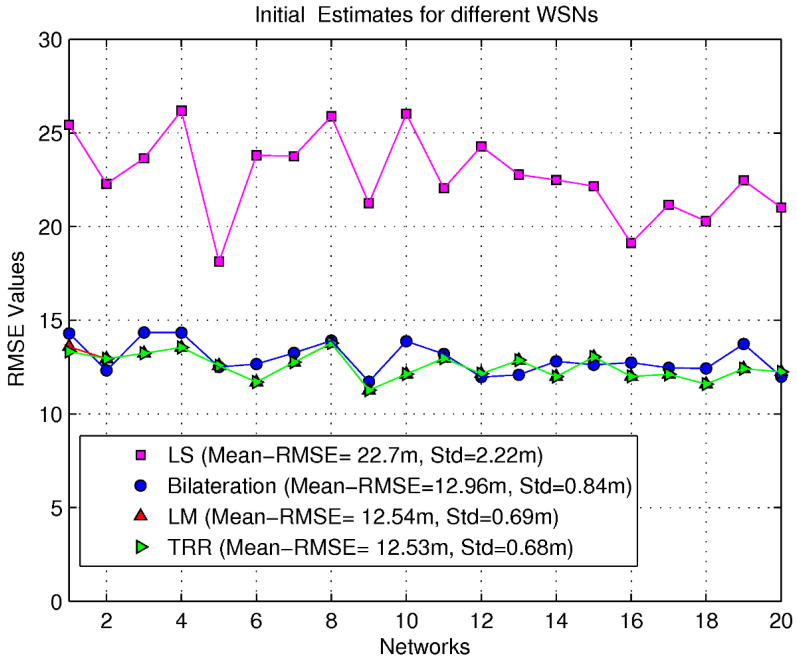


Figure 2.4: Algorithms used for initial estimations

2.5 Computational Complexity Analysis

The efficiency of an algorithm can be described in terms of the time or space complexity [16]. Time complexity refers to the relation between the number of required computations for a given input, and the space used for a given input provides the space complexity of an algorithm, so the computational complexity of an algorithm could be described as the number of operations that it takes to find a solution.

In this section we provide an operation count on the number of additions (ADDs), multipliers (MULs), divides (DIVs), and potentially (SQRTs) exactly in the way that DSP algorithms are described [27, 68]. This will allow an "apple-to-apple" comparison. Moreover, an accurate description lends itself to a cycle accurate description for any microprocessor and more significantly, the use of energy models based on computing cycles to estimate the energy consumption for a given algorithm. An energy analysis will be explored on Chapter 5.

Next we present the computational complexity analysis for the iterative LS and the bilateration algorithm.

2.5.1 Computational Analysis of the LM Algorithm

The LM algorithm could be considered as too expensive for motes given its iterative nature and the need to estimate first and second order information (i.e., gradients, Jacobians, and Hessians). The number of iterations K is highly dependant on the initial point and could be considered a random variable. On the other hand, if a good $\mathbf{x}_0$ is provided, then the number of iterations is expected to be low given the convergence properties of LM.

We are interested on providing an algorithmic analysis that provides a detailed description in terms of additions and subtractions (jointly referred as ADDs), multiplications (MULs), divisions (DIVs), and square roots (SQRTs). For simplicity in the next paragraphs consider that $\mathbf{J}_k \equiv \mathbf{J}(\mathbf{x}_k)$ and $\mathbf{R}_k \equiv \mathbf{R}(\mathbf{x}_k)$.

The square root is a relevant operation as the error function $\mathbf{R}_k$ and the Jacobian estimate requires ℓ_2 norms to compute distances between sensor and anchors. We also note that the complexity of the operations is not the same in terms of the processing resources (hardware and software) they take; abusing notation we have

$$\text{ADD} < \text{MUL} < \text{DIV} < \text{SQRT}. \tag{2.33}$$

This analysis also focuses on the most efficient implementation in terms of the proper operation sequencing in order to favor reuse of terms (i.e., avoid computing the same quantity twice).

We perform the analysis for a single iteration of the LM algorithm, and the total cost for each operation is multiplied by K. We also note that K can be modeled as a random variable; the usefulness of this approach is discussed later. We assume there are M anchors which have broadcast their position to all the nodes. Each node will run the LM algorithm to find its initial position as described before. We identify three core operations: ℓ_2 or Euclidean norm, the error vector $\mathbf{R}_k$ and an estimate of $\mathbf{J}_k$.

The ℓ_2 will be used to compute the magnitude of the difference between two vectors $\mathbf{a}, \mathbf{b} \in \mathbb{R}^2$ given by $\|\mathbf{a}-\mathbf{b}\| = \sqrt{(a_x-b_x)^2+(a_y-b_y)^2}$. This requires three ADDS, two MULs and one SQRT. The norm is used to compute $\mathbf{R}_k$ given in equation (2.17) and to estimate the Jacobian as

follows:

$$\mathbf{J}_k = \begin{bmatrix} \frac{-(x_1-x_k)}{\sqrt{(x_1-x_k)^2+(y_1-y_k)^2}} & \frac{-(y_1-y_k)}{\sqrt{(x_1-x_k)^2+(y_1-y_k)^2}} \\ \vdots & \vdots \\ \frac{-(x_M-x_k)}{\sqrt{(x_M-x_k)^2+(y_M-y_k)^2}} & \frac{-(y_M-y_k)}{\sqrt{(x_M-x_k)^2+(y_M-y_k)^2}} \end{bmatrix}_{Mx2} \tag{2.34}$$

For $\mathbf{R}_k$ we see that we require M ADDs and M ℓ_2-norms. Accounting for the norms, the error function requires $4M$ ADDs, $2M$ MULs, and M SQRTs. These numbers are recorder in Table 2.1. A similar analysis follows for $\mathbf{J}_k$. A direct look at equation (2.34) indicates that we have the same norm across rows, so we can compute them first and then we would need an additional $2M$ ADDs and $2M$ DIVs. However, a better approach would be to compute the terms $1/\|\mathbf{x}_j - \mathbf{x}_k\|$ first so that we would require M DIVs, $2M$ MULs, and $2M$ ADDs. We exchange M DIVs by $2M$ MULs under the typical case that MULs have a much lower complexity than DIVs, particularly for the case of floating point operations. The complexity for the Jacobian estimate is also shown on Table 2.1.

Once these two quantities have been evaluated, their use trickles down through the algorithm. The costs for the different steps or operations is presented in the remaining part of Table 2.1. We just make two more remarks on the algorithm complexity. First, note that the approximation to the Hessian matrix $\mathbf{J}_K^T\mathbf{J}_K$

$$\nabla^2 f(\mathbf{x}_k) = \mathbf{J}_k^T\mathbf{J}_k = \begin{bmatrix} \sum_{j=1}^{M}\left(\frac{(x_j-x_k)^2}{\left((x_j-x_k)^2+(y_k-y_j)^2\right)}\right) & \sum_{j=1}^{M}\left(\frac{(x_j-x_k)(y_j-y_k)}{\left((x_j-x_k)^2+(y_j-y_k)^2\right)}\right) \\ \sum_{j=1}^{M}\left(\frac{(x_j-x_k)(y_j-y_k)}{\left((x_j-x_k)^2+(y_j-y_k)^2\right)}\right) & \sum_{j=1}^{M}\left(\frac{(y_j-y_k)^2}{\left((x_j-x_k)^2+(y_k-y_j)^2\right)}\right) \end{bmatrix}_{2x2} \tag{2.35}$$

is of size 2×2 which makes its inversion trivial when computing the LM step Δ_{LM}.

$$\triangle_{LM} = \left(\mathbf{J}_k\mathbf{J}_k^T + \mu_k\mathbf{I}\right)^{-1}\mathbf{J}_k^T\mathbf{R}_k, \tag{2.36}$$

where the gradient of the function $\mathbf{J}_k^T\mathbf{R}_k$ is given by

$$\nabla f(\mathbf{x}_k) = \mathbf{J}_k^T \mathbf{R}_k = \begin{bmatrix} \sum_{j=1}^{M} \left(\frac{(x_j - x_k) \cdot \left(R_{kj} - \sqrt{(x_j - x_k)^2 + (y_j - y_k)^2} \right)}{\sqrt{(x_j - x_k)^2 + (y_j - y_k)^2}} \right) \\ \sum_{j=1}^{M} \left(\frac{(y_j - y_k) \cdot \left(R_{kj} - \sqrt{(x_j - x_k)^2 + (y_j - y_k)^2} \right)}{\sqrt{(x_j - x_k)^2 + (y_j - y_k)^2}} \right) \end{bmatrix}_{2x1} \tag{2.37}$$

Second, satisfying the sufficient decrease condition is also an iterative procedure where different values of α_k are tested. We identify T as the number of iterations needed to satisfy this condition. As we discuss later, we will model T as a random variable.

Table 2.1: LM Cost Functions

	ADD	MUL	DIV	SQRT
$\mathbf{R}_k$	$4M$	$2M$	0	M
$\mathbf{J}_k$	$5M$	$4M$	M	M
$\mathbf{H}_k$	$3M-3$	$3M$	0	0
M'_f	$2M-2$	$2M$	0	0
M_f	$M-1$	$M+1$	0	0
μ_k	3	3	0	1
$\mathbf{H}_k^{-1}$	3	6	1	0
$\triangle_{LM}$	2	4	0	0
Sufficient Decrease	$T(M+4)$	$T(M+2)$	0	0
Update	2	2	0	2
Stopping Condition	3	2	0	1
Total	$(M+4)T+15M+7$	$(M+2)T+12M+18$	$M+1$	$2M+4$

The last row of the table provides the total which we identify as T_{ADD}, T_{MUL}, T_{DIV} and T_{SQRT} respectively. These numbers are the operations for a single iteration of the LM algorithm. Then, for K iterations we have the total number of operations to be

$$K_{ADD} = K \cdot T_{ADD}, \tag{2.38}$$

$$K_{MUL} = K \cdot T_{MUL}, \tag{2.39}$$

$$K_{DIV} = K \cdot T_{DIV}, \tag{2.40}$$

$$K_{SQRT} = K \cdot T_{SQRT}. \tag{2.41}$$

Since the values of T and K are random variables, then a more convenient approach to quantify the number of operations would be to look at the average number of operations, i.e., the expected value. It is intuitive to assume that T and K are independent, and that for a given network their distributions will be identical. Hence, we define

$$\overline{K}_{ADD} = \varepsilon\{K_{ADD}\} = \varepsilon\{K\}\varepsilon\{T_{ADD}\} = \varepsilon\{K\}(\varepsilon\{T\}(M+4)+15M+7) \tag{2.42}$$

$$\overline{K}_{MUL} = \varepsilon\{K_{MUL}\} = \varepsilon\{K\}\varepsilon\{T_{MUL}\} = \varepsilon\{K\}(\varepsilon\{T\}(M+2)+10M+15) \tag{2.43}$$

$$\overline{K}_{DIV} = \varepsilon\{K_{DIV}\} = \varepsilon\{K\}(M+1) \tag{2.44}$$

$$\overline{K}_{SQRT} = \varepsilon\{K_{SQRT}\} = \varepsilon\{K\}(2M+2) \tag{2.45}$$

where $\varepsilon\{\chi\}$ represents the expected value of the random variable χ. Finally, we can quantify the total complexity of the LM algorithm by converting operations to a common denominator and compute a single representative number that can be used for comparison with other algorithms. The typical way to quantify operations is to use the number of processor cycles (on the average) required to complete each type of operation. Let us define N_{ADD}, N_{MUL}, N_{DIV}, and N_{SQRT} as the number of cycles required for floating addition (or subtraction), a multiplication, a division, and square root, respectively. We should note that these numbers depend on the processor used by the mote and the compiler tools used to develop the software. Hence, in practice the best way to obtain these values is through code profiling using a cycle-accurate processor specific simulator. Moreover, as we will see in Chapter 5, the number of task cycles can be used as part of models that measure energy consumption.

Hence, as a final measure of complexity for the LM algorithm we compute the total number of cycles as

$$N_{LM} = N_{ADD}\overline{K}_{ADD} + N_{MUL}\overline{K}_{MUL} + N_{DIV}\overline{K}_{DIV} + N_{SQRT}\overline{K}_{SQRT}. \tag{2.46}$$

2.5.2 Computational Analysis of the Bilateration Algorithm

The proposed bilateration algorithm is very simple and non-iterative. For M anchors, a sensor node picks $\binom{M}{2}$ pairs of sensors and computes the intersections of the imaginary circles around each anchor with a radius given between the anchor and the sensor node. These intersections are computed using geometry with a procedure described by equations (2.23) to (2.27). Then, a cluster with half of the computed intersections is found, providing an indication of the area where the node position is located. The number of operations required to compute two intersections is presented in Table 2.2.

Table 2.2: Bilateration Cost Operations

Operations	ADD	MUL	DIV	SQRT
d	3	2	0	1
d_{jt}	1	5	1	0
h	1	1	0	1
$r = \frac{h}{d}$	0	0	1	0
$\mathbf{f}_t$	2	2	1	0
$\mathbf{x}_i$	1	1	0	0
$\overline{\mathbf{x}}_i$	1	0	0	0
$\mathbf{y}_i$	1	1	0	0
$\overline{\mathbf{y}}_i$	1	0	0	0
Total (2 circle intersections)	11	12	3	2
Total Q node combinations	$11Q$	$12Q$	$3Q$	$2Q$

Since this process is repeated Q= $\binom{M}{2}$ times, then the final row reflects the total operations multiplied by this factor. As the intersections are computed, the search for the cluster is performed by equations (2.28) and (2.29). Since there are $2Q$ intersections, we need to select the Q that cluster together (i.e., eliminate mirrors). The clustering is based on looking at the distance between all possible pairs of intersections and selecting those that exhibit the closets distances among themselves. This requires the calculation of $S = \frac{2Q(2Q-1)}{2}$ squared norms, and the use of a clustering or sorting algorithm to find the smallest Q elements from the list of S norm values. Taking advantage of the structure of the location points (i.e., the two intersections from the same anchor pair are not compared), we can expect an average complexity of $O(S)$ sorting steps

using an algorithm like Quickselect algorithm [65]. Hence the final computational cost for the bilateration algorithm is presented in Table 2.3.

Table 2.3: Final Computational Cost for the Bilateration Scheme

Action	ADD	MUL	DIV	SQRT	SORT
Circle Intersections	$11Q$	$12Q$	$3Q$	$2Q$	0
Squared Norms	3S	2S	0	0	0
Number of Comparisons	0	0	0	0	$O(S)$

As with the LM algorithm, we close this section by providing an expression in terms of processor cycles. Using the same characterization for all main operations of the algorithm, we can provide a total cycle count that can be directly compared with other algorithms. Obviously, a lower cycle implies lower complexity when the hardware and software development tools are identical. The expression for total cycles is

$$N_{BL} = N_{ADD} \cdot (11Q + 3S) + N_{MUL} \cdot (12Q + 2S) + N_{DIV} \cdot (3Q) + N_{SQRT} \cdot (2Q) + N_{SORT}. \quad (2.47)$$

It is easy to see that the bilateration scheme uses a significantly less number of cycle for all operations. In particular, for $M = 4$, we have a low count of DIV and SQRT operations. Experimental data in [65] indicates that the cycle count for the complete sorting step with the *QuickSelect* algorithm with a pipelined architecture can be achieved with 2500 to 3000 cycles.

Chapter 3

Distributed Localization Algorithms based on Local and Sub-local Problems with Spatial Search Constraints

With the rapid evolution of the computer technology, iterative algorithms have been extensively used in computational mathematics and optimization areas. Basically an iterative algorithm repeats a procedure using old outputs and/or new data as feedback to generate successive approximations that gradually tend to converge to a solution or the best solution to a problem when no exact solution exists. As described before, the formulation to solve the WSN localization results in a non-linear NP-Hard problem. Hence, iterative algorithms represent the best option to solve such kind of problem. In the next section we will present two iterative distributed algorithms that can be used to solve the WSN localization problem.

3.1 An Overview of Optimization-Based Localization

To describe formally our proposed distributed localization algorithms, first we need to describe requirements and general procedures that share both algorithms. We require for every sensor node s_i to know the anchor positions, the range estimates (r_{ij} and R_{ik}) to each one of its neighbors, and also the initial position estimates $\mathbf{p}_k^0$ for all k. The initial estimates can be obtained with the algorithms described in the previous chapter or any other algorithm deemed appropriate. In addition, due to their iterative nature each node updates its current position $\mathbf{p}_i^\ell$ to $\mathbf{p}_i^{\ell+1}$ using the most recent neighbor position estimates $\mathbf{p}_j^\ell$ with $j \neq i$. This implies that iterative algorithms

require wireless transmissions of the node position estimates after they have been updated. Finally, we assume the nodes are equipped with the communication protocols needed to share this information between nodes [73]. We should note of relying this information across the network is by no means a simple task. A communication protocol to coordinate and schedule these tasks needs to be designed as part of a localization service, perhaps as part of a cross-layer design if the ranging operations are done in hardware [73]. However, our focus on this research is on the localization schemes rather than protocol design.

3.2 Localization from a Global Optimization Perspective

As discussed before, node localization can be possed as an optimization problem where the set of unknown positions $\mathbf{Z} = \{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_N\}$ are estimated by finding the optimal set $\mathbf{L} = \{\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_N\}$ that minimizes a cost function C that captures the geometrical configuration of the network. A possible global cost function can be defined as follows:

$$C(\mathbf{p}_1, \ldots, \mathbf{p}_N) = \sum_i \sum_j \left| r_{ij} - \left\| \mathbf{p}_i - \mathbf{p}_j \right\| \right| + \sum_i \sum_k \left| R_{ik} - \left\| \mathbf{p}_i - \mathbf{q}_k \right\| \right|. \tag{3.1}$$

Each term in the summations represents the absolute difference between the measured range and the Euclidean distance between two candidate positions $\mathbf{p}_i$ and $\mathbf{p}_j$, or a candidate position $\mathbf{p}_i$ and an anchor $\mathbf{q}_k$. This is the most general form where channel asymmetry ($r_{ij} \neq r_{ji}$ and $R_{ik} \neq R_{ki}$) is assumed. The cost function assumes a fully connected network. An intuitive explanation to this function is to imagine the network as a set of nodes connected by springs. The measured ranges represent measured spring forces. In the noiseless cases, these forces are in equilibrium and would provide the correct location of the nodes. On the other hand, noisy measurements require the search of a set of optimal positions in the sense of approaching force equilibrium as best as possible. This model has been used in some localization algorithms to develop "push-pull" force vector methods [38].

Trying to find the set $\mathbf{L}$ that globally minimizes this function is difficult and sometimes be-

comes an intractable problem. A global minimum is not commonly reached because many local minima are generated by noisy range measurements [13, 47, 69, 9, 6, 3, 21, 26]. Typical approaches to address this issue are 1) to find a sequential solution where each position is updated at a time, or 2) to split the problem into a set of sub problems which are solved iteratively with the goal that the process will converge to a solution [13, 69, 9, 26].

Let us define

$$\alpha_{ij}(\mathbf{p}_i) = \left| r_{ij} - \left\| \mathbf{p}_i - \mathbf{p}_j \right\| \right| \tag{3.2}$$

as the range error between the sensor s_i and the sensor s_j, and

$$\beta_{ik}(\mathbf{p}_i) = \left| R_{ik} - \left\| \mathbf{p}_i - \mathbf{q}_k \right\| \right| \tag{3.3}$$

as the range error between the sensor s_i and the anchor a_k. It is possible to rewrite (3.1) as

$$C(\mathbf{p}_1, \ldots, \mathbf{p}_N) = \sum_{i=1}^{N} F(\mathbf{p}_i) \tag{3.4}$$

where

$$F(\mathbf{p}_i) = \sum_{\substack{j=1 \\ j \neq i}}^{N} \alpha_{ij}(\mathbf{p}_i) + \sum_{k=1}^{M} \beta_{ik}(\mathbf{p}_i) \tag{3.5}$$

represents the total range error between s_i and all its neighboring nodes and anchors. Expressing the cost function with this equation provides us with some insight on the possibility of producing a distributed algorithm. Given that each sensor s_i has partial or complete knowledge of a set of positions $\mathbf{L}$, then the evaluation of the function $F(\mathbf{x})$ could be done locally at each sensor. This however, does not change the complexity of the problem since we are still trying to solve the problem globally. Moreover, the complexity of the problem is increased by the necessary communication overhead. In the next section we describe our first proposed distributed local approach for localization.

3.3 A Distributed Spatially Constrained Localization Scheme Based on the Solutions of Local Problems (DSCL-L)

In this section we consider a set of *local* problems where each node finds its position $\mathbf{p}_i$ by minimizing $F(\mathbf{x})$ with respect to the other positions. Obviously, since these positions are not known, we can suggest an iterative approach:

$$\mathbf{p}_i^{\ell+1} = \arg\min_{\mathbf{x}} \mathrm{F}(\mathbf{x}) = \sum_{\substack{j=1 \\ j\neq i}}^{N} \alpha_{ij}^{\ell}(\mathbf{x}) + \sum_{k=1}^{M} \beta_{ik}^{\ell}(\mathbf{x}) \tag{3.6}$$

where $\alpha_{ij}^{\ell}(\mathbf{x})$ and $\beta_{ik}^{\ell}(\mathbf{x})$ are given by equations (3.2) and (3.3), respectively, considering the iteration, ℓ, of the refining algorithm. We need to assume an initial set of positions, $\mathbf{L}^0$, obtained by an initialization algorithm (like our proposed bilateration scheme).

We want to constrain the problem (3.6) such that at each iteration the updated positions have short movements along the correct direction. This would allow all sensors to move "collaboratively" across iterations so that each sensor s_i can progressively adapt to the position updates from the other sensors to gradually reduce the margin of error given by α_{ij}^{ℓ} and β_{ik}^{ℓ}.

As a way to resolve these issues, we define a non-linear programming problem for each node s_i where its position at iteration ℓ is given by

$$\mathbf{p}_i^{\ell+1} = \arg\min_{\mathbf{x}} \left(\sum_{\substack{j=1 \\ j\neq i}}^{N} \alpha_{ij}^{\ell}(\mathbf{x}) + \sum_{k=1}^{M} \beta_{ik}^{\ell}(\mathbf{x}) \right) \tag{3.7a}$$

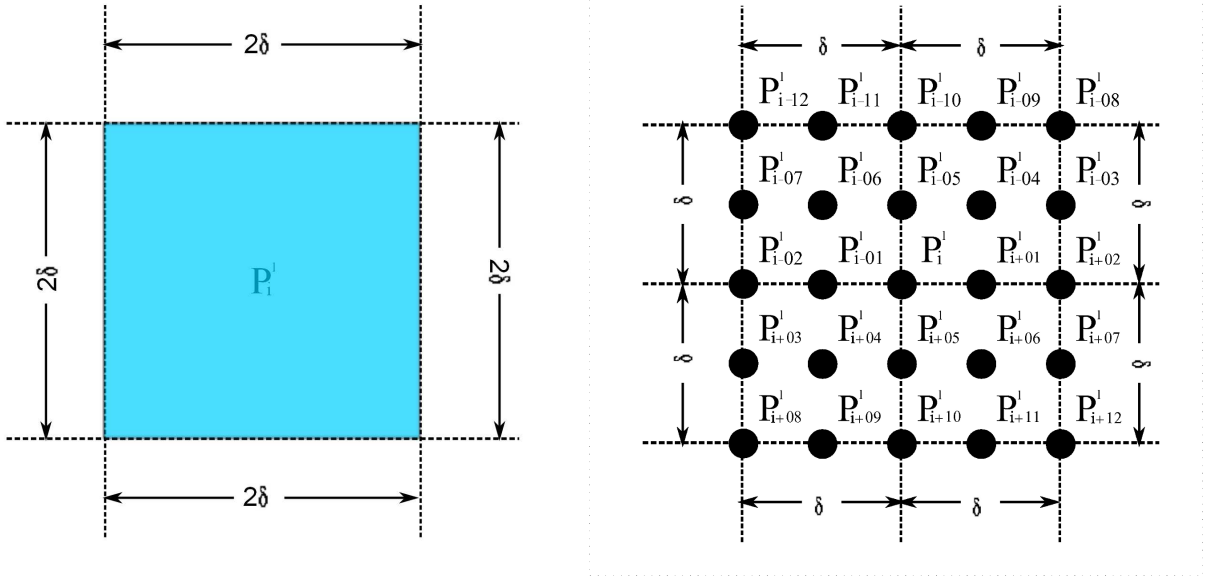
subject to

$$p_{x_i}^{\ell} - \delta^{\ell} \leq x_i \leq p_{x_i}^{\ell} + \delta^{\ell} \tag{3.7b}$$

$$p_{y_i}^{\ell} - \delta^{\ell} \leq y_i \leq p_{y_i}^{\ell} + \delta^{\ell} \tag{3.7c}$$

where p_x and p_y denote the cartesian components of $\mathbf{p}_i^{\ell}$, $\mathbf{x} = [x_i, y_i]^T$, and δ^{ℓ} is an appropriately chosen parameter. These constraints are geometrical in the sense that they delimit the solution to

a 2D search region. We identify the search region as Ω_i^ℓ which is $2\delta \times 2\delta$ box centered at $\mathbf{p}_i^\ell$ as depicted in Figure 3.1a.



(a) Constrained Continuous-Search Area in 2-D. (b) Constrained Discrete-Search Area in 2-D.

Figure 3.1: Area centered at position $\mathbf{p}_i^\ell$.

The value for δ^ℓ can be fixed or adapted across iterations. We found the following heuristic value to be useful:

$$\delta^\ell = \frac{\mathrm{F}(\mathbf{p}_i^\ell)}{\sum\limits_{\substack{i=1 \\ i \neq j}}^{N} r_{ij} + \sum\limits_{k=1}^{M} R_{ik}} \tag{3.8}$$

We note that it depends on the measurement errors and range measurements. Hence, as the iterative process converges to an optimal set of locations, we expect the value of δ^ℓ decreases at each iteration, further constraining the search region. The validity of using these constraints depends on having some level of reliability on the range measurements; otherwise the use of RSS, ToA (or some other ranging scheme) would be of little value to the localization problem.

The program described in (3.7a), (3.7b) and (3.7c) is a non-linear programming problem that could be solved with an interior point method [48]. For a sensor node, the implementation of such

methods may be prohibitive given the computational and storage limitations of the hardware.

We propose the following discretization approach. We discretize each Ω_i^ℓ and consider only a set of 25 candidate solutions $\mathbf{P}_c^\ell$ for $(-12 \leq c \leq 12)$ which are identified graphically in Figure 3.1b. The value at $c = 0$ corresponds to $\mathbf{p}_i^\ell$. Hence, a position update $\mathbf{p}_i^{\ell+1}$ can be easily obtained by minimizing (3.7a) over the candidate set using direct substitution. In conclusion, the distributed spatially constraint localization using local problems (DSCL-L) can be summarized as follows:

1. Assume a set of initial locations $\mathbf{L}^0$ obtained with an algorithm like the bilateration scheme discussed in this paper.

2. Repeat steps 3 and 4 for $\ell = 1, 2, \ldots$ until convergence.

3. Each sensor computes δ^ℓ and solves the program (3.7a), (3.7b), and (3.7c).

4. The set of position updates $\mathbf{L}^{\ell+1} = \{\mathbf{p}_i^{\ell+1} | i = 1, \ldots, N\}$ is broadcast through the network.

One additional benefit of solving local problems is that each sensor could solve (3.7a) individually to produce the set $\mathbf{L}^{\ell+1} = \{\mathbf{p}_i^{\ell+1} | i = 1, \ldots, N\}$ on a distributed way. Each node can transmit its updated value to all neighbors in order to perform the next iteration. Algorithm 3 shows the proposed DSCL-L scheme.

Algorithm 3 Sensor s_i refining its current position $\mathbf{p}_i^{\ell}$ using the DSCL-L approach

Require: ε_i, r_{ij}, with $j \neq i \{j \leftarrow 1,\ldots,N\}$
Ensure: $\mathbf{p}_i^{\ell+n}$

Initialize: $\ell \leftarrow 0$
repeat
  for j←1 **to** N-1 **do**
    $\mathbf{p}_j^{\ell} \leftarrow$ Receive_from (s_j)
  end for
  $\delta^{\ell} \leftarrow \frac{\mathrm{F}(\mathbf{p}_i^{\ell})}{\sum\limits_{\substack{i=1 \\ i\neq j}}^{N} r_{ij} + \sum\limits_{k=1}^{M} R_{ik}}$
  $\mathbf{p}_i^{\ell+1} = \arg\min\limits_{\mathbf{x}} \left(\sum\limits_{\substack{j=1 \\ j\neq i}}^{N} \alpha_{ij}^{\ell}(\mathbf{x}) + \sum\limits_{k=1}^{M} \beta_{ik}^{\ell}(\mathbf{x}) \right)$
  subject to :
  $p_{x_i}^{\ell} - \delta^{\ell} \leq x_i \leq p_{x_i}^{\ell} + \delta^{\ell}$
  $p_{y_i}^{\ell} - \delta^{\ell} \leq y_i \leq p_{y_i}^{\ell} + \delta^{\ell}$
  Broadcast$\left(\mathbf{p}_i^{\ell+1}\right)$
  $\ell \leftarrow \ell + 1$
until $\ell \leq 100$ **or** $\left\| \mathbf{p}_i^{\ell} - \mathbf{p}_i^{\ell+1} \right\| < \varepsilon_i$

3.4 A Distributed Spatially Constrained Localization Scheme Based on the Solutions of Local Problems (DSCL-SL)

The DSCL-L approach introduced in the previous section allows an implementation on sensor nodes with limited computational resources. Some insight was gained during the development of this scheme that bring us to the second algorithm proposed on this research. As can be seen, every time (3.7a) is solved, we find that the point simultaneously minimizes $M+N-1$ range error terms. Since the range measurements r_{ij} and R_{ij} are random variables (i.e., they are noisy), we hypothesize that the solution will converge slowly to the correct location. If there are relatively large range errors, they will limit the ability of the optimization program to have a large update step towards a final position.

We propose to take the DSCL-L problem approach one step further. The scheme we present could be considered as a relaxation of the objective function in addition to the discretization on the constraints. We start by analyzing the objective function in equation (3.5) which computes the total local range error between s_i and its neighbors. Solving (3.7a), s_i will update its current position, $\mathbf{p}_i^\ell$, based on the most current neighbor sensor positions $\{\mathbf{p}_k^\ell \ | k = 1 \ldots N, i \neq k\}$ shared through wireless broadcasts.

Let us to start describing the approach using one neighbor sensor s_j. It is easy to see that equation (3.5) can be further split into a set of objectives of the form (3.2) (and (3.3) for the anchor case), so we can define a set of *sub-local* problems as follows:

$$\mathbf{p}_{ij}^{\ell+1} = \underset{\mathbf{x}\in\Omega_{ij}}{arg\,min} \left| \left\| \mathbf{x} - \mathbf{p}_j^\ell \right\| - r_{ij} \right|, \tag{3.9}$$

for $j = 1 \ldots N$ and $i \neq j$. In essence, we find the values of $\mathbf{p}_{ij}^{\ell+1}$ that minimize the pair-wise range errors between s_i and any of its neighbors s_j. We have also defined a search region Ω_{ij} that we visualize as a box centered at the current location $\mathbf{p}_i^\ell$ and as before we use a discretized search region Ω_{ij} as shown in Figure 3.1b.

The search regions Ω_{ij} are effectively a set of constraints. These constraints are geometrical

in the sense that they delimit the solutions to a 2D search region. We aim to avoid position updates $\mathbf{p}_{ij}^{\ell+1}$ that are too far from the current estimate $\mathbf{p}_i^{\ell}$. Without such constraints, each pairwise position update would be a greedy process not accounting for the other sub-local problems being solved. Hence, any r_{ij} with large measurement errors would tend to take $\mathbf{p}_{ij}^{\ell+1}$ away from the true location of s_i, introducing a position bias.

We define the search region to be a $2\delta \times 2\delta$ box centered at $\mathbf{p}_i^{\ell}$ as depicted in Figure 3.1b, where the value $\delta = \alpha_{ij}^{\ell}/r_{ij}$ is the range error normalized by r_{ij}. Similar values have been proposed to control the step size in the push-pull schemes [38].

Problem (3.9) is solved by simple substitution. Obviously this solution is not guaranteed to be the actual minimum on Ω_{ij}, but it is an approximation that can be computed in a predictable way using a resource constrained processor.

In a similar way to compute pairwise positions $\mathbf{p}_{ik}^{\ell+1}$ within s_i and the anchor nodes a_k, we can define a similar set of problems:

$$\mathbf{p}_{ik}^{\ell+1} = \underset{\mathbf{x}\in\Omega_{ik}}{arg\,min} \, |\|\mathbf{x}-\mathbf{q}_k\| - R_{ik}| \tag{3.10}$$

for $k = 1 \ldots M$. The search region is computed using $\delta = \beta_{ik}^{\ell}/R_{ik}$ and the same solving method is applied. The only difference in this case is that the anchor positions a_k do not change. In summary, each sensor s_i produces a set of pairwise position updates $\mathbf{p}_{ij}^{\ell+1}$ and $\mathbf{p}_{ik}^{\ell+1}$. Our final step is to find $\mathbf{p}_i^{\ell+1}$ from these position updates.

To find $\mathbf{p}_i^{\ell+1}$ we resource to a statistical argument. Intuitively, we expect that after each iteration, the solutions to (3.9) and (3.10) will close into the true sensor location. We expect these positions to form a cloud around the true position $\mathbf{z}_i$. We can define a zero mean random variable N_{ij} such that $\mathbf{p}_{ij}^{\ell+1} = \mathbf{z}_i + n_{ij}$. Hence, $\mathbf{z}_i = \varepsilon[Z]$ where $\varepsilon[\cdot]$ is the expectation operator. Hence, we can get an estimate $\hat{\mathbf{z}}_i$, or equivalently $\mathbf{p}_i^{\ell+1}$, using the sample mean of the pairwise position estimates; namely,

$$\mathbf{p}_i^{\ell+1} = \frac{1}{M+N-1}\left(\sum_{\substack{j=1 \\ j\neq i}}^{N} \mathbf{p}_{ij}^{\ell+1} + \sum_{k=1}^{M} \mathbf{p}_{ik}^{\ell+1}\right). \tag{3.11}$$

We have embedded an estimator as the last step of our localization procedure. This provides a potential benefit of exploring the use of other estimators and incorporating additional knowledge about the network with little effort. For instance, connectivity can change from iteration to iteration according to realistic channel conditions. In this case, the estimator can eliminate those sensors who did not provide an update position or reuse values from previous iterations in equation (3.11).

We summarize the DSCL-SL scheme in Algorithm 4. This algorithm can be deployed on each sensor, and only requires to receive and provide position updates at the end of each iteration. Effectively each node self-localizes and can decide when to stop updating its position according to some local criteria. This issue will be evaluated in detail in Chapter 5.

Algorithm 4 Sensor s_i refining its current position $\mathbf{p}_i^{\ell}$ using the DSCL-SL approach

Require: ε_i, r_{ij}, with $j \neq i \{ j \leftarrow 1, \ldots, N \}$

Ensure: $\mathbf{p}_i^{\ell+n}$

Initialize: $\ell \leftarrow 0$

repeat

  for j$\leftarrow$1 **to** N-1 **do**

    $\mathbf{p}_j^{\ell} \leftarrow Receive\_from(s_j)$

    $\delta \leftarrow \frac{\left| r_{ij} - \left\| \mathbf{p}_i^{\ell} - \mathbf{p}_j^{\ell} \right\| \right|}{r_{ij}}$

    $\mathbf{P}_i^{\ell} \leftarrow from\_picture(3.1)$

    $\mathbf{p}_{ij}^{\ell+1} \leftarrow \underset{\{\mathbf{P}_{i+s}^{\ell}\}}{argmin} \left| \left\| \mathbf{P}_{i+s}^{\ell} - \mathbf{p}_j^{\ell} \right\| - r_{ij} \right| \ (-12 \leq s \leq 12)$

  end for

  for k$\leftarrow$1 **to** M **do**

    $\delta \leftarrow \frac{\left| R_{ik} - \left\| \mathbf{p}_i^{\ell} - \mathbf{q}_k \right\| \right|}{R_{ik}}$

    $\mathbf{P}_i^{\ell} \leftarrow from\_picture(3.1)$

    $\mathbf{p}_{ij}^{\ell+1} \leftarrow \underset{\{\mathbf{P}_{i+s}^{\ell}\}}{argmin} \left| \left\| \mathbf{P}_{i+s}^{\ell} - \mathbf{q}_k \right\| - R_{ik} \right| \ (-12 \leq s \leq 12)$

  end for

  $\mathbf{p}_i^{\ell+1} \leftarrow \frac{1}{M+N-1} \left(\sum_{\substack{j=1 \\ j \neq i}}^{N} \mathbf{p}_{ij}^{\ell+1} + \sum_{k=1}^{M} \mathbf{p}_{ik}^{\ell+1} \right)$

  $Broadcast\left(\mathbf{p}_i^{\ell+1} \right)$

  $\ell \leftarrow \ell + 1$

until $\ell \leq 100$ **or** $\left\| \mathbf{p}_i^{\ell} - \mathbf{p}_i^{\ell+1} \right\| < \varepsilon_i$

Chapter 4

Evaluation of Localization Algorithms

It is well known that the accuracy performance of localization algorithms is closely related to factors like the number and location of anchors, the accuracy in range estimations, the coverage range of each sensor, the size of the area, and the iterative algorithm by itself. For this reason, this chapter presents a methodological evaluation to test the accuracy performance and robustness of our proposed localization algorithms. In section 4.1 we present two well known propagation models that are of importance for our experimental and simulated range-based schemes. In Section 4.2, we analyze the performance of our algorithms using a short-scale WSN testbed, a realistic testing environment, that provides ranging measurements based on ToA and RSS measurements. Section 4.3 validates the propagation models of section 4.1 that will be used in subsequent sections. Section 4.4 presents a detailed analysis based on real and synthetic dataset for the case where sensors have a limited radio frequency range. Finally, Section 4.5 presents the use of large areas and large deployments to evaluate the performance of our algorithms under extensive simulations based on large-scale WSN scenarios using ToA and RSS measurements.

4.1 Propagation Models

For our analysis framework we consider two well known models for the characterization of ToA and RSS range measurements. For ToA techniques, the true distance d_{ij} between the sensors s_i and s_j has a measurement error modeled as additive Gaussian noise such that

$$r_{ij} \sim \mathcal{N}(d_{ij}, \sigma_d^2) \tag{4.1}$$

where σ_d (in meters) is the effect of multi-path signals and additive noise, and remains constant over long distances [24]. Clearly the expected value $\varepsilon\left[r_{ij}\right] = d_{ij}$. The same model can be assumed for R_{ik}, the measured range between sensor s_i and anchor a_k.

For RSS, it is well known that signal power varies inversely to distance. In noiseless environments the power signal travelling from a sensor s_j to a sensor s_i can be measured according to the relation [56]

$$P_{ij} = P_0 \left(\frac{d_0}{d_{ij}}\right)^{\eta_p} \tag{4.2}$$

where the path-loss factor (η_p) depends directly on the environmental conditions. P_0 is the received power at the short reference distance of $d_0 = 1m$ from the transmitter. Also, P_0 can be computed by the Friis free space equation [56]. The log-distance path loss model

$$\overline{P}_L^{(ij)}(dB) = P_0 - 10\eta_p \log \frac{d_{ij}}{d_0}, \tag{4.3}$$

measures the average large-scale path loss between sensors s_i and s_j. The actual path-loss (in dB) is a normally distributed random variable:

$$P_L^{(ij)} \sim \mathcal{N}(\overline{P}_L^{(ij)}, \sigma_{SH}^2), \tag{4.4}$$

where σ_{SH} is given in dB and reflects the degradations on signal propagation due to reflection, refraction, diffraction, and shadowing. It can be seen that the linear measurements and distance estimates have a log-normal distribution with a multiplicative effect on the measurements. The noisy range measurement r_{ij} can be obtained from (4.3) and (4.4) as

$$r_{ij} = 10^{\frac{P_0 - P_L^{(ij)}}{10 \cdot \eta_p}}. \tag{4.5}$$

Finally the measured power $P_L^{(ik)}$ between a sensor s_i and an anchor a_k can be stated in a similar way.

4.2 Optimized Localization for a Real Indoor WSN Benchmark

In this section, we use the set of real network measurements reported in [50] to evaluate the proposed algorithms. This data set has been used in other works [31, 38], making it a good comparison reference. The data set presents a fully connected network of 44 sensors randomly deployed in an office environment within a $14m$ by $13m$ area. The data set consists of ToA and RSS measurements which can be used to estimate ranges among nodes. The ToA measurement errors are Gaussian with a standard deviation around $1.84m$. For RSS a log-normal model was assumed with an estimated standard deviation of $3.92dB$. Also, this WSN scenario uses four of the sensors as anchors located intentionally in the corners to avoid the collinear anchor problem (see [49, 52]).

We start by evaluating the proposed localization scheme assuming a fully connected network where each sensor can produce a range measurement with the other 43 nodes. This network configuration is feasible since the nodes are deployed over a small area. Our algorithms were applied to estimate the locations of 40 unknown sensors following similar procedures as in [49, 31, 38]. First, the bilateration procedure was applied to generate an initial set of locations. Using this initial point, the distributed algorithm under DSCL-L and DSCL-SL approaches was used to refine the positions over 100 iterations. We present the accuracy of these algorithms using the root mean square error (RMSE) as a performance metric. The evolution of the RMSE through 100 iterations for both approaches is shown in Figure 4.1a for the ToA scheme. The localization process was repeated using the RSS measurements. It was necessary to eliminate a bias in the RSS range estimates as discussed in [31]. Figure 4.1b shows the RMSE curve over 100 iterations for the RSS case.

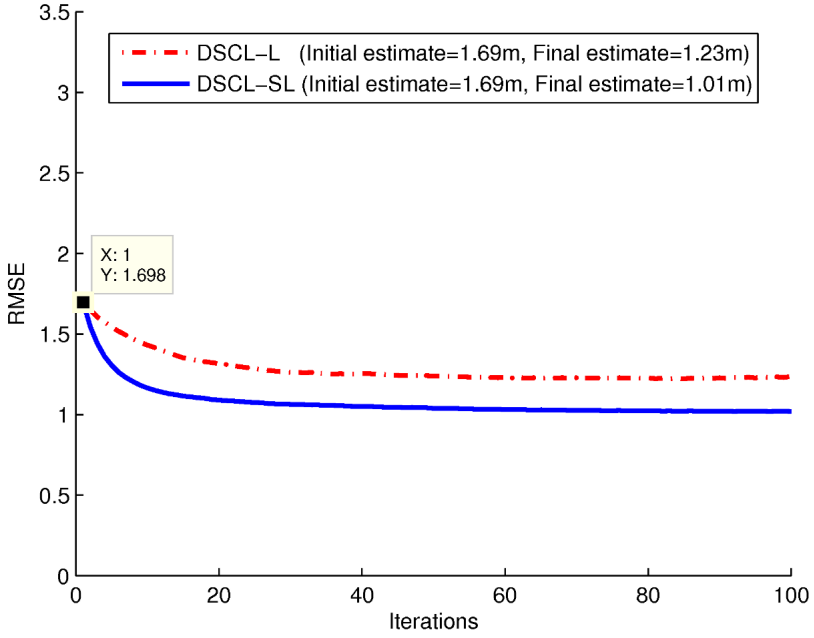
Figures 4.1a and 4.1b show that both approaches (DSCL-L optimization and DSCL-SL) have good performance for both ranging schemes. Both RMSE curves have a smooth decay with convergence towards a minimum point. There is a fast decay for the first 10 to 20 iterations, and then the decay slows down significantly. However, the DSCL-SL approach presents a significant

improvement over the DSCL-L scheme. Our belief of this good behavior is that the DSCL-SL optimization algorithm uses one-to-one constrained minimization problems that are easier to satisfy as opposed to the one-to-many constrained minimization problems for the DSCL-L approach. Figure 4.2a and 4.2b show the position estimates for both ranging techniques (ToA an RSS measurements) under the DSCL-SL approach. In the Figures, anchors are marked with the symbol '▲', true unknown sensors are marked with the symbol '◇', and estimated positions are marked with the symbol '♦'. Initial sensor positions were found using Algorithm 2, and then positions were refined by performing 100 iterations of the DSCL-SL Algorithm 4.

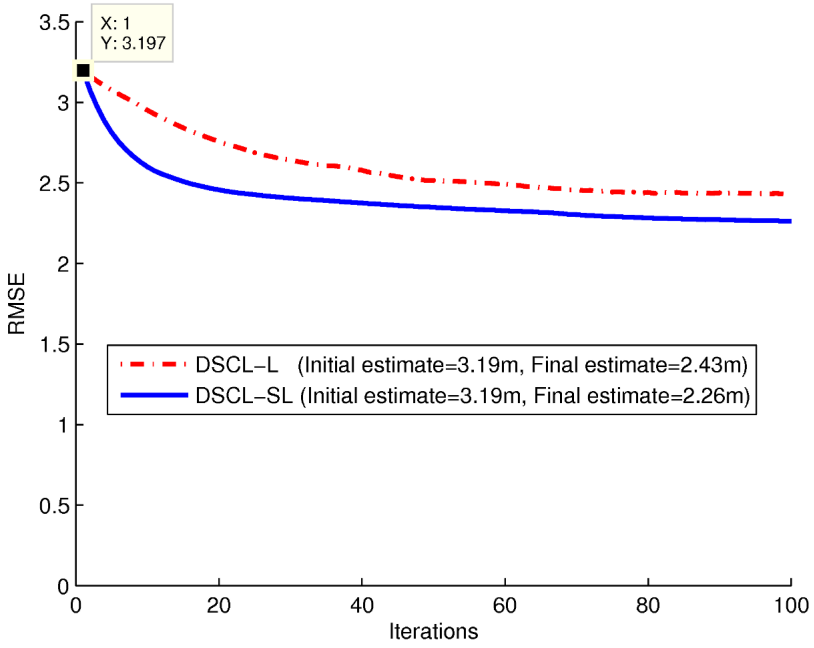
Table 4.1 presents the RMSE comparison with other localization schemes that have used the same data set. As can be seen, both algorithms DSCL-L and DSCL-SL are very competitive. In particular, the DSCL-SL approach provides the best result for ToA. In the case of RSS, the DSCL-SL approach provides the second best RMSE only outscored by the MLE scheme [49] which is a centralized method that requires *a priori* knowledge of the range measurements distribution. We should also note that the dwMDS results reported in [31] are obtained by running dwMDS twice with a range threshold of six meters. Next, we evaluate the performance of the propagation models which will be used in future simulations.

Table 4.1: RMSE obtained from different localization algorithms.

	Classical MDS	MLE	dwMDS	PPE	DSCL-L	DSCL-SL
RSS	4.26m	**2.18m**	2.48m	2.44m	2.43m	**2.26m**
ToA	1.85m	1.23m	1.12m	**1.10m**	1.23m	**1.01m**

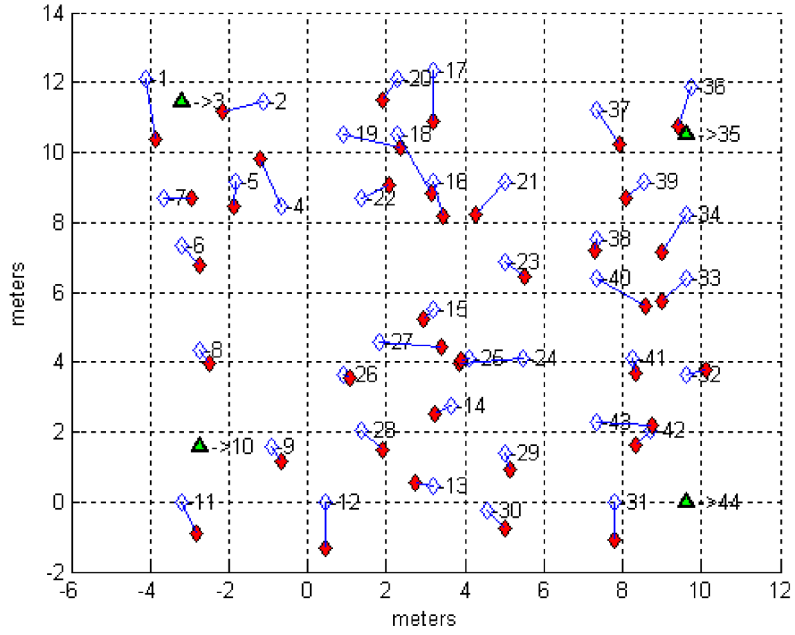


(a) ToA

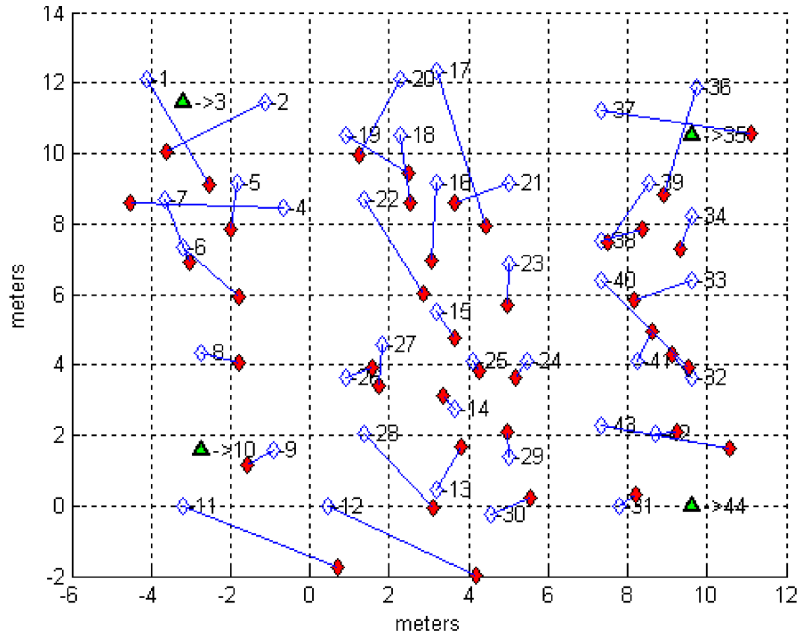


(b) RSS

Figure 4.1: Evolution of RMSE as a function of algorithm iterations considering DSCL-L and DSCL-SL approaches.



(a) ToA



(b) RSS

Figure 4.2: Position estimates using the bilateration and DSCL-SL approaches with TOA and RSS measurements. Anchors =‘▲’, true positions=‘◇’, estimated positions=‘⧫’. The RMSEs are 1.01m and 2.26m for the ToA and RSS schemes respectively.

4.3 Evaluation/Validation of Propagation Models

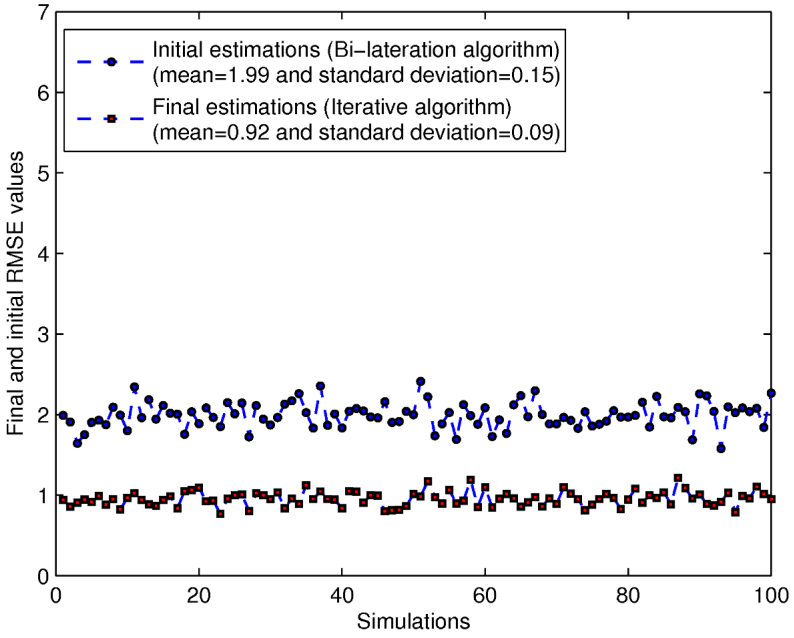
In this section, we test the propagation models with the network configuration used in the real indoor network from the last section. Having knowledge of the localization performance on real data, this test allowed us to validate the performance on the algorithms with simulated data and to ensure that the implementation of the propagation models was correct. In other words, we expect to see similar behavior of the algorithms with synthetic data that characterizes real measurements.

We consider first the ToA measurement model reported in [49] where the range errors are modeled as zero-mean additive Gaussian noise. Hence, it would be possible to simulate the true locations of the 44 node network from the previous subsection with measurement error model given by the random variable $E \sim \mathcal{N}(0,(1.84m)^2)$. We performed 100 ToA localization simulations using the 44 node network specifications, each with a different realization of the noise process. The anchors remained fixed to the positions used in the original set up. For each network realization, the bilateration algorithm was applied as the initial step, followed by 100 iterations of the refinement scheme (i.e., DSCL-SL approach). In Figure 4.3 we plot the initial RMSEs provided by the bilateration algorithm versus the network realization, and also the final RMSEs (after 100 iterations). A robust uniform behavior is observed across the 100 network realizations. In all cases, the algorithm converges with behavior similar to Figure 4.3-(a). We also plot the RMSE histograms for bilateration and refinement on Figure 4.3-(b). The mean and the standard deviation for the initialization step are $1.99m$ and $0.15m$, respectively, while for the final RMSE we get $0.92m$ and $0.09m$, respectively. Our optimal localization scheme reduced the error by half. With respect to the real measurements used on the previous section, we see (on the average) a similar performance as in Table 4.2 which empirically validates the additive error model. Furthermore, in [51], a Kolmogorov-Smirnov (KS) test was applied to the real ToA data showing that it fits the Gaussian model with a high p-value of 0.5 at a level of significance of 0.05.

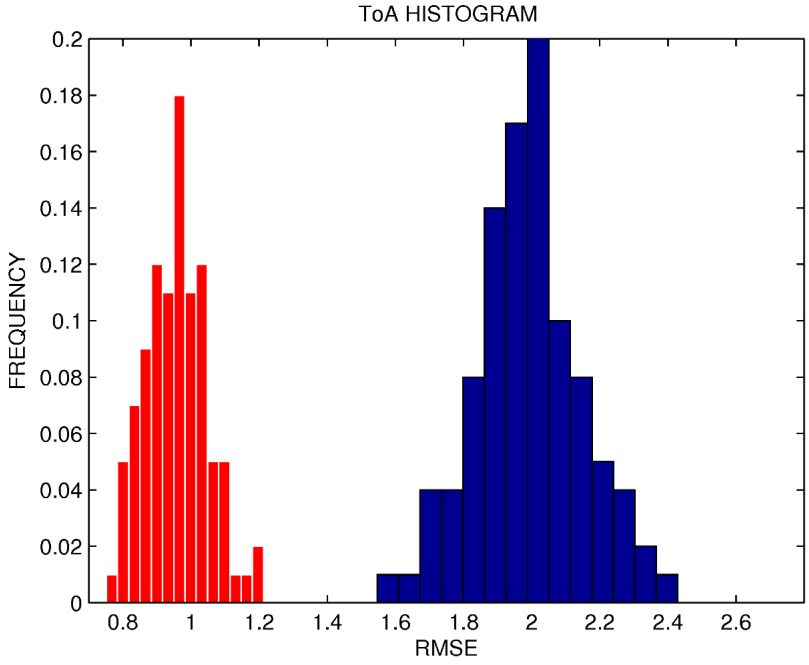
In a similar way, the simulations were performed using RSS. In our simulations we set $P_0 = -37.4663$, $d_0 = 1m$, $\eta_p = 2.3022$, $\sigma^2_{dB} = 3.92$, which are based on the real network measurements. The bilateration step presents high RMSE values as shown Figure 4.4. This is related to

the multiplicative behavior of RSS errors (given in dB) as a function of distance. The same measurement error will have a more significant effect at large ranges than for short ranges. Hence, during the initialization step the sensors located at the edge of the network may present large range errors for the anchors located at the opposite corners in the network, as illustrated in Figure 4.2. On the contrary, position of sensors that are relatively equally spaced to the anchors tend to be well estimated.

Despite the larger (multiplicative) errors given by RSS measurements, we find that our DSCL-SL optimization algorithm provides good localization results using the bilateration positions as an initial point. Our conjecture is that the collaborative nature of both schemes quickly averages out the bias caused by the large measurement errors. We show in Figure 4.4 the bilateration RMSE and the RMSE for the iterative algorithm over 100 network simulations; in all cases the localization process converges numerically. The histograms for bilateration and optimal refinement are presented in Figure 4.4. The mean and the standard deviation for the final RMSE values are $1.27m$ and $0.149m$ respectively which is significantly lower than those on Table 4.1. Clearly, the histograms show heavy tails that do not fit a $\mathcal{N}(0,(3.92dB)^2)$ distribution. As shown in [51], a KS test indicates low correspondence with the Gaussian model implying that the model does not capture other signal degradation factors in (4.4). Nonetheless, the ToA and RSS models provide a good reference for further evaluation of our localization schemes. In particular the RSS shadowing model is broadly used by the community of wireless communications. Next, we show how the radio transmission range can affect the quality of localization.

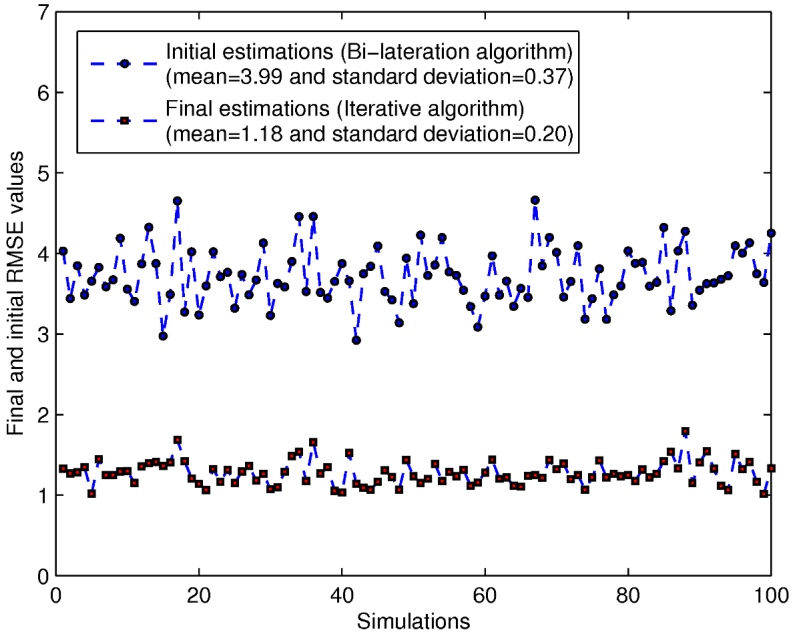


(a) RMSE for 100 network simulations.

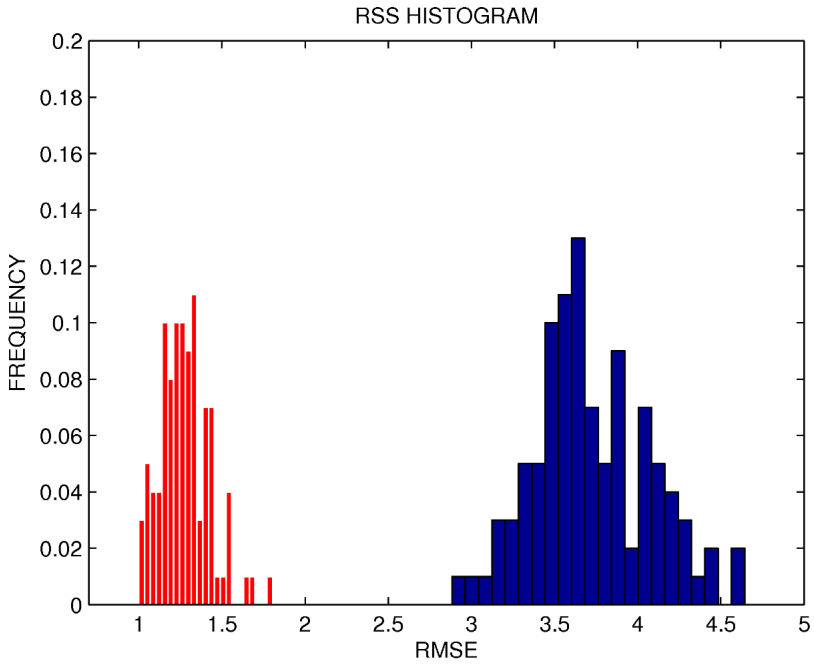


(b) RMSE Histograms for both algorithms.

Figure 4.3: RMSE for bilateration on optimized localization of 100 runs using simulated ToA measurements.



(a) RMSE for 100 network simulations.



(b) RMSE Histograms for both algorithms.

Figure 4.4: Analysis of RMSE for bilateration on optimized localization of 100 runs using simulated RSS measurements.

4.4 Impact of Communication Range on Localization Schemes

The results discussed on previous sections present a fully connected network deployed over a relatively small area (14m by 13m). Hence, each node has 43 range measurements to estimate its location. Even for this small area, large errors are introduced for nodes that are far apart, so it could be advisable to reduce the RF range of the nodes and estimate locations using a reduced set of neighbors. We use a short range scenario in order to evaluate the robustness of the localization algorithms as the number of available measurements is reduced. In essence, our estimates from equation (3.11) would be noisier and the impact on convergence and localization error are of concern. Furthermore, a fully connected network would not be possible over areas far larger than the range provided by the transceivers. This last scenario is explored in detail in the next section.

We allow anchors to connect with all sensor nodes (i.e., having a large RF range) in order to use the bilateration algorithm for initialization. Other initialization schemes like closed form LS and iterative LS can also be considered. Let N_i be the set of sensors available to each sensor s_i,

$$N_i = \left\{ (i,j) \,|\, i < j, \left\| \mathbf{z}_i - \mathbf{z}_j \right\| < R \right\}, \tag{4.6}$$

where R represents the *well-known* isotropic radio range, and $\mathbf{z}_i$ is the true location for node s_i [13]. To ameliorate the uncertainty introduced on the ranging measurements r_{ij}, each node s_i applies the following relation

$$r_{ij} = R \quad iff \;\; r_{ij} > R, \tag{4.7}$$

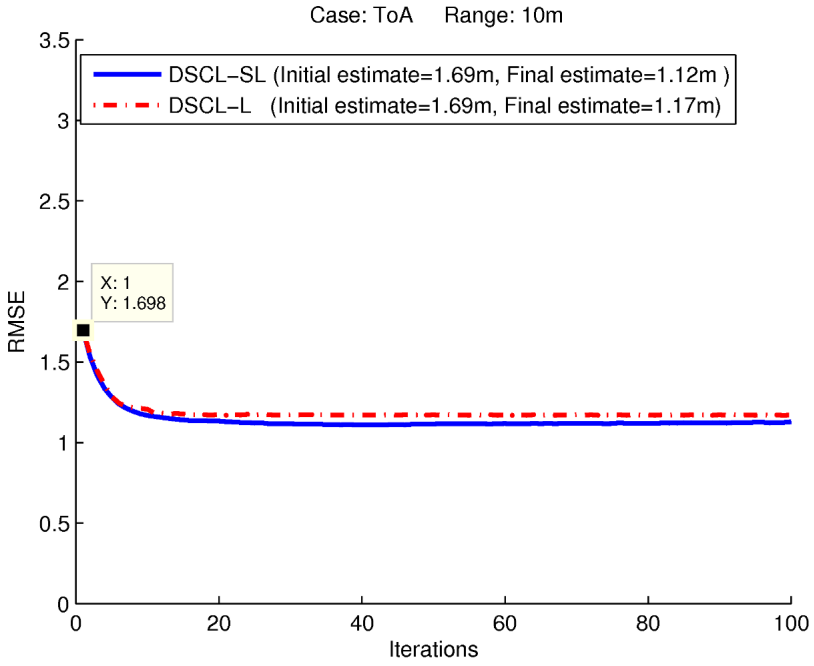
to adjusts its range estimate r_{ij} to node s_j when it is greater than R. Similar geometrical criteria has been considered in other works [31]. This rule is effectively truncating measurement errors above the range limit. As before described, in this work we assume that R *ideally* remains constant along all directions.

For the network described in section 4.2 we consider three scenarios with $R = 10m$, $R = 5m$, and $R = 3m$ with an average number of connected neighbors, $|N_i|$, of 33.75, 14.7, and 8.3 respectively. Note that the anchors are not used in these averages. We test the effect of short

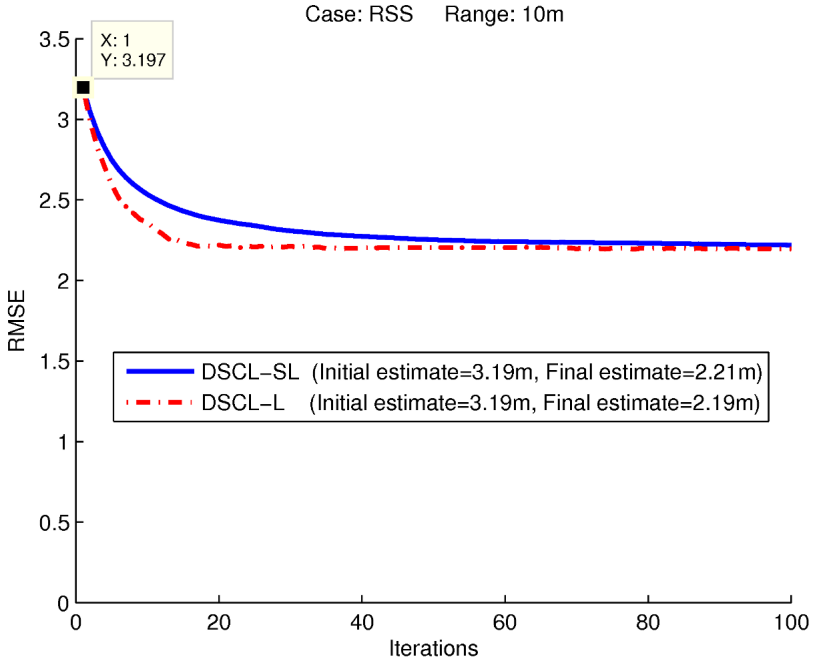
range in the estimated locations under ToA and RSS error measurements. Figure 4.5 shows the performance for both ToA and RSS measurements considering that each sensor s_i has a range of $R = 10m$, and they satisfy the restrictions imposed by equations (4.6) and (4.7).

In a similar way, Figure 4.6 and Figure 4.7 show the RMSE decay over 100 iterations at $R = 5$ and $R = 3$.

The RMSEs after 100 iterations are presented in Table 4.2, including the full-connectivity case. We see the expected degradation in accuracy performance as the range decreases. On the other hand, the degradation at $3m$ is within the accuracy performance of the competing algorithms in Table 4.1. In the case of RSS at $10m$ there was a small improvement compared to full connectivity which illustrates the stronger effect of shadowing on measurement ranges when sensors further apart are used. The $3m$ case is remarkable as it performs very similar to dwMDS (with a reported range of $6m$) and PPE (full connectivity). Looking at the RMSE curves for ToA over short ranges, there is clearly an impact on performance as the range is reduced. In all ToA cases, a minimum was reached before the 100 iterations after which RMSE curves show small oscillations and a tendency to increase. For instance, at a range of $3m$ in the DSCL-SL scheme, a minimum RMSE of $1.21m$ occurs at iteration seven; more dramatically, at $5m$ the error goes down to $1.17m$ at iteration nine and then gradually increases to $1.38m$ over the next 99 iterations. A less noticeable effect is observed when R set at $10m$. The last results suggest that it is necessary to develop a stopping criterion that keeps track of the error trends such that the localization process stops if the localization process starts to present small variations on the estimated positions. Since the algorithm is distributed in nature, each node will have to track and decide locally when to stop and how to communicate the decision to its neighbors. This has the side benefit of reducing the overall network energy consumption. This idea is further discussed in the next chapter.

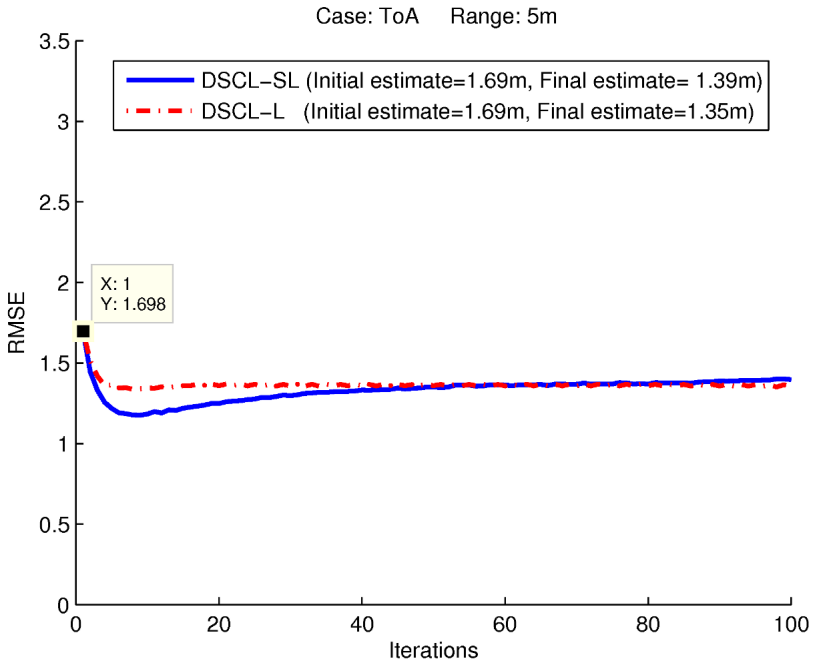


(a) ToA scheme.

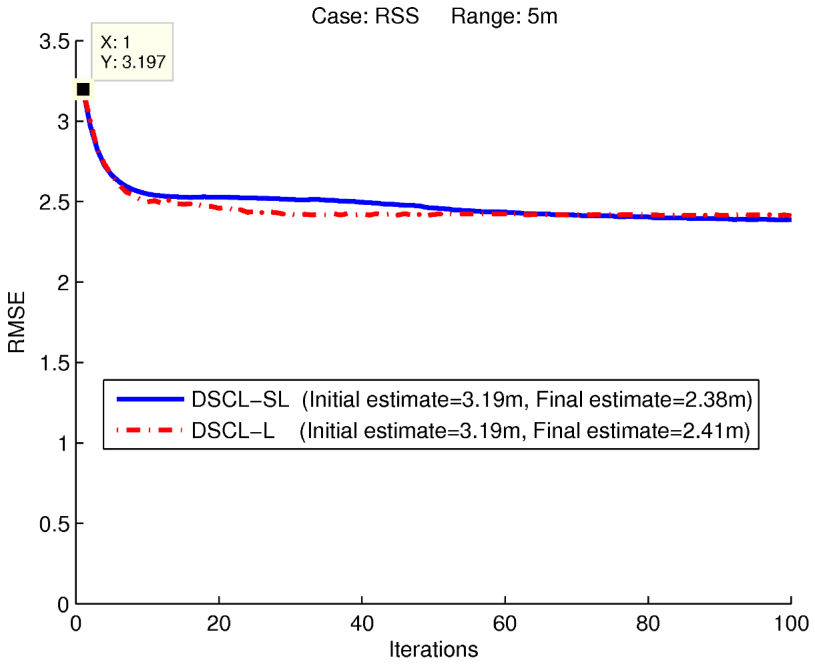


(b) RSS scheme.

Figure 4.5: Accuracy performance of DSCL-L and DSCL-SL algorithms considering a range of 10m with ToA and RSS network measurements.

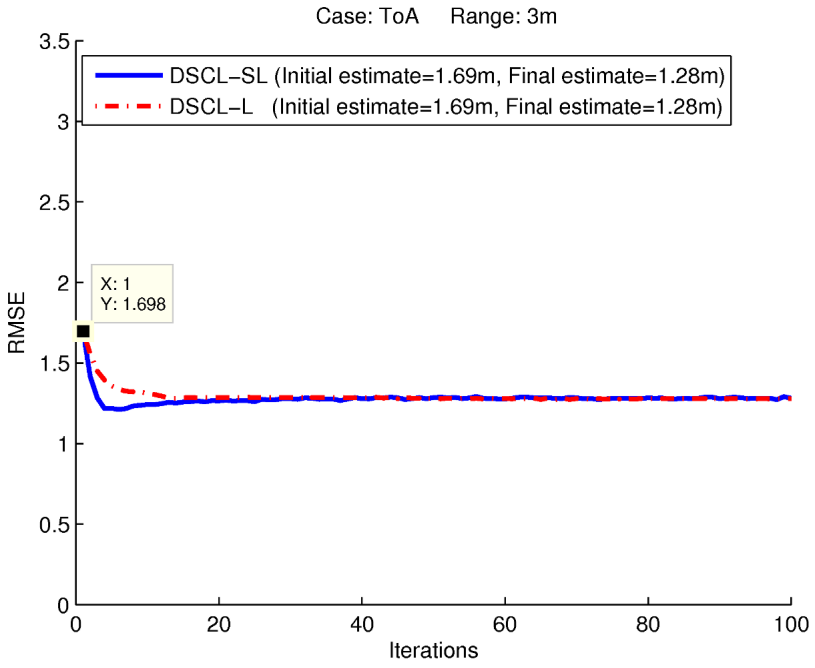


(a) ToA scheme.

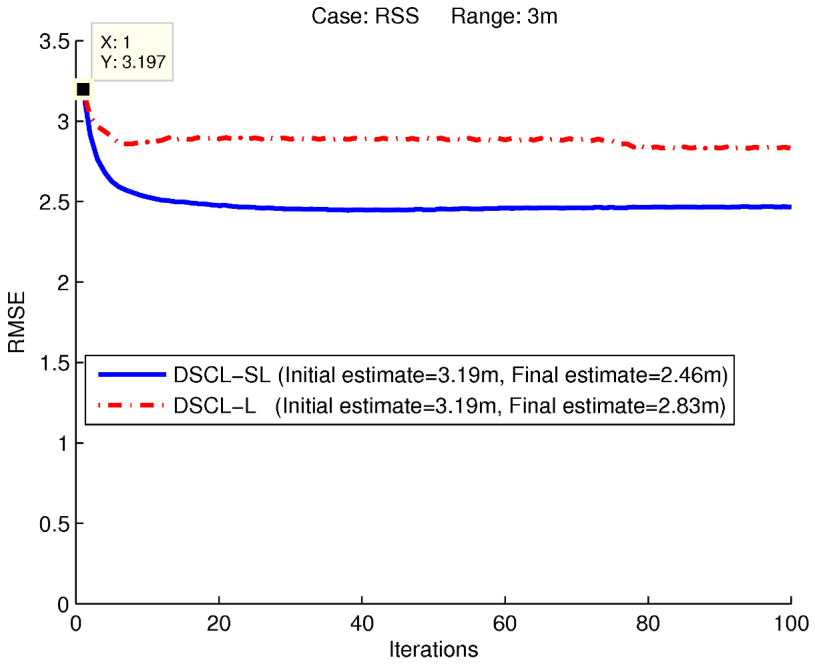


(b) RSS scheme.

Figure 4.6: Accuracy performance of DSCL-L and DSCL-SL algorithms considering a range of 5m on regular sensors and different error measurements, ToA with parameters $\mathcal{N}\left(0, 1.84^2\right)$ and RSS with parameters $\mathcal{N}\left(0, 3.92^2\right)$



(a) ToA technique.



(b) RSS technique.

Figure 4.7: Accuracy performance of DSCL-L and DSCL-SL algorithms considering a range of 3m on regular sensors and different error measurements, ToA with parameters $\mathcal{N}\left(0, 1.84^2\right)$ and RSS with parameters $\mathcal{N}\left(0, 3.92^2\right)$.

Table 4.2: Accuracy on the estimated positions under different short range schemes. *Note: The initial RMSE estimates for the ToA and RSS schemes using the bilateration algorithm are* 1.69*m and* 3.19*m respectively.*

		Full connectivity	Range=10*m*	Range=5*m*	Range=3*m*
DSCL-L	RSS	2.43*m*	2.19*m*	2.41*m*	2.83*m*
	ToA	1.23*m*	1.17*m*	1.1.35*m*	1.28*m*
DSCL-SL	RSS	2.26*m*	2.21*m*	2.38*m*	2.46*m*
	ToA	1.01*m*	1.12*m*	1.39*m*	1.28*m*

4.5 Localization Over Large Geographical Areas

Perhaps, the main limitation of the localization experiments on the previous section is that the measurements correspond to a small indoor office area. However, this comment is not to demerit the great effort needed to coordinate the measurement campaign that provided this data. Measurement campaigns are typically a cumbersome process requiring a complex setup. This illustrates the difficulties of dealing with the distributed nature of WSNs for test and evaluation. For instance, ToA measurements require hardware that is out of reach for feasible sensor node architectures. On the other hand, RSS circuitry is commonplace in wireless devices, however some works indicate that reliable measurements are difficult to obtain [41, 4].

In this section, we resort to simulation to evaluate the performance of the proposed localization algorithms over large areas and large deployments. We consider a large area anything above 100 meters by 100 meters, and a large deployment a WSN with 100 or more sensors. These values are nominal in the sense that they have been considered in many works; however, to the best of our knowledge there is no formal definition for such parameters. A basis of justification would be given by the communication range and power constraints commonly assumed in WSNs. In particular, range is typically considered to be on the 30 meter range. We are interested on implementing a realistic simulation framework which will be representative of a large WSN outdoor deployment. Moreover, we want the simulations to be as realistic as possible with respect to hardware and wireless specifications that reflect currently available technologies.

Regarding the wireless technology used (or being considered) for WSN at the time of this writing, the use of 802.15.4 has become the standard for the physical layer and medium access control of WSN protocol stacks. On top of this standard, other protocols can be constructed all the way to the application layer. The Zigbee specification seems to be at the forefront of standardization efforts (although other new comers like 6LowPaN are gaining traction). Hence, it would be of practical importance to evaluate localization algorithms under a context that is as close as possible to a real deployment. Given its current clout at the industry level, we develop our simulations assuming a Zigbee-based network. Current COTS (commercial-off-the-shelf)

products like XBee modules specify ranges of 100 meters with a chip antenna (up to 258m with a whip antenna) with a transmit power of 1 mW [70]. It is possible for these modules to lower the power to reduce the range and increase battery life. For reasons to be discussed below, we also consider the use of Zigbee-PRO which is a newer specification with a range upwards of 1.5 Km and a transmit power of 60 mW. Besides, a higher transmit power, the Zigbee-PRO radios have a much higher receiver sensitivity. In this research, we particularly consider 2.4 GHz as the carrier. A relevant point to note is that Zigbee and Zigbee-PRO are interoperable specifications, and there is already an XBee-PRO devices in the market.

The ability for these two specifications to interoperate opens the door to the feasibility of an heterogeneous deployment where one could envision using sensor nodes outfitted with Zigbee modules and anchor nodes equipped with Zigbee-PRO radios. This differentiation between sensor nodes and anchors seems to be a logical one supported by many works. The technical profile of an anchor node should reflect a more capable device from the computation and energy perspective. Since anchors are expected to have GPS in the case of outdoor deployments, then a larger battery would be needed. This also implies the possibility of using a more powerful microcontroller and radio units (e.g., Zigbee-PRO). With respect to the computing capabilities, it is easy to see that the proposed algorithms can be deployed on computationally constrained micro-controller units (MCUs). Current WSN mote technology has maintained the use of 8 or 16-bit MCUs with low power profiles as part of its power conservation philosophy. Nonetheless, these devices are capable of emulating floating point operations. A salient feature of the proposed algorithms is that they can be implemented on these devices with relatively low computational cost for each iteration. As computation is an order of magnitude (or more) below the power consumption of the RF module, then we claim that computational requirements of our localization algorithms are of less concern in our simulations for this chapter. A full power profile of the localization process is discussed in the next chapter.

4.5.1 WSN Localization for RSS Measurements over Large Areas

As it is well known, RF signals represent a common medium to share information among nodes in WSNs. This characteristic is exploited through the process of measuring the path loss of a signal traveling from a transmitter node to a receiver node (i.e., RSSI), which indirectly provides an estimate of the distance between node-pairs; thus, this technique represents an attractive and inexpensive way for range-based schemes.

Based on the aforementioned observations, we built a simulation framework using the propagation models from the section 4.3 and a WSN model based on the characteristics of Zigbee and Zigbee-PRO radios. The network is heterogeneous, assuming that anchors that meet a Zigbee-PRO profile and higher power/computational capacities. Giving this characteristic to anchors allows communication with other anchors and the lower profile (i.e., shorter range) nodes, such that the bilateration algorithm can be used to find the initial positions. The sensor nodes can be outfitted with regular Zigbee since full connectivity over large areas would be prohibitive. As expected, the ratio of anchors to sensors will be low. We assume the availability of a network protocol that will allow the network to self-organize the order in which the sensors broadcast their updates. The research on such protocols is vast, and it is outside of the scope of this dissertation.

We simulate RSS-based localization using the propagation model from equation (4.4) assuming that an heterogeneous WSN with carrier frequency of 2.4 GHz is used on the transceiver of both anchors and sensor nodes. The anchors a_j have full connectivity with a transmission power of 60mW (18 dBm) and the sensor nodes s_i have a transmission power of 1mw (0 dBm). We assume uniform ideal connectivity for $R \leq 100m$. The last considerations have two purposes, the first is to enable the sensors to generate initial position estimates from the anchor ranges using the bilateration algorithm, and second to avoid the issue of irregular transmission patterns. The latter will be the discussion of our future work. In our simulations, we are using $P_0 = -70dB$ for nodes and $P_0 = -52dB$ for anchors according to current commercial specifications for wireless motes. We test our algorithms at different scenarios considering variations on R, η_p, and σ_{SH}. All noisy pair-wise simulated ranges r_{ij} *and* R_{ik} are obtained averaging 10 range measurements

as typically done in practice [52].

For the simulations that follow, we consider 20 different sensor networks where each one is composed by $N = 100$ sensors, randomly distributed, in a $100m$ by $100m$ area. Also, we select four non-collinear anchors on every realization. For each network, we generate a set of noisy ranges using the log-distance path loss model (4.4) and generate a set of initial locations using the bilateration algorithm. Using these initial positions, both iterative algorithms (i.e., DSCL-L and DSCL-SL approaches) are ran for 100 iterations. As before, RMSE is used as a performance metric by comparing the estimated locations with the true locations. Also, we assume a representative path-loss factor of 2.6 for outdoors scenarios [5].

The log-normal σ_{SH} parameter is varied at steps of 1.3 starting from 0 to 9.1 where each value of σ_{SH} is tested over the 20 network topologies, and also each value of σ_{SH} corresponds to a different noise process in each network. Considering that we have $R = 100m$, and equations (4.6) and (4.7) are satisfied, 20 RMSE values are obtained for each value of σ_{SH} as shown the setup of Figure 4.8.

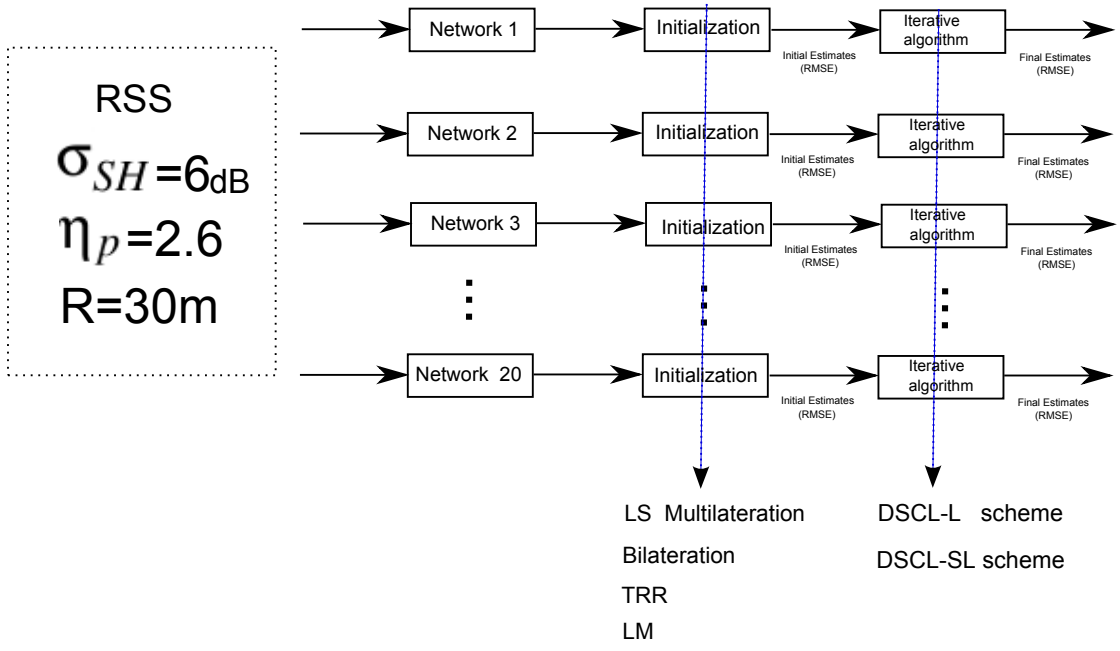
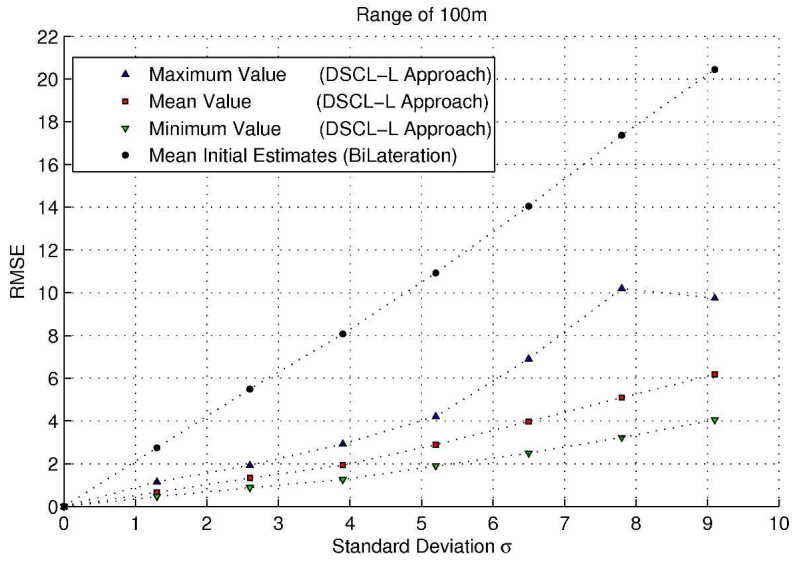


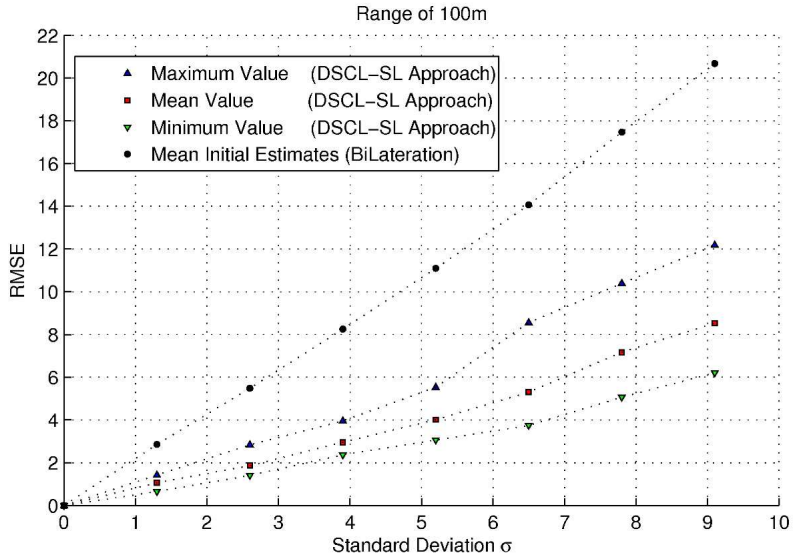
Figure 4.8: Setup used to test the accuracy performance of proposed algorithms.

In Figures 4.9a and 4.9b, we plot RMSE versus σ_{SH} for these simulations under the DSCL-L

and DSCL-SL approaches respectively.



(a) DSCL-L algorithm.



(b) DSCL-SL algorithm.

Figure 4.9: Localization accuracy at R=100*m* on regular sensors using the RSS model with $\eta_p = 2.6$ at different σ_{SH}.

The highest curve marked by symbols '●' represents the mean RMSE initial estimates provided by the bilateration algorithm, the curve marked by the symbols '▲' shows the maximum

RMSE value, the middle curve marked with the symbols '■' represents the mean RMSE value, and, finally, the lower curve marked with the symbols '▼' represents the minimum RMSE value over the 20 WSNs provided by both iterative algorithms.

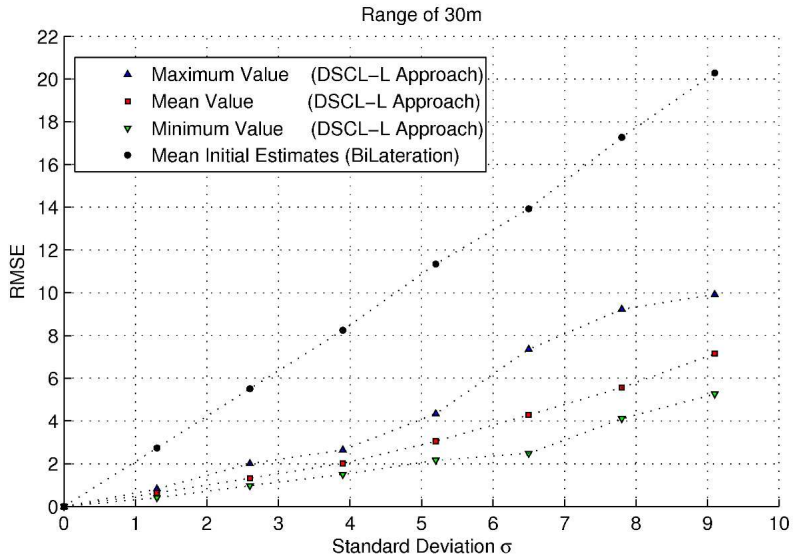
For typical σ_{SH} values, between 4_{dB} and 6_{dB}, Figure 4.9a shows narrow variations around the mean RMSE values implying that the DSCL-L approach provides robust estimates. For example, RMSE values at $\sigma_{SH} = 4dB$ present $1.71m$ of error while RMSE values at $\sigma_{SH} = 6dB$ presents a higher error of $3.61m$ as expected. As can be seen mean RMSE values are relative low implying that our DSCL-L iterative algorithm is well behaved. Also, there is an improvement of $6.28m$ and $9.24m$ between bilateration and the DSCL-L l iterative scheme at $\sigma_{SH} = 4_{dB}$ and 6_{dB} respectively.

On the other hand, a similar relation between initial and final position estimates is presented by the DSCL-SL approach shown in Figure 4.9b. The iterative DSCL-SL approach also shows narrow variations around the mean RMSE value. For example, RMSE values at $\sigma_{SH} = 4_{dB}$ present $1.66m$ of error while RMSE values at $\sigma_{SH} = 6_{dB}$ presents a higher error of $3.91m$. Also, there is an improvement of $5.43m$ and $8.12m$ between bilateration and the DSCL-SL scheme respectively. The DSCL-L approach presents slightly better results in our simulations than the DSCL-SL approach.

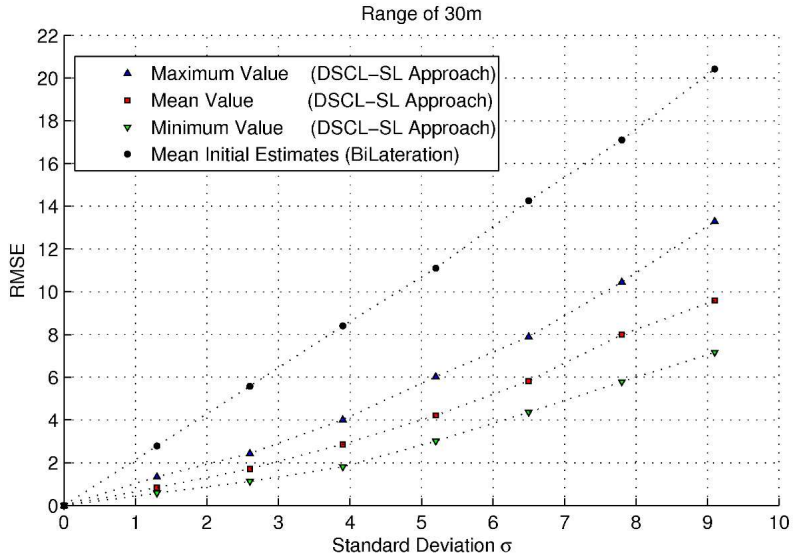
We repeated the last procedure over the 20 networks using R=$30m$. The bilateration algorithm was used to determine initial estimates. Figures 4.10a and 4.10b show the relation between the accuracy (RMSE) versus noise (σ_{SH}) for DSCL-L and DSCL-SL approaches respectively. Surprisingly both algorithms provide mean RMSE values almost similar to R=$100m$. We believe that reducing R and using equation (4.7) helps the localization process in both ways: 1) by bounding noisy distance estimates larger than R and 2) by eliminating large distances, such that the multiplicative error-effect over long estimated distances is reduced by considering distances with a maximum range of $30m$.

As can be seen the effect of reducing R from $100m$ to $30m$ provokes an small increment in the mean RMSE values for R=$30m$, so this effect is desirable in realistic scenarios where the goal is to save energy (in this case by reducing R) without sacrifice accuracy performance.

As final point we remark that the DSCL-SL approach presents slightly less accuracy in the



(a) DSCL-L algorithm.



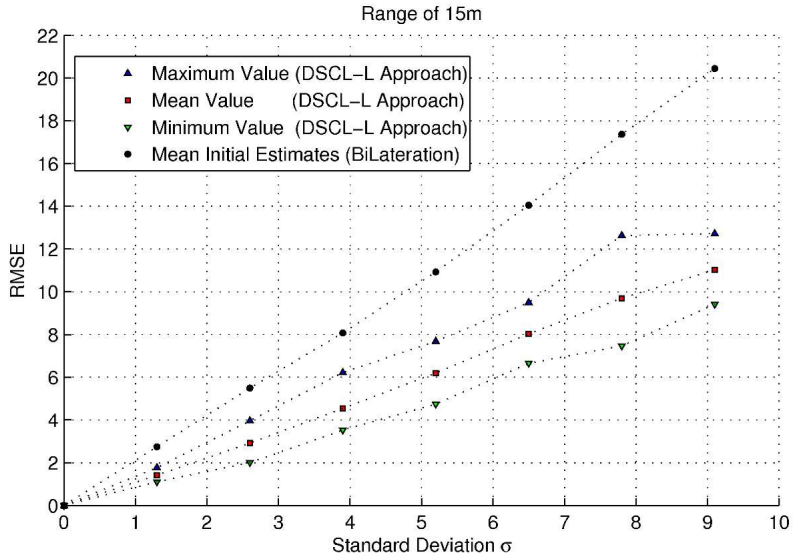
(b) DSCL-SL algorithm.

Figure 4.10: Localization accuracy at R=30m on regular sensors using the RSS model with $\eta_p = 2.6$ at different σ_{SH}.

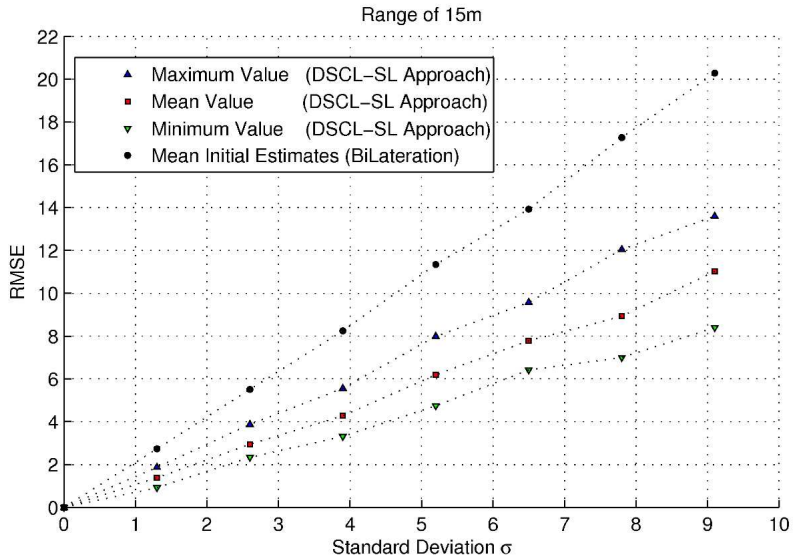
position estimates, which also indicates that the DSCL-L approach is more robust in large-scale scenarios with RSS measurements.

Finally, we repeated the same procedure using R=15m over the 20 networks where the bilat-

eration algorithm is used to provide initial estimates. Figures 4.11a and 4.11b show the relation between accuracy (RMSE) versus noise (σ_{SH}) for the DSCL-L and DSCL-SL approaches respectively.



(a) DSCL-L algorithm.



(b) DSCL-SL algorithm.

Figure 4.11: Localization accuracy at R=15m on regular sensors using the RSS model with $\eta_p = 2.6$ at different σ_{SH}.

We can deduce that based on the mean RMSE values there are no substantial differences between the DSCL-L and DSCL-SL algorithms. However, we can see that the mean RMSE values of position estimates for this coverage range is higher than R=30m and R=100m implying that iterative algorithms have the worst performance under this scenario as expected. For example, at $\sigma_{dB} = 6_{dB}$ the DSCL-L approach provides a mean RMSE error of 7.32m while 3.8m and 3.56m are reported using R=30m and R=100m respectively. In a similar way, the DSCL-SL approach at R=15m provides a mean RMSE value of 7.16m while 5.2m and 4.80m are reported using R=30m and R=100m respectively. This last results imply that decreasing R clearly affects the accuracy in the estimated positions.

Figure 4.12 plots the mean RMSE values of each range of coverage R. Also, we plot the mean RMSE values of the bilateration algorithm. We can observe that both iterative algorithms achieve good estimate positions in spite of high RMSEs in the initial estimates.

In Table 4.3, we summarize some statistics of neighbor densities, (NDs), obtained from the 20 WSNs. We present RSS RMSE behaviors at different R with $\sigma_d = 3.9_{dB}$ and $\eta_p = 2.6$. This table shows interesting relations among R, RMSE, and ND. For example, if we reduce R from 100m to 15m (a factor around 6), the neighbor density ND per sensor is reduced from 93.79 to 6.2 (a factor of 15), while only increasing the mean error from 1.92m and 2.95m to 4.53m and 4.27m for the DSCL-L and DSCL-SL approaches respectively. This shows an excellent degree of robustness under highly constraint scenarios. Also, it implies that we can use shorter ranges with significant energy savings at the expense of little loss in accuracy performance if this is acceptable. These and other issues will be treated in the next chapter.

Table 4.3: Effects of ranging nodes in the accuracy of the estimated positions (RMSE metric) using 100 sensor nodes.

	$\sigma_{SH} = 3.9dB and \eta_p = 2.6$						NEIGHBOR DENSITY				
	DSCL-L Approach			DSCL-SL Approach							
Range	min	mean	max	min	mean	max	median	mean	mode	min	max
100m	1.25	1.92	2.93	2.36	2.95	3.95	95	93.79	95	81	95
30m	1.49	2.02	2.64	1.81	2.85	4.01	21.5	21.2	23	8	35
15m	3.52	4.53	6.22	3.30	4.27	5.55	6	6.2	5	1	14

On the other hand, it is very interesting to compare both parameters R and ND individually to analyze which one affects, in a stronger way, the estimated positions. Under this perspective, we take as basis the last row from table 4.3 which provides information for both R=15m and ND$\approx$ 6 nodes. In order to increase the node densities in each WSN, we add new nodes in a random way to the 100 already deployed nodes. The new WSNs are tested considering R=15m. Table 4.4 shows how the accuracy of the estimated positions in both iterative algorithms are also improved by increasing the neighboring density. However, up to this point, we have not compared (quantitatively) the real effects on doubling the range R versus doubling the number of sensor nodes on the estimated positions. For example, if we double the Range from R=15m to R=30m (taking as basis the the last row from Table 4.3), we can appreciate that the mean RMSE in both algorithms have improved from 4.53m to 2.02m and from 4.27m to 2.85m for DSCL-L and DSCL-SL approaches respectively. On the other hand, if we double the number of nodes from 100 to 200, we can realize that the mean RMSE has improved from 4.53m to 2.63m and from 4.27m to 3.25m respectively. Last results imply that if we increase the range of coverage, we can obtain better estimated positions than adding nodes to the network in both iterative algorithms. However, advantages and disadvantages should be considered in both situations. For example, increasing the coverage range R in a node also increases the energy consumption in a node which is a disadvantage. On the other hand, increasing the number of nodes increases the cost of the whole network; however, it brings more robustness in the case of a node failure, which is an advantage.

Table 4.4: Effects of neighbor densities on the accuracy of the estimated positions (RMSE metric) using R=15m.

	$\sigma_{SH} = 3.9dB and \eta_p = 2.6$						NEIGHBOR DENSITY				
	DSCL-L Approach			DSCL-SL Approach							
Nodes	min	mean	max	min	mean	max	median	mean	mode	min	max
100	3.52	4.53	6.22	3.30	4.27	5.55	6	6.2	5	0	15
150	3.19	3.72	5.09	2.99	3.69	4.89	9	9.1	10	1	21
200	1.78	2.63	3.62	2.37	3.25	4.87	12	12.2	12	2	25

As an additional assessment, we present in Table 4.5 the relation between the most represen-

Table 4.5: n_p versus σ_{SH} using 30m of coverage range on sensor nodes.

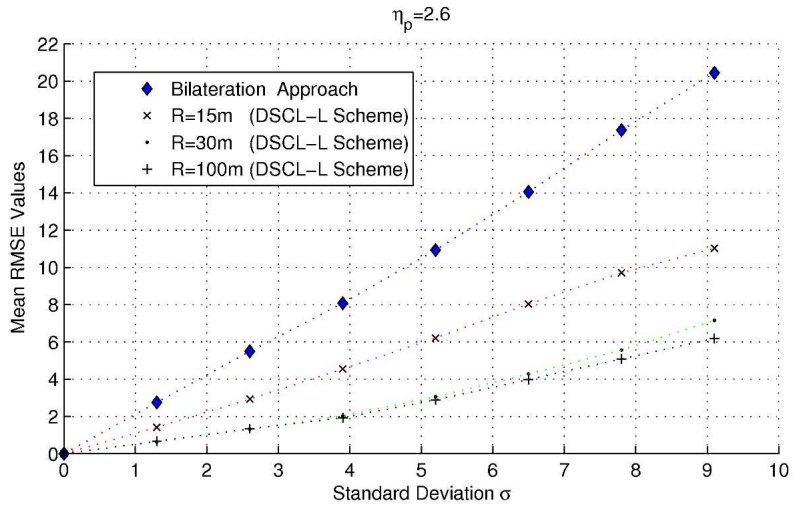
(b) Bilateration algorithm results.

η_p \ σ_{SH}	4.0	4.5	5.0	5.5	6.0
2.0	10.9	13.0	14.35	15.67	17.32
2.5	9.24	9.94	11.09	12.7	13.8
3.0	7.34	8.22	9.16	10.1	11.2
3.5	6.61	7.20	7.81	8.81	9.71
4.0	5.42	6.03	6.89	7.49	8.17

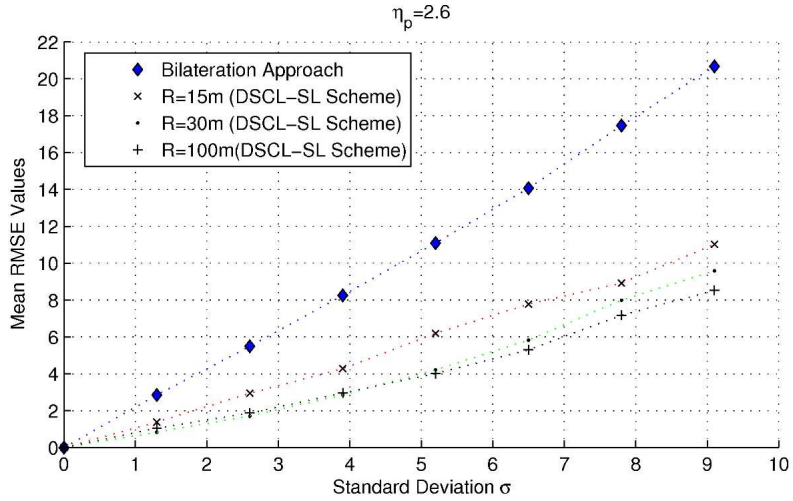
(a) DSCL-SL algorithm results.

η_p \ σ_{SH}	4.0	4.5	5.0	5.5	6.0
2.0	4.0	5.29	5.89	6.82	8.23
2.5	3.39	3.73	4.19	5.17	5.60
3.0	2.40	2.93	3.28	3.60	4.26
3.5	2.19	2.45	2.76	3.18	3.55
4.0	1.69	1.88	2.05	2.59	2.73

tative values in η_p, σ_{SH}, with R=30m using the bilateration and the iterative DSCL-SL algorithm. We assume that equation (4.7) is satisfied, and the parameters described earlier for the transmitter/receiver are considered in our simulations. In each combination of values we are using the average RMSE of the 20 WSNs. Table 4.5 shows that both algorithms provides better estimated positions at high values in η_p and low values in σ_{SH} implying that our procedures and simulations have good correspondence with [54, 35], which discusses that better estimated distances are obtained for large values in η_p and low values in σ_{SH}. In addition, we can observe that the RMSEs, for the initialization step in Table 4.5 (a), are greatly improved by the sublocal algorithm shown in Table 4.5 (b) which shows an improvement of at least 4m. These results show that our iterative algorithm is capable of performing localization despite the adverse conditions presented in RSS measurements. Simlilar results were obtained for R=100m and R=15m.



(a) DSCL-L Approach.



(b) DSCL-SL Approach.

Figure 4.12: Mean RMSE values at different coverage ranges on sensor nodes using the RSS model with a fixed $\eta_p = 2.6$ and different σ_{SH}.

4.5.2 Effect of Additive Errors on Localization Over Large Areas

In this subsection, we consider the DSCL-SL approach performance in cases where the ranging errors are additive Gaussian as described by the model (4.1). As discussed before, these cases correspond to ranging errors on ToA measurements. Furthermore, the additive case (not necessarily Gaussian) has been commonly used to evaluate many localization schemes. For our assessment, we varied two parameters in our simulation: sensor range R and σ_d. Our simulation was ran for each range, by varying σ_d between 0 and 9.9 with increments of 0.3. Each value of σ_d was tested over the 20 network configurations, resulting on 20 RMSE values. In Figure 4.13, we consider the case where sensor range is 100 meters. We plot RMSE versus σ_d where the lower curve marked with the symbols '▼' represents the minimum RMSE value over the 20 network configurations, the curve marked with the symbols '▲' represents the maximum RMSE, the middle curve marked with the symbols '■' represents the mean RMSE over the 20 networks, and the highest curve represents the initial average RMSE over the 20 networks provided by the bilateration algorithm. We can infer there is a linear relationship between RMSE and σ_d. It can be seen that for rather large values of σ_d, the localization algorithm is well behaved as there is not a big deviation between the three lower curves. Also, the iterative algorithm provides a significant improvement on the estimated positions given the initial estimations marked with the symbol '•'. We repeated the experiment for R=45m and R=10m with the results shown in Figure 4.14. The same consistent linear behavior is observed for all σ_d over the 20 networks, even at R=10m where the neighbor density is low.

In Figure 4.15, we have plotted the mean RMSE curves for different coverage ranges at R=3m, R=5m, R=10m, R=15m, R=30m, R=45m, R=60m, and R=100m. The top curve represents the mean RMSE curve for the bilateration algorithm (i.e, this curve represents the localization error for the initial positions averaged over the 20 networks). We can appreciate that the slope of the curves gradually increases as R decreases. We also see that, for $R \leq 15$ and $\sigma_d < 4$, there is no big improvement from the bilateration results. This range for σ_d could be considered a low noise region where there is no incentive to apply the refinement algorithm. In fact, if σ_d could

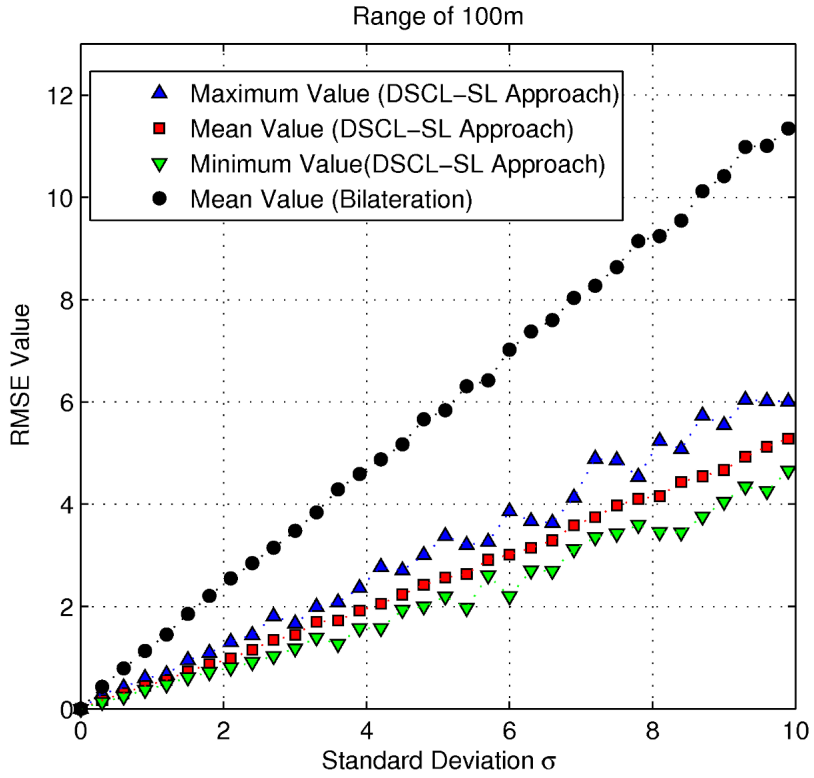


Figure 4.13: DSCL-SL accuracy performance under different levels of noise using a coverage range of 100*m* on sensor nodes and full connectivity on anchors.

be estimated in the field a decision could be made early in the localization process to only do bilateration and achieve significant energy savings. For low values in R (e.g., below 10*m*) there is no localization gain for all σ_d, this can be explained by the low neighbor density in each node where basically anchor range measurements are used to estimate positions. For $R \geq 30$, we see an improvement by a factor of 1.5 to 2.0, which checks with our previous observations for the indoor network.

In Table 4.6 we summarize some statistics of neighbor densities NDs obtained from the 20 WSNs as before described. Also, we present ToA RMSE behaviors at different R using a fixed σ_d=6*m*. This table shows interesting relations among R, RMSE, and ND. For example, if we reduce R from 100*m* to 15*m* (a factor of 6.25), the mean neighbor density ND is also reduced from 93.79 to 6.2 (a factor of 15.12), and the mean error is increased from 3.1*m* to 5.85*m* (a

factor of 1.81). This shows robustness under highly constrained scenarios. Also, RMSE results imply that we can use shorter ranges with significant energy savings at the expense of little loss in accuracy performance if this is convenient.

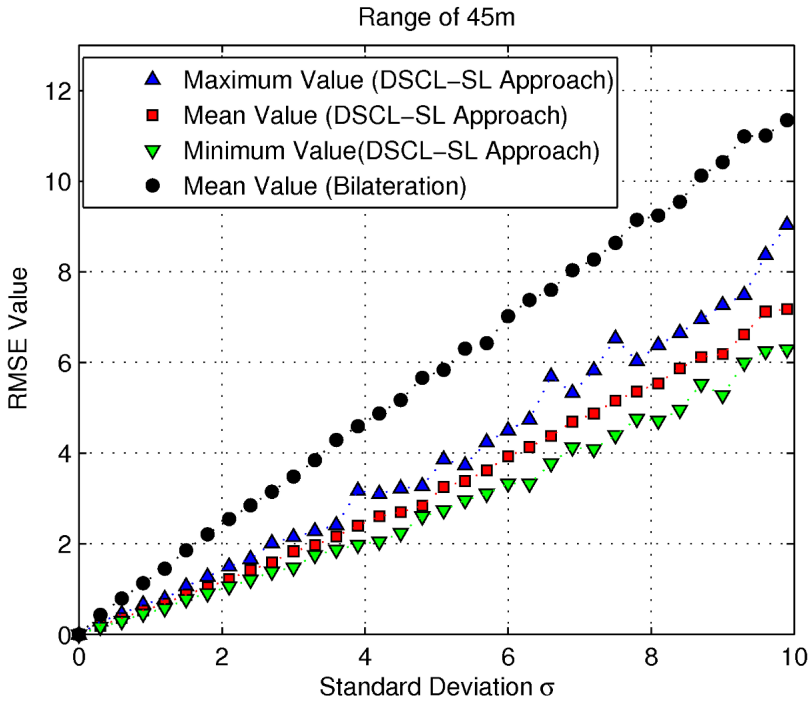
Table 4.6: Effects of ranging nodes in the accuracy of the estimated positions using 100 sensor nodes with the DSCL-SL scheme.

	RMSE (ToA)			NEIGHBOR DENSITY				
	$\sigma_d = 6m$							
Range	min	mean	max	median	mean	mode	min	max
100m	2.61	3.10	3.91	95	93.75	95	81	95
30m	4.10	4.73	5.77	21.5	21.2	23	8	35
15m	5.03	5.85	6.55	6	6.2	5	1	14

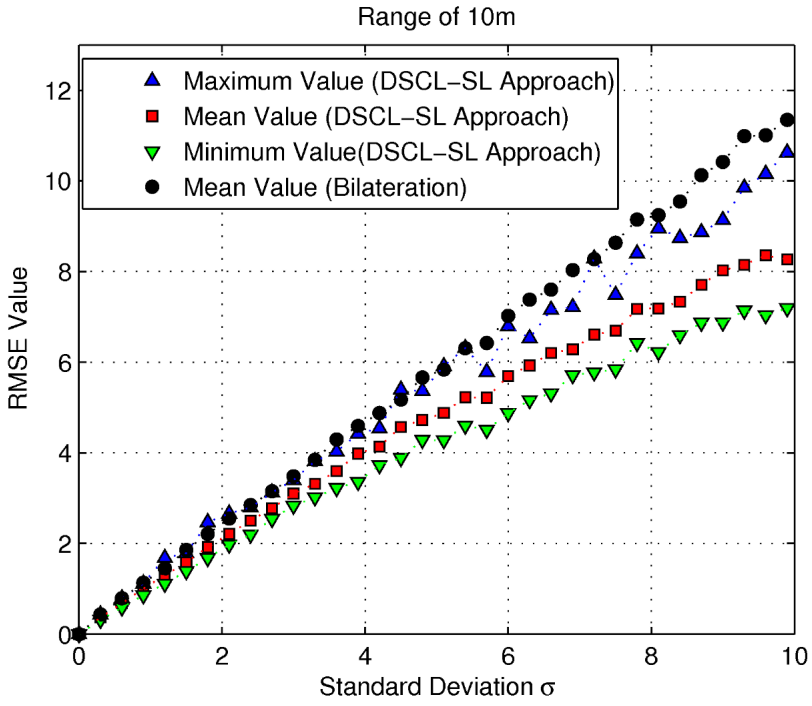
As final point we compare both parameters R and ND individually to analyze which one affects more the estimated positions (such as the case RSS). We take as basis the last row from Table 4.6 which provides information for both parameters R=15m and ND$\approx$ 6 nodes. Next, we add new nodes in a random way to the 100 already deployed nodes. The new WSNs are tested considering R=15m, and Table 4.7 shows how the accuracy of the estimated positions are also improved by increasing the neiboring density. However, we can observe that, if we double the range from R=15m to R=30m (taking as basis the the last row from table 4.6), the mean RMSE improves from 5.85m to 4.73m while if we double the number of nodes from 100 to 200, we can realize that the mean RMSE improves from 5.85m to 5.26m implying that increasing the range of the nodes present a more relevant effect on the estimated positions than increasing the neighbor density. However, we should take into account doubling that the range of coverage on sensor nodes from R=15m to R=30m, for the last case, also increased ND from 6 to 21 (a factor of 3.5), from which we conclude that R and ND are closely related and important aspects like energy consumption (i.e., network lifetime), node prices, and robustness always should be contemplated when deciding if a high density or large range has to be used. The next chapter discusses the modification of these algorithms to achieve energy efficiency in localization.

Table 4.7: Effects of neighbor densities on the accuracy of the estimated positions using R=15m using the DSCL-SL scheme.

	RMSE (ToA)			NEIGHBOR DENSITY				
	$\sigma_d = 6m$							
Nodes	min	mean	max	median	mean	mode	min	max
100	5.03	5.85	6.55	6	6.2	5	0	15
150	4.98	5.48	6.07	9	9.1	10	1	21
200	4.56	5.26	5.62	12	12.2	12	2	25



(a)



(b)

Figure 4.14: Algorithm performance under different error measurements using different short ranges on regular sensors considering full connectivity on anchors.

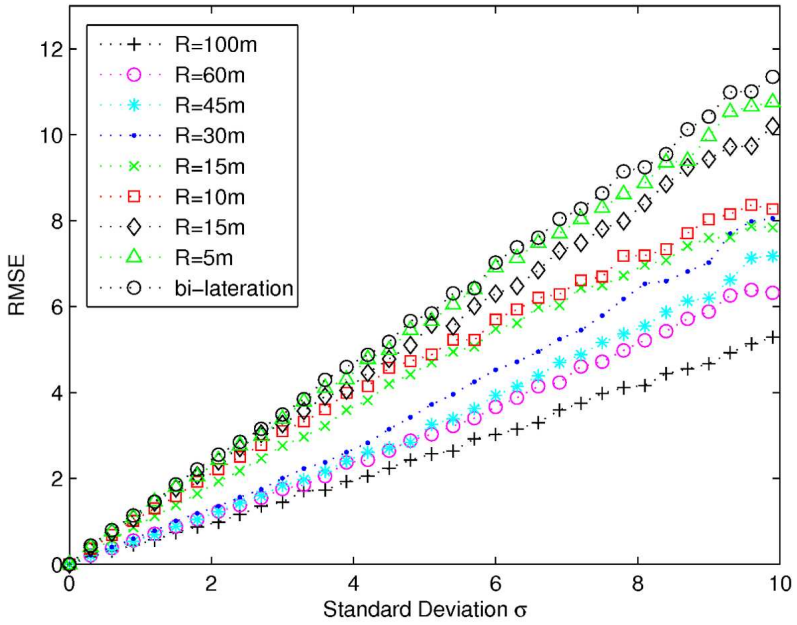


Figure 4.15: Localization accuracy at different short ranges in regular sensors.

Chapter 5

An Analysis of the Trade-off Between Energy Consumption and Localization Accuracy Using Stopping Criteria

We have introduced a type of distributed iterative localization algorithm that relies on local search regions that allows all nodes to update their positions simultaneously. In Chapter 2, we also introduced a bilateration algorithm for location initialization (or seeding) and compared it to a closed-form least-square (LS) scheme and an iterative LS scheme using the Levenberg-Marquardt (LM) algorithm.

Up to this point, we have only considered low computational complexity and high localization accuracy as the performance metrics for these algorithms. Although these are important metrics for WSNs, it is also important to evaluate them within the context of energy consumption. Perhaps, the most important constraint of a WSN is the battery life of the nodes. The energy (or power) consumption is crucial for the lifetime of the network; it is typically assumed that batteries can never be replaced nor the use of energy scavenging methods can provide the required power over long periods.

There are different approaches to minimize energy consumption in WSNs. First, the design of low power hardware with low operating voltage or varying/frequency parameters has been subject of attention [68]. Second, the design of efficient software that accounts for sleep or low duty hardware cycles, and the use of energy power protocols are also of relevance. Third, the use of collaborative and distributed computing algorithms that maximize the amount of in-sensor computing while minimizing the amount of communication among nodes has been extensively

studied. This last criterion is where most energy savings can be expected on the long run, since at some point the hardware and software is "frozen," and no further improvements are possible at this level.

This maximum computation versus minimum communication paradigm is based on the fact that the energy cost of wireless communications can be orders of magnitude higher than the energy cost of computation [55]. Since our proposed algorithms are iterative (and collaborative) in nature and require wireless messages to transmit position updates, then a key requirement is to minimize the amount of iterations (i.e., wireless transmissions), required to achieve acceptable localization accuracy. In this chapter we introduce the use of a stopping criterion in our iterative algorithms as a simple and effective method to limit the overall energy consumption of the WSN. In Section 5.1, we analyze the impact of reducing the number of wireless transmissions (e.g., using a stopping criterion) on localization accuracy. In Section 5.3, we present a detailed analysis of energy consumption of a WSN node. This section uses an energy model that allows characterizing, at a very detailed level, the energy consumption in nodes through simulations.

5.1 Design Criteria for Energy-Aware in WSN

Iterative methods, in essence, require the repetition of a procedure J times to find the solution to a problem. If the localization algorithm converges, each iteration takes us closer to a solution while the improvement on the solution starts to decrease. In a WSN, it is assumed that motes have a finite amount of energy (i.e., non renewable) which should be optimized to extend the network lifetime. For each mote, an iteration involves processing and wireless transmissions/receptions which in turn reduces its energy budget.

A trade-off between energy consumption and accuracy can be achieved with the use of a stopping criterion that controls the number of iterations incurred. In this section we evaluate the accuracy (RMSE) versus energy efficiency when the following stopping criterion is used in our

proposed iterative-distributed algorithms:

$$\|\mathbf{p}_i^{\ell+1} - \mathbf{p}_i^{\ell}\| \leq \tau. \tag{5.1}$$

At each iteration, each sensor s_i compares its current and previous position estimate. As long as the change in position is larger than τ (in meters), the sensor keeps updating and broadcasting its position to its neighbors. If a sensor s_i satisfies the stopping criterion (5.1) after J_i iterations, it will transmit its final estimate $\mathbf{p}_i^{J_i}$ with a "*stopping flag*" indicating that it will not longer transmit a new position. Its remaining neighboring nodes should use $\mathbf{p}_i^{J_i}$ on further position updates. In this way, all nodes will gradually stop the process of updating/broadcasting their positions.

5.2 Impact of a Stopping Criterion on Distributed Localization Algorithms

As can be expected, as larger values for τ are used, on the average, the sensors will require a lower number of iterations while producing location estimates with larger errors. In this section we present an extensive evaluation of this tradeoff. We perform a similar set of simulations as in the previous chapter with the difference of adding (5.1) to Algorithm 3. Hence, a mote will stop if τ is met or the maximum number of iterations is reached (set to 100 in our case).

As part of this evaluation, we should consider that the initial point for an iterative algorithm is closely related with its robustness and convergence, so in this section we also evaluate the impact of the initialization algorithms discussed in Chapter 2 on the performance of our iterative approaches. The initialization RMSEs are summarized in Table 5.1 over the 20 test WSNs used previously. For instance, on the average (over 20 WSNs), the Bilateration algorithm has an average localization error (RMSE) of 12.96 meters with a standard deviation of 0.84 meters per mote.

In our simulations, we are using the same set of ranging measurements for the 20 WSNs presented in Section 2.4. Given a set of initial estimates (provided by initialization algorithms)

Table 5.1: Mean RMSE and standard deviation for initial estimates of 20 networks.

Initialization Algorithm	Mean and SD over 20 WSN nodes	
	Mean RMSE	SD
LS Multilateration	22.7m	2.22m
Bilateration	12.96m	0.84m
LM	12.54m	0.69m
TRR	12.53m	0.68m

shown in Table 5.1 and a fixed threshold τ, each node s_i refines its position $\mathbf{p}_i^{\ell}$ using one of our distributed spatially constrained localization (DSCL) algorithms until either equation (5.1) is satisfied or a maximum number of iterations (bounded to 100) is completed.

Using DSCL with local objective functions (DSCL-L), we performed the localization process for each one of the 20 networks over different values of $\tau \in [0, 0.5]$. The particular case of $\tau = 0$ refers to the situation where each mote will perform 100 iterations as before. The average RMSE (over the 20 WSNs) vs. τ is presented Figure 5.1. There are four plots in the figure, "■" presents RMSE values using the closed-form LS algorithm for initialization, "•" presents the bilateration results, and the "▲" and "►" curves represent the LM and the TRR algorithms respectively. For the last two cases, we can see that both initialization algorithms provide the same initial position estimates, as shown Figure 2.4, so the final position estimates are also the same, and they are overlapped.

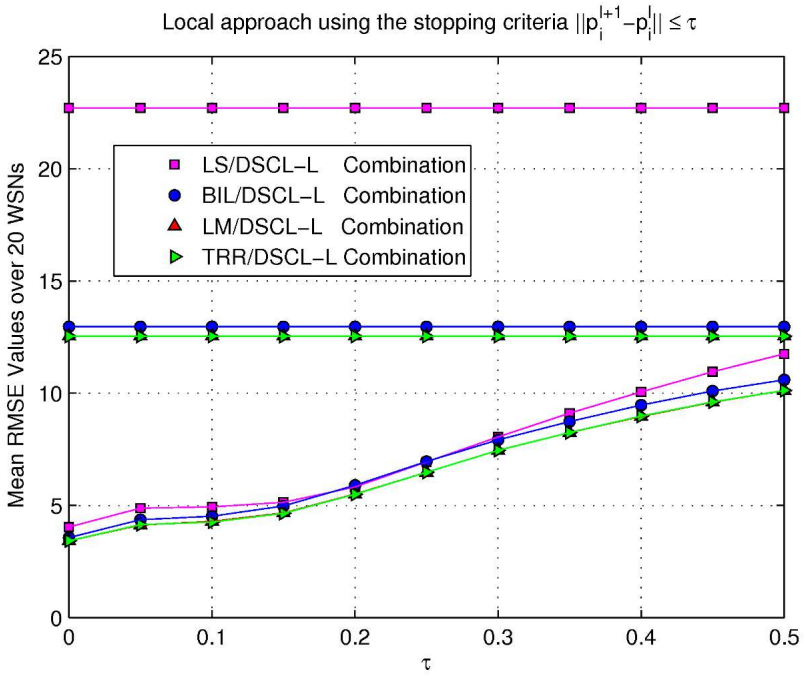


Figure 5.1: Accuracy versus stopping criteria in the iterative DSCL-L scheme at different initial estimates.

As expected, Figure 5.1 shows higher RMSE values when τ is incremented as sensors node stops its iterative process earlier. We can observe that better position estimates are obtained using the LM and the TRR initialization. However, when the bilateration scheme provides the initial estimates, final RMSE values are very close to the last two schemes specially at lower values in τ. Also, we observe that the worst scenario is presented when the LS multilateration provides the initial estimates. However, position estimates are greatly improved indicating that DSCL-L shows robustness to poor initial estimates. For example, the mean RMSE value over 20 networks provided by the LS multilateration is 22.7*m* as shown Table 5.1, and the DSCL-L approach improves these initial estimates reaching a mean RMSE value of 4.04*m* for a τ=0. Figure 5.2 shows the standard deviation of the 20 RMSE values for each τ. All cases show narrow variations indicating that the iterative DSCL-L algorithm is well behaved.

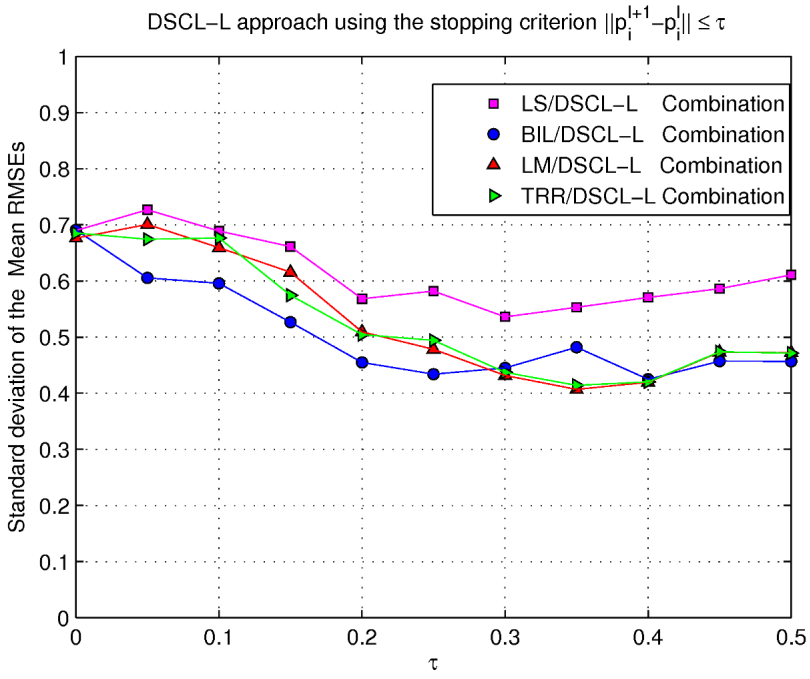


Figure 5.2: SD of the mean RMSE values.

Next, we analyze the relation between the number of wireless transmissions and position accuracy considering different thresholds in the stopping criterion (5.1). Let us define Ψ_{ij}^{τ} as the maximum number of wireless transmissions, n, spent by the sensor s_i in the network j to reach its final estimate $\mathbf{p}_i^n$, where $i = 1,\ldots,96$, $j = 1,\ldots,20$, and τ is varied between 0 and 0.5. Then

$$\varsigma_j^{\tau} = \frac{1}{96}\sum_{i=1}^{96}\Psi_{ij}^{\tau} \tag{5.2}$$

is the average wireless transmissions for all sensors belonging to network j with a fixed threshold τ. Next, we define

$$\xi^{\tau} = \frac{1}{20}\sum_{j=1}^{20}\varsigma_j^{\tau}, \tag{5.3}$$

as the average number of transmissions per mote over the 20 simulated WSNs.

Figure 5.3 plots ξ^{τ} at different thresholds. As can be seen, DSCL-L spends practically the same number of wireless transmissions when the bilateration, LM, and TRR schemes provide

initial estimates. Also, it is easy to see how increasing τ decreases the number of wireless transmissions which should lead to energy savings (as discussed later on this chapter). On the other hand, for the closed-form LS scheme, an additional 1000 wireless transmissions (on average) are generated for a given τ. We deduce that more iterations are required given the poor initial estimates of LS multi-lateration. Figure 5.4 shows the standard deviation (SD) of ξ^{τ} at different τ values. Clearly the DSCL-L approach provides lower standard deviations when using the TRR or the LM as initial estimates. Also, the DSCL-L approach generates the best position estimates for these two schemes followed by our proposed bilateration algorithm, as shown Figure 5.1.

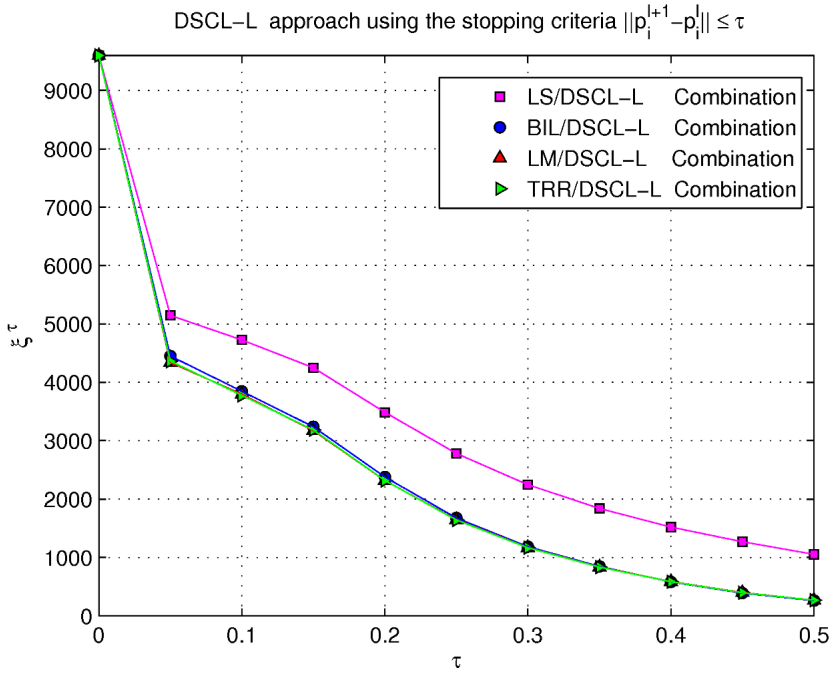


Figure 5.3: Wireless transmissions average (ξ^{τ}) at each τ using the iterative DSCL-L scheme.

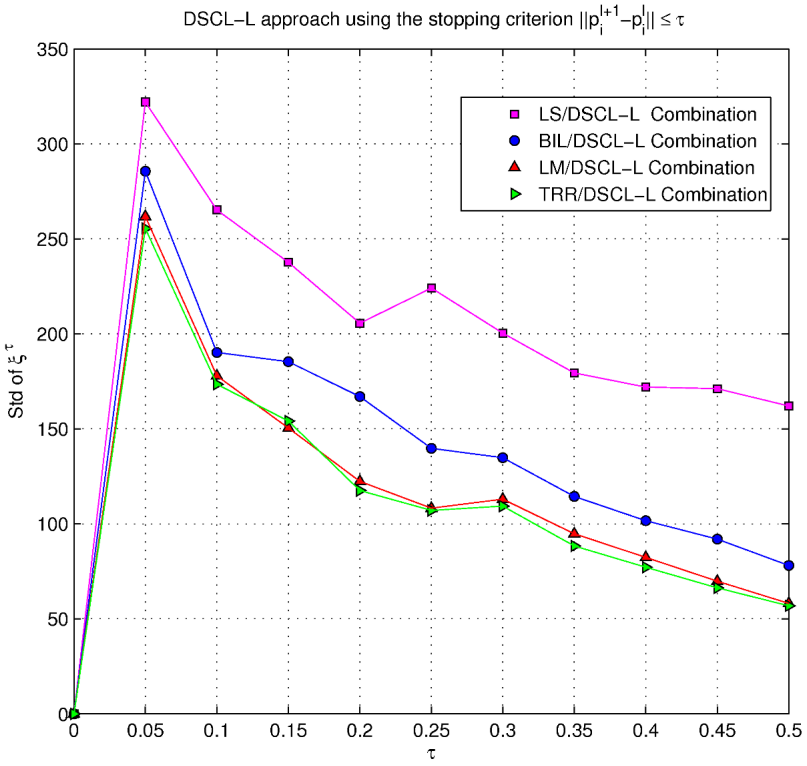


Figure 5.4: Standard deviation of ξ^τ at each τ.

Figure 5.5 summarizes localization accuracy versus energy efficiency, considering only wireless transmissions using DSCL-L at different thresholds τ. We simplify our analysis considering only transmissions, and present a more detailed (transmission/reception) analysis on the next section. Clearly, we can observe that the best performance happens when the LM or TRR algorithms provides the initial estimates. Note that at $\tau = 0$, we achieve the lowest RMSE of 3.4 meters with 9600 transmissions (96 sensors broadcasting their updated 100 times). As before, there is a dependence on the initialization algorithm with the same order of performance.

What is more relevant about Figure 5.5 is the effect of the threshold over performance. We note that for $\tau = 0.05$ (i.e., 5 cm.), the number of radio transmissions is halved while the average localization performance decreases about 0.5 meters. For $\tau = 0.05$, we loose around a meter in average accuracy while reducing the number of transmission to about a third with respect to $\tau = 0$. This leads us to the question on which of the two metrics is more important given the

energy constraints of a WSN. Given that the error for low cost GPS is on the order of four to five meters, we can argue that relaxing the value of τ to 0.15 meters should lead us to satisfactory localization performance at a much lower energy cost. Moreover, if the quantities measured by the WSN are slow to medium varying, the effect of localization error would be attenuated by the characteristics of the signal.

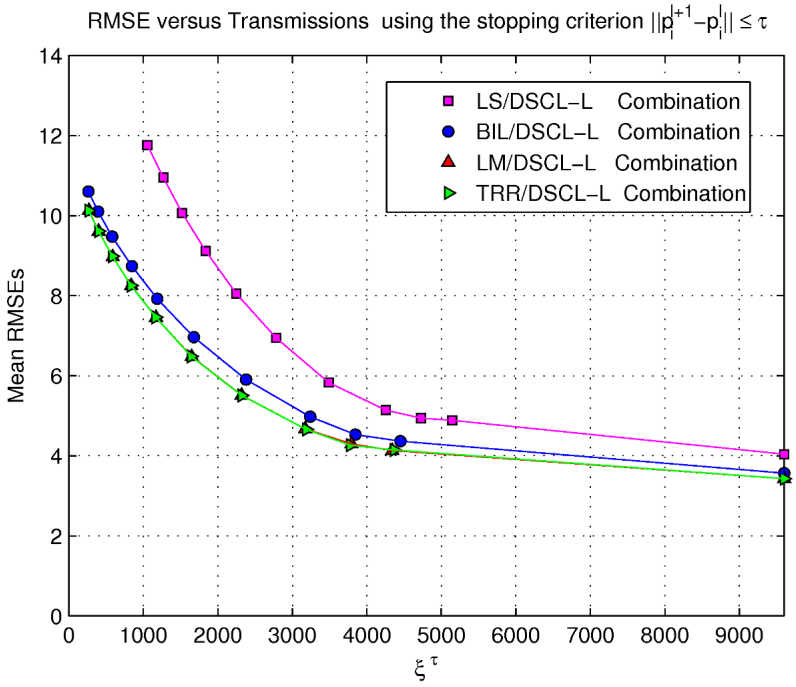


Figure 5.5: Accuracy versus wireless-transmissions for the iterative DSCL-L approach at different initial estimates.

Using DSCL with sub-local objective functions (DSCL-SL) presented similar RMSE behavior in the sense that as reducing τ, the RMSE also is reduced as shown Figure 5.6. Clearly, the DSCL-SL approach provides the lowest RMSE values when using the TRR and LM initial estimates. They are around $0.6m$ below the bilateration and around $0.8m$ below the LS approach on average. Moreover, RMSE standard deviations for both initialization approaches, the TRR and LM, are lower than the bilateration and LS multilateration schemes, as shown Figure 5.7.

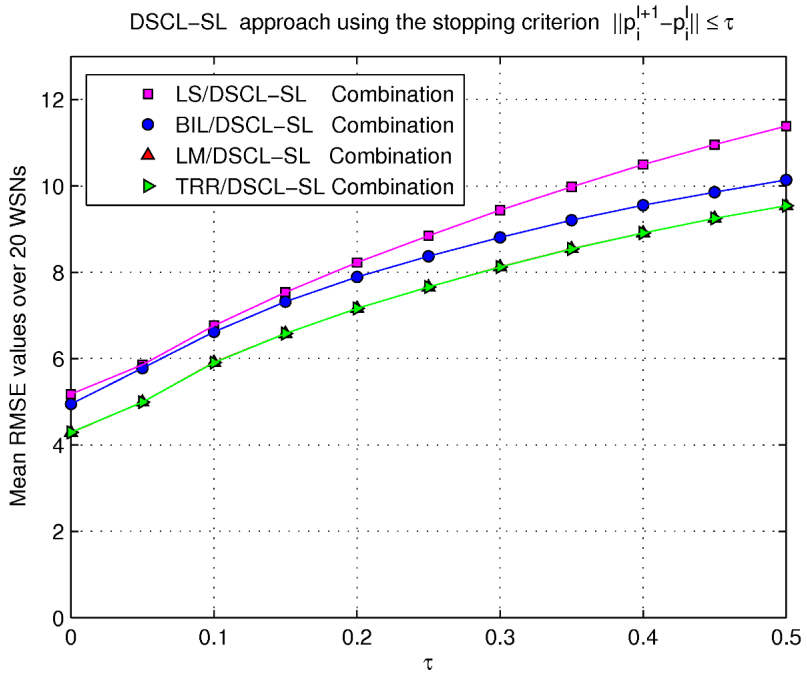


Figure 5.6: Accuracy versus stopping criteria in the iterative DSCL-SL scheme at different initial estimates.

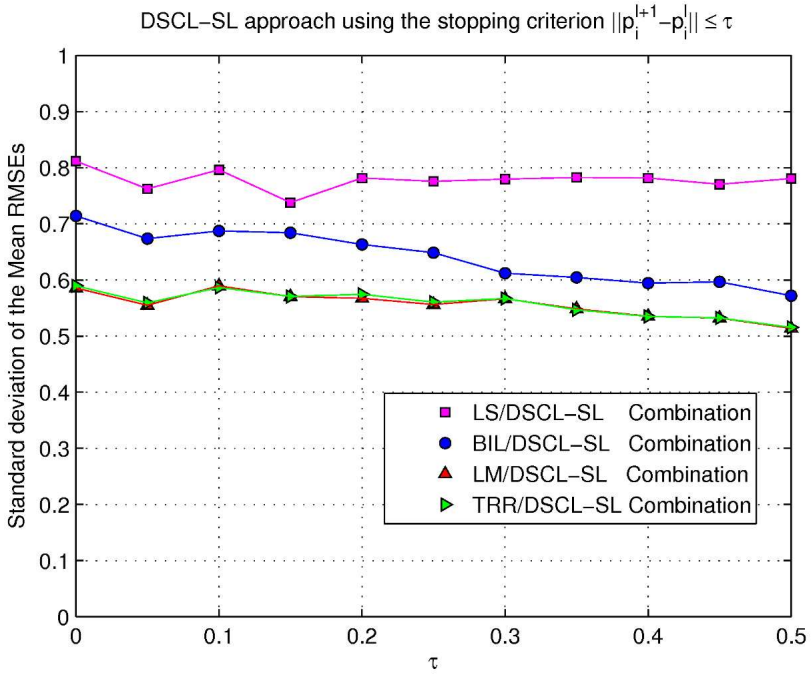


Figure 5.7: SD of the mean RMSE values.

Figure 5.8 shows ξ^{τ} at different thresholds τ using the DSCL-SL algorithm. We can see that the DSCL-SL approach uses the same number of transmissions when using the LM, TRR, and bilateration initial estimates. However, the RMSE for each case is different as shown Figure 5.6. Also, when LS multilateration is used, it requires around 800 iterations above other schemes for a given threshold. However, we should consider that initial estimates for this method provided, in average, an initial error of 22.7*m* as shown Table 5.1. We can observe that, similar to DSCL-L, this iterative algorithm is robust to bad initial estimates.

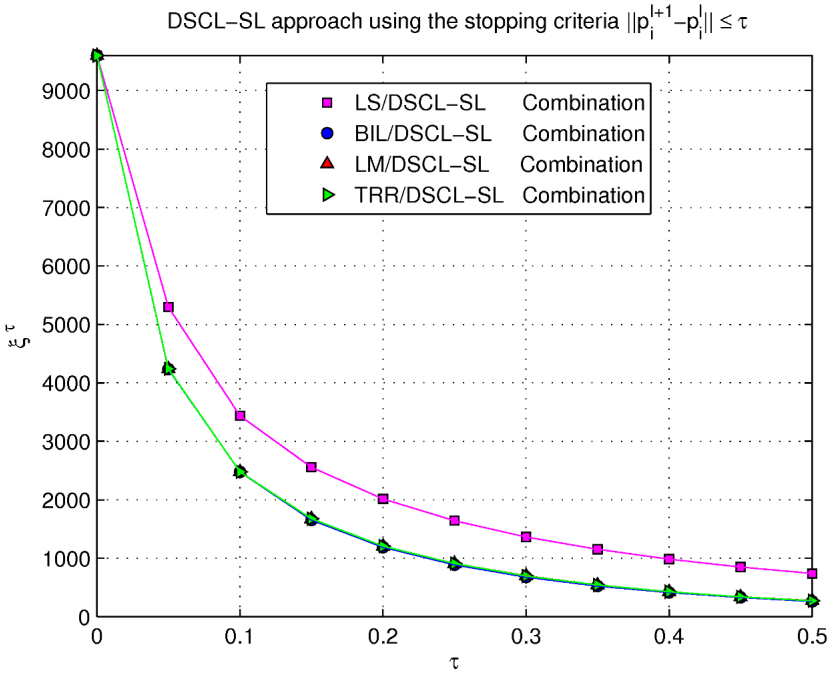


Figure 5.8: Wireless transmissions average ξ^{τ} at each τ using the iterative DSCL-SL scheme.

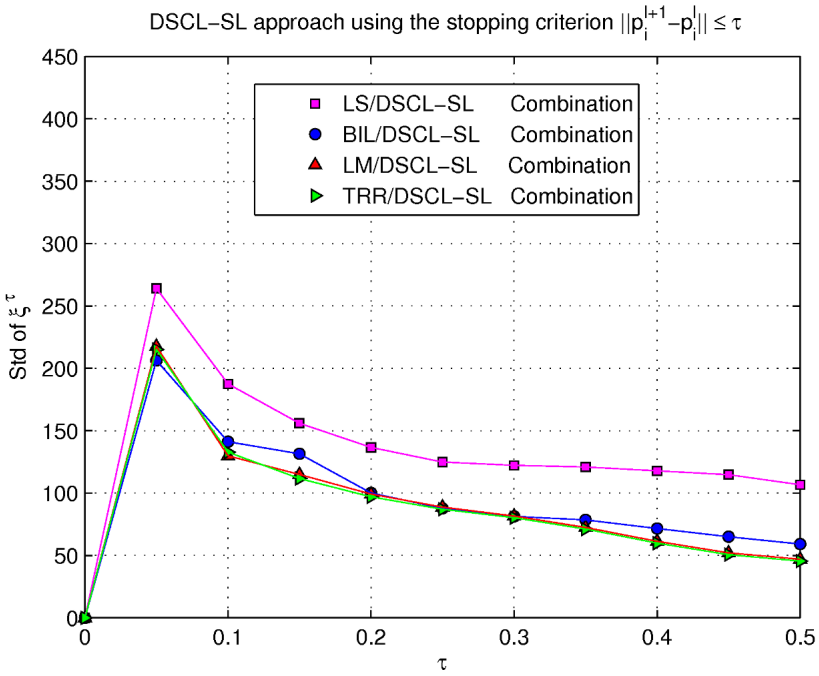


Figure 5.9: SD of (ξ^τ) at each τ.

Figure 5.10 summarizes RMSE vs. number of wireless transmissions. From the figure we conclude that the best tradeoff between position estimates and wireless transmissions happens when initial estimates are provided either by the LM or TRR algorithms. As before, we see that small values of τ lower significantly the number of transmissions while maintaining acceptable localization performance. On the other hand, DSCL-L always performed better that the DSCL-SL counterpart.

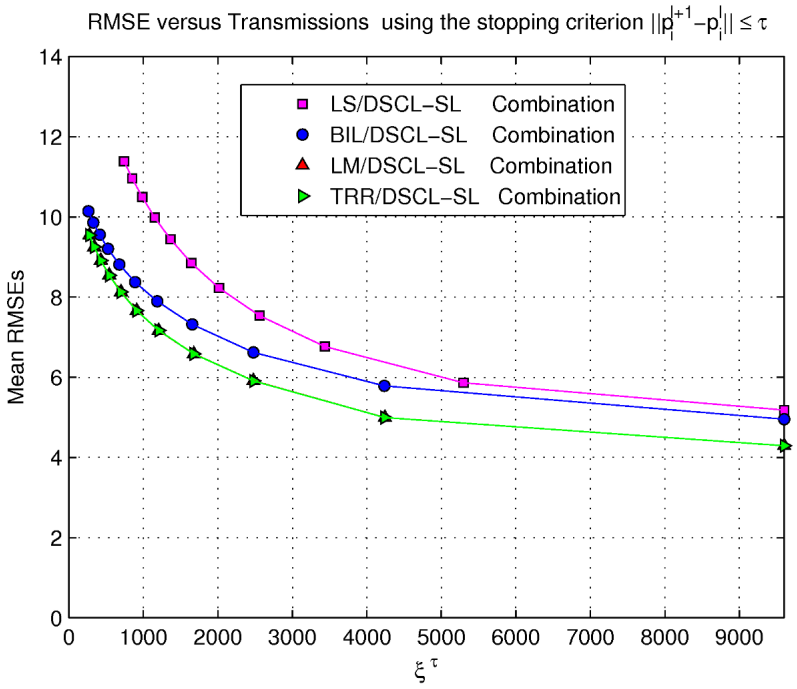


Figure 5.10: Accuracy versus wireless-transmissions for the iterative DSCL-L approach at different initial estimates.

In conclusion, DSCL-L has a better performance than DSCL-SL as reflected by the RMSE vs. wireless-transmission results presented in Figures 5.5 and 5.10 respectively. For example, considering the LM initialization algorithm and 2500 wireless transmissions in both iterative algorithms, DSCL-L provides an RMSE of around $5.32m$ while DSCL-SL provides a RMSE around $5.89m$. This implies that DSCL-L can achieve the same RMSE as DSCL-SL with fewer transmissions. For $\tau = 0$, the LM initialization with DSCL-SL requires 9600 wireless transmissions to achieve an average RMSE value of $4.29m$ while DSCL-L only uses around 3700 wireless transmissions to reach the same accuracy position.

Up until this point, we have only considered one part of the energy consumption in a mote (i.e. wireless transmissions). In the next section, we develop a complete energy consumption analysis for the full end-to-end localization process of our proposed algorithms.

5.3 An End-to-End Cycle Accurate Analysis of Energy Consumption for Localization

5.3.1 An Energy-Consumption Model for Distributed/Collaborative Localization Schemes

There has been considerable work on developing energy efficient paradigms for WSN [68, 27, 60, 37]. We are not aware of an extensive and detailed study of the impact of energy consumption on the localization task and vice versa. In this section, we are interested on a cycle and transmission accurate model that captures energy consumption at the finest level, namely, computation cycles and single bit transmissions. These are common denominators that can be used to compare all possible localization schemes.

Specifically, we adopt the energy model developed by Wang and ChandraKasan [68]. In spirit, they pursue a similar goal. To make our development down to earth, we also adopt as a case of study the node architecture they develop with an SA-1100 processor and a 2.45 GHz radio. The SA-1100 is a 32 bit MCU with more than enough computational resources to execute the localization schemes. In their paper, they evaluate the performance of the SA-1100 over the operating voltage and clock frequency space, providing a voltage-frequency curve that characterized the minimum voltage needed for a given operating clock frequency or vice versa.

The energy model considers three main sources of consumption: 1) computation, 2) leakage under realistic voltage/frequency, and 3) transceiver costs; namely

$$E_{tot} = E_{cyc} + E_{leak} + E_{com}. \tag{5.4}$$

The computational energy in our case reflects the complexity of the localization algorithms in terms of processor cycles. For the end-to-end computational localization process we have

$$E_{cyc} = E_{INIT} + E_{ITER}, \tag{5.5}$$

where E_{INIT} is the energy spent on the initialization algorithm and E_{ITER} is the total energy spent on the distributed-iterative localization scheme. From [68], we have

$$E_{INIT} = N \cdot K_{INIT} \cdot C \cdot V_{dd}^2, \tag{5.6}$$

and

$$E_{ITER} = J \cdot K_{ITER} \cdot C \cdot V_{dd}^2, \tag{5.7}$$

where C is the average switched capacitance (per clock cycle), V_{dd} is the power supply voltage, N is the number of sensor nodes, K_{INIT} the number of cycles needed by a single sensor to execute the initialization algorithm, and K_{ITER} refers to the number of cycles needed by a single sensor to execute one iteration of the iterative algorithm. The value of J refers to the total number of iterations over the full localization process. We note that values of K_{INIT}, J, and K_{ITER} are highly dependent on the network topology and number of local neighbors and hence the communication range. In this sense, these parameters can be seen as random variables. Hence, we can look at the average performance by

$$\begin{aligned} \overline{K}_{ITER} &= \varepsilon\{K_{ITER}\} \\ \overline{K}_{INIT} &= \varepsilon\{K_{INIT}\} \\ \overline{J} &= \varepsilon\{J\} \end{aligned} \tag{5.8}$$

and using these expected values in equations (5.6) and (5.7).

Similar arguments can be used to derive the Leakage energy which is expressed by the following general formula [68]:

$$E_{leak} = (V_{dd}, f, K) = V_{dd}\left(I_0 e^{\frac{V_{dd}}{nV_T}}\right)\left(\frac{K}{f}\right), \tag{5.9}$$

where f is the clock frequency, K is the number of cycles for a given task, and I_0, n, V_T are microprocessor dependent parameters. We have performed an analysis of the number of processor cycles for the initialization and iterative algorithms. In further sections we will provide explicit parameterizations.

The cost of wireless communications is known to be substantially larger than computation. Wang and Chandrakasan [68] cite a 1 to 150 instruction to transmitted bit cost. Hence, for a 100 byte package, it would take approximately $150 \times 8 \times 100$ processor cycles to spend the same energy. The transmitter/receiver cost consists of two components. For transmission of a k-bit package we have

$$E_{TX}(k,d) = E_{elec} \cdot k + \varepsilon_{amp} \cdot k \cdot d^2, \tag{5.10}$$

where $E_{elec} = 50\frac{nJ}{b}$, $\varepsilon_{amp} = 100pJ/b/m^2$, and d is the transmission range. Similarly, for the reception of a k-bit package we have

$$E_{RX}(k) = E_{elec} \cdot k. \tag{5.11}$$

The communication cost for the initialization stage is negligible in our case. However, the iterative schemes have a strong dependence on local wireless messaging. Hence, our development of a stopping criterion is the first line for reducing energy consumption. Let us consider the E_{com} component for a sensor i during iteration i modeled as

$$E_{com}^{(i,j)}(k,d) = E_{TX}(k,d) + \beta_{i,j} \cdot E_{RX}(k). \tag{5.12}$$

At each iteration j, a sensor s_i broadcast its position update and receives $\beta_{i,j}$ messages from its neighbors. We can expect that $E_{com}^{i,j} \to 0$ as $j \to J_{max}$, or similarly, $\beta_{i,j} \to 0$ as $j \to J_{max}$, so the total transceiver energy for a sensor i can be denoted as

$$\begin{aligned} E_{com}^{(i)} &= \sum_{j=1}^{J_i} \left(E_{TX}(k,d) + \beta_{i,j} \cdot E_{RX}(k) \right) \\ &= J_i E_{TX}(k,d) + \left(\sum_{i=1}^{J_i} \beta_{i,j} \right) \cdot E_{RX}(k) \\ &= J_i E_{TX}(k,d) + B_i E_{RX}(k), \end{aligned} \tag{5.13}$$

where J_i represents the total number of transmissions required by sensor s_i before it stops updating

its position (i.e., meets the stopping criteria), and

$$B_i = \sum_{i=1}^{J_i} \beta_{i,j}, \tag{5.14}$$

is the total number of wireless receptions for s_i before meeting the stop criteria. We note that J_i and B_i are RVs since it is not possible to know the number of iterations J_i required for the mote to reach the stopping criteria. The issue is more complex for package reception since the term $\beta_{i,j}$ is also a RV that represents the number of active neighbors of s_i at iteration j. A more extreme situation would be to establish an algorithmic policy to stop transmissions if $\beta_{i,j} = 0$ before the stop criterion is reached for sensor s_i. However, this situation is left for further study in the future. Now, we are ready to compute the total communication energy for an N-node network. This is given as

$$E_{com} = \sum_{i=1}^{N} E_{com}^{(i)}(k,d), \tag{5.15}$$

where

$$\begin{aligned} E_{com}(k,d) &= \sum_{i=1}^{N} J_i \cdot E_{TX}(k,d) + B_i \cdot E_{RX}(k) \\ &= J \cdot E_{TX}(k,d) + B \cdot E_{RX}(k), \end{aligned} \tag{5.16}$$

$J = \sum_{i=1}^{N} J_i$ and $B = \sum_{i=1}^{N} B_i$. In this case J represents the total number of transmissions and B the total number of receptions over the complete localization process. We note that both B and J can be measured by motes. We also note that J and B are the sum of N random variables. If we assume that the network properties are uniform over space, then J_i and B_i are assumed to have the same distribution for all i. By the central limit theorem (CLT), both J and N can be approached as Gaussian RVs.

Given the statistical nature of the number of transmissions and receptions, we then consider the communication energy cost $E_{com}(k,d)$ as a RV, and use its expected value as the representative

value “on the average” for the total energy spent by the WSN in localization. Hence we have

$$\begin{aligned} \overline{E}_{com} &= \varepsilon\{E_{com}(k,d)\} \\ &= \varepsilon\{J\}E_{TX}(k,d) + \varepsilon\{B\}E_{RX}(k). \end{aligned} \tag{5.17}$$

We note that this expression provides an average behavior and does not reflect the fact that some sensors will incur on more energy costs. However, simulations indicate that only a small fraction of the nodes will go incur a significant extra cost.

Reviewing our expression for energy we have

$$\begin{aligned} E_{tot} &= E_{cyc} + E_{leak} + \overline{E}_{com}(k,d) \\ &= E_{cyc} + E_{leak} + \overline{J}E_{TX}(k,d) + \overline{B}E_{RX}(k). \end{aligned} \tag{5.18}$$

We note that the value $\overline{J}$ refers to the total number of transmissions in the network and in fact is the same as the sum of the total number of iterations used by each sensor in the network. As we are mostly interested on a comparative analysis, among localization algorithms, we can simplify our expression by removing the E_{leak} factor to finally use

$$E_{tot} = E_{cyc} + \overline{J}E_{TX}(k,d) + \overline{B}E_{RX}(k) \tag{5.19}$$

as the total energy required to perform the complete localization process over a WSN.

5.4 Quantitative Analysis of Energy Consumption in Localization

5.4.1 Energy Consumption of Localization Algorithms

In this section, we look closely at the energy consumption of the localization algorithms used in the initialization and refinement stages, described in Chapter 2 and Chapter 3 respectively. Specifically, we are interested on comparing the bilateration (BL), Levenberg-Marquardt (LM),

DSCL-L, and DSCL-LS approaches. Our objective is to present a detailed analysis of energy consumption based on CPU cycles and the number of wireless transmissions/receptions (Tx/Rx) required during the localization process.

First, we compare the RMSE-Energy performance between our proposed BL and the LM algorithms since the LM algorithm is an efficient optimization algorithm (computationally speaking) and guarantees in the worst of the cases convergence to a local minimal. We analyze in detail energy consumption versus localization accuracy for each of the four possible combinations between initialization and refinement algorithms: LM with DSCL-L, LM with DSCL-SL, BL with DSCL-L, and BL with DSCL-SL.

On previous chapters we analyzed the computational complexity of the initialization and refinement algorithms in terms of the four basic operations (i.e., ADD, MUL, DIV, and SQRT). Furthermore, in Section 5.3, we presented a detailed analysis for energy consumption in terms of CPU cycles and wireless Tx/Rx. Hence, to complete the energy analysis, we need the number of CPU cycles required for the four basic operations as floating point operations. These values are highly dependant on the architecture of the mote processor. They also depend on whether they are implemented in software or if there is a floating-point unit. Also, they need to be determined through extensive profiling using a cycle-accurate simulator.

For example, for the ATmega32 microcontroller, we found that floating addition takes about 800 cycles using software emulation, similarly we have 2000, 1500, and 700 for MUL, DIV and SQRT respectively. An extensive study in [19] provides good representative values for processors with some level of hardware support. The values are summarized on Table 5.2, and it shows the relation between basic operations and CPU cycles. We have selected these values for all our subsequent simulations.

Table 5.2: Operation cycle counts

ADD	MUL	DIV	SQRT
11	25	112	119

Next, we use Table 2.1 and Table 2.2 to obtain the number of CPU cycles required by each

initialization stage, LM, and BL, respectively. For the LM initialization stage we are using M=4 anchors and the random variables T and k. We recall that k is the number of iterations spent by the LM algorithm to find a solution. These values are obtained through simulations where $\varepsilon\{T\} = 2$ and $\varepsilon\{k\} = 13$. In this way the total cycles required by the LM algorithm according to equation (2.46) is given by

$$\begin{aligned} K_{LM} = \quad & \varepsilon\{k\}\,[((M+4)\,\varepsilon\{T\} + 15M + 7)\,(11) + \\ & ((M+2)\,\varepsilon\{T\} + 12M + 18)\,(25) + \\ & (M+1)(112) + \\ & (2M+4)(119)\,] = 63063\,\text{cycles}. \end{aligned} \tag{5.20}$$

Similarly, the total number of cycles used by the BL stage is given by equation (2.47) as

$$\begin{aligned} K_{BL} = \quad & (11)(11Q + 3S) + \\ & (25)(12Q + S) + \\ & (112)(3Q) + \\ & (119)(2Q) + \\ & (N_{SORT}) = 9965\,\text{cycles}, \end{aligned} \tag{5.21}$$

where $Q = 6, S = 15$, and $N_{SORT} = 2750$. The value for N_{SORT} represents the total number of cycles required to perform the sorting step of the BL algorithm. This step can be performed using efficiently the Quickselect algorithm [42]. The value for N_{SORT} was obtained from a profiling study done for several sorting algorithms [65] implemented on microcontrollers similar to the SA-1000. As expected, the LM algorithm is more expensive (computationally speaking) than de BL scheme.

Given the total cycle counts for each initialization algorithm, we can apply equation (5.6) to compute their energy consumption. For convenience, we restate the equation as for initialization stages according to the next equation:

$$E_{INIT} = N \cdot K_{INIT} \cdot C \cdot V_{dd}^2, \tag{5.22}$$

where $N = 96$ represents the number of nodes in the network, $C = 0.67$nF was obtained experimentally in [60], and $V_{dd} = 0.85$ volts according to [68]. Then, the energy spent by the LM algorithm is $E_{LM} = 2.93$mJ, and the energy spent by the BL algorithm is $E_{BL} = 0.463$mJ. We observe a six to one relationship among both algorithms. Hence, our initial inclination would be to advocate the use of the BL algorithm. However, given the results on the previous sections where using LM improved the performance of the iterative schemes, we need to perform a holistic analysis to see if these improvements offset the additional energy cost of LM initialization.

5.4.2 Computational Complexity of Iterative Localization Algorithms

The next step consists on analyzing the computational complexity of our iterative algorithms. Without loss of generality, the next paragraphs show the analysis for the DSCL-L approach (i.e., local objective functions). The derivation for the DSCL-SL algorithm is essentially the same and we only show the results of the DSCL-L analysis.

Given that the algorithms are iterative, and that each iteration is closely tied to broadcast and reception of position updates, it should not be a surprise that the computational complexity of our distributed algorithms is also tied to the random variables B_i and J_i, introduced on the previous sections (see equations (5.13) and (5.14), respectively). Using the RSS propagation models with our WSN simulator, we estimated the the expected values for B_i and J_i for each s_i. The total number of wireless transmissions J and receptions B are simply given by equation 5.16. This estimation process was repeated for a range of τ between zero and 0.5 meters. This process was repeated over the 20 test WSNs. Table 5.3 shows the estimates for $\varepsilon\{J\}$ and $\varepsilon\{B\}$ obtained as the average over the 20 networks. We see that for all four cases, the number of receptions dominates transmissions making the former a major component of the energy expenses. However, the decay on the number of transmissions as a function of τ is very fast which indicates that a major focus for energy savings should be on finding ways to reduce receptions early on the iterative process.

Let us to consider Table 5.4 which shows the number of basic operations required by a sensor s_i at iteration j for the DSCL-L algorithm. Using Table 5.2, we can compute the number of CPU

Table 5.3: Average wireless transmissions and receptions for the complete localization process.

Initialization	Expectation	$\tau = 0$	$\tau = 0.05$	$\tau = 0.1$	$\tau = 0.15$	$\tau = 0.2$	$\tau = 0.25$	$\tau = 0.3$	$\tau = 0.35$	$\tau = 0.4$	$\tau = 0.45$	$\tau = 0.5$
LM-DSCL-L	$\varepsilon\{B\}$	192300	62359	51827	42100	29982	20066	12783	8146	4887	2612	1312
	$\varepsilon\{J\}$	9600	4481	4345	3970	3691	3269	2764	2270	1844	1536	1265
Bilateration-DSCL-L	$\varepsilon\{B\}$	192300	63361	52286	42806	30644	20106	12979	8157	4626	2385	1178
	$\varepsilon\{J\}$	9600	4620	4456	4032	3762	3351	2820	2318	1902	1560	1281
LM-DSCL-SL	$\varepsilon\{B\}$	192300	59757	31610	19133	12252	8133	5486	3728	2565	1764	1260
	$\varepsilon\{J\}$	9600	5699	3755	2720	2089	1677	1370	1135	959	815	698
BL-DSCL-SL	$\varepsilon\{B\}$	192300	59436	31311	18883	12051	7902	5270	3562	2450	1672	1166
	$\varepsilon\{J\}$	9600	5690	3739	2702	2088	1655	1346	1116	941	797	677

cycles required for a single iteration j of the iterative algorithm at node i as

$$\begin{aligned} K_{ITER}^{(i,j)} = \quad & \big((P+1)\left(\beta_{i,j}+M-2\right)+1+3\left(\beta_{i,j}+M\right)+3\big)\,(11)+ \\ & \big(2(P+1)(\beta_{i,j}+M)+2\big)\,(25)+ \\ & (112)+ \\ & \big((P+1)\left(\beta_{i,j}+M\right)+1\big)\,(119), \end{aligned} \tag{5.23}$$

where $M = 4$ is the number of anchors, $\beta_{i,j}$ is the number of neighbors (i.e, receptions), and $P = 25$ is the number of points inside the search region Ω_i as defined in Algorithm 3.

Table 5.4: Basic operations in the DSCL-L approach in a single iteration.

Operation	Number of operations
ADD	(P+1)($\beta_{i,j}$+M-2)+1+3($\beta_{i,j}$+M)+3
MUL	2(P+1)($\beta_{i,j}$+M)+2
DIV	1
SQR	(P+1)($\beta_{i,j}$+M)+1

Finally, the total number of cycles for the DSCL-L algorithm over the complete localization process can be computed as follows:

$$K_{ITER} = \sum_{j}^{N}\sum_{i}^{J_i} N_{Local}^{(i,j)}, \tag{5.24}$$

where N is the number of sensors on the network and J_i is the number of iterations needed by sensor s_i before the stop criteria is met. As before, $\beta_{i,j}$ and J_i are random variables that depend on the (random) spatial distribution of the nodes and the initial estimates, provided by the BL and LM algorithms. Thus, K_{ITER} should be determined through simulation. Table 5.5 shows the estimated of the expected value (i.e., mean value) of the total number of CPU cycles for an N-node network at different values of τ. The estimates are obtained as the average over the 20 WSNs. Table 5.5 also presents the cycle estimates for the DSCL-SL which were derived following a similar methodology.

Table 5.5: Expected total number of CPU cycles in a N-node network using the DSCL-L and DSCL-SL approaches at different thresholds and initial estimates. The expected values are multiplied by 10^7.

Initialization	Refinement	$\tau = 0$	$\tau = 0.05$	$\tau = 0.1$	$\tau = 0.15$	$\tau = 0.2$	$\tau = 0.25$	$\tau = 0.3$	$\tau = 0.35$	$\tau = 0.4$	$\tau = 0.45$	$\tau = 0.5$
LM	DSCL-L	1.3865	0.6715	0.5874	0.4992	0.3789	0.2791	0.2868	0.1525	0.1144	0.0835	0.061
	DSCL-SL	1.3483	0.6633	0.3907	0.2714	0.1985	0.1528	0.1191	0.0955	0.0773	0.0629	0.0525
Bilateration	DSCL-L	1.3865	0.6948	0.5965	0.5132	0.3893	0.2837	0.2148	0.1598	0.1195	0.0834	0.0592
	DSCL-SL	1.3483	0.6714	0.3961	0.2690	0.1958	0.1495	0.1160	0.0927	0.0762	0.0628	0.0519

5.4.3 Overall Energy Consumption of Localization Process

Based on our extensive development of the energy model, the final step is to combine equations (5.5),(5.18) (5.17), (5.22), (5.24), and (5.25) to determine the total energy consumed. The final expression is

$$E_{TOT} = E_{INIT} + E_{ITER} + \overline{J} \cdot E_{TX}(k,d) + \overline{B} \cdot E_{RX}(k), \tag{5.25}$$

where E_{INIT} could be E_{LM} or E_{BL}, and E_{ITER} refers to the DSCL-L or DSCL-SL iterative schemes. We evaluate E_{TOT} for different values of τ assuming RSS range measurements with $d = 30$ and $k = 56$ bits. The value of k was determined based on the discussion presented in [30].

Figure 5.11 shows the relation between RMSE (localization error) and τ for each combination of initialization-refinement algorithms. These results summarize the content of Figure 5.1 and Figure 5.6. We observe that for low values in the threshold, $\tau < 0.30$m, the LM/DSCL-L combination provides the best position estimates closely followed by BL/DSCL-L. In Figure 5.12, we finally present the relationship between E_{TOT} and τ. We see that the DSCL-SL algorithm uses the lowest energy for a given τ. However, this plot is misleading, as there is no evidence that using the same τ implies a uniform assessment of performance.

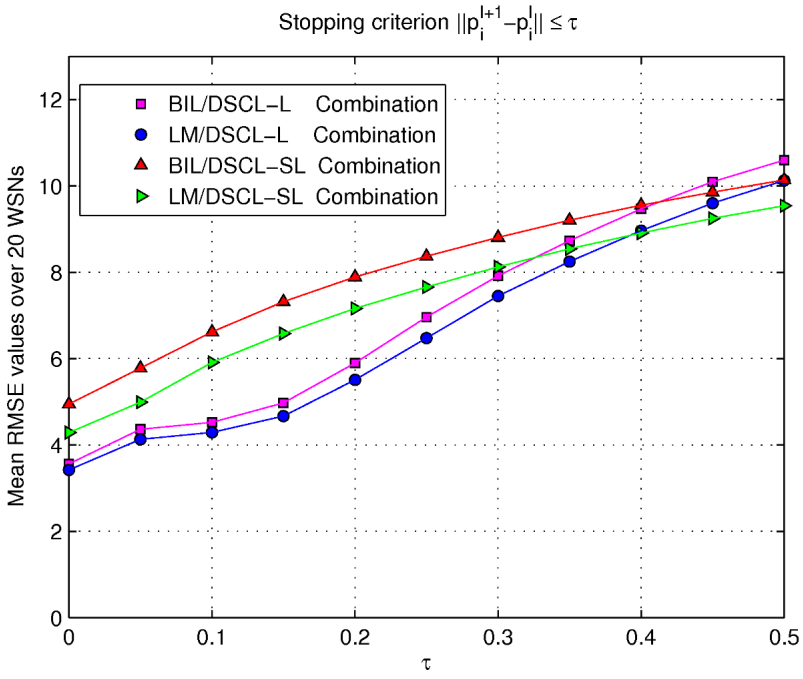


Figure 5.11: RMSE versus τ for the DSCL-L and DSCL-SL approaches at different initial estimates.

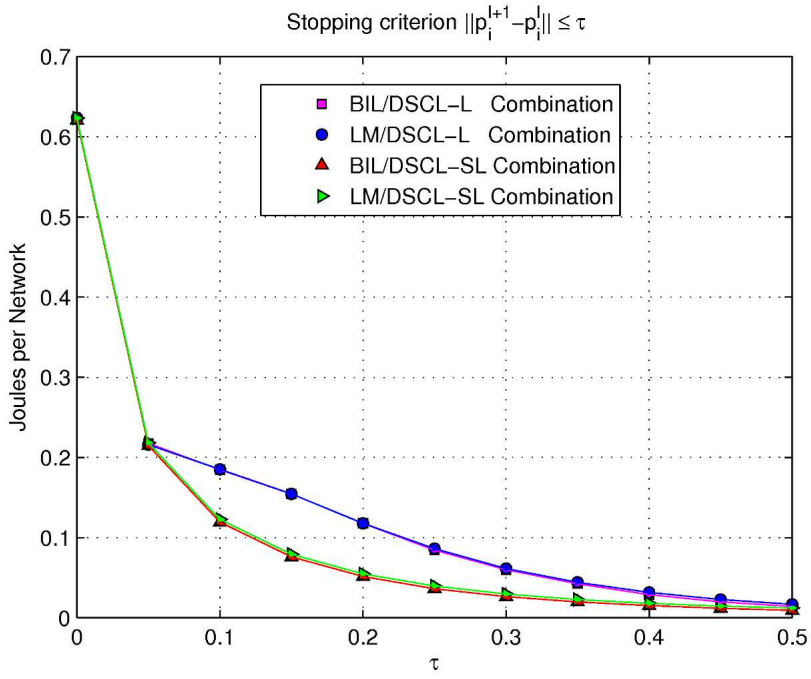


Figure 5.12: Energy versus stopping criteria using the DSCL-L and DSCL-SL approaches at different initial estimates.

If we compare the RMSE-Energy performance between the LM/DSCL-L and LM/DSCL-SL with $\tau = 0.1m$, we can see that the LM/DSCL-L combination provides a RMSE of 4.29m as shown Figure 5.11, but it consumes 0.185 Joules as shown Figure 5.12, while the LM/DSCL-SL combination consumes 0.122 Joules showing a higher RMSE of $5.91m$, so clearly the former combination provides a better RMSE, but the second combination can save more energy. A more appropriate comparison would be to assess the relationship between energy consumption and RMSE. The most desirable localization system would be the one that provides the lowest RMSE, using the lowest possible amount of energy. Figure 5.13 merges Figures 5.11 and 5.12 to detail the relation between RMSE and Energy.

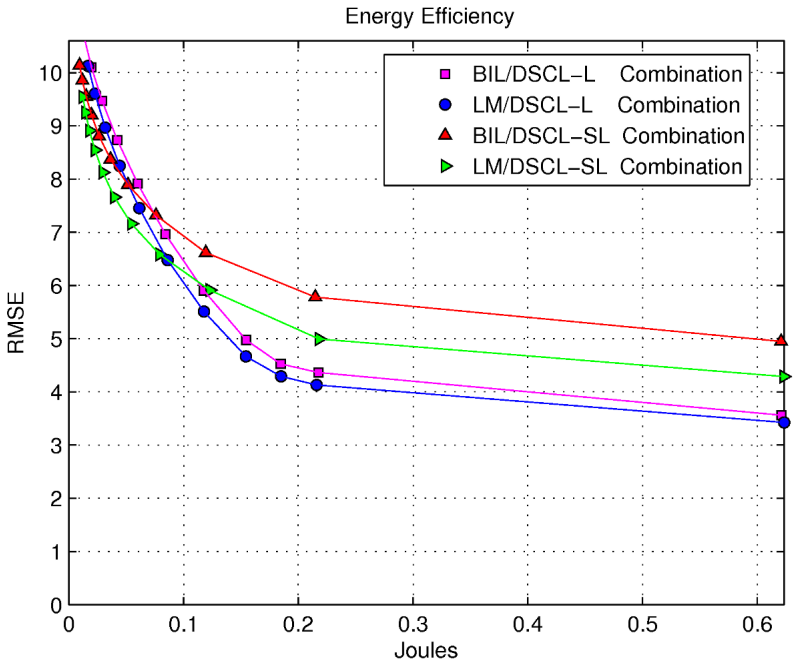


Figure 5.13: RMSE versus Energy for the four combinations of initialization-iterative schemes.

We can appreciate in Figure 5.11 that there is an intersection between the LM/DSCL-L and LM/DSCL-SL performances. It occurs at RMSE=6.5m and 0.07J, so clearly we can observe that for low power consumption or higher RMSEs, the combination LM/DSCL-SL presents the best performance. RMSEs higher than 6.5m in the position estimates may not be very useful in real applications. However in such applications where a larger errors is tolerable, these combination of algorithms should be considered. On the other hand, for RMSEs below to 6.5m the combination LM/DSCL-L provides the best performance, closely followed by the combination BL/DSCL-L approach.

It is interesting to observe that even though the BL scheme required significantly less CPU cycles to obtain initial estimates than the LM algorithm, the latter provides an overall lower power consumption with a better RMSE performance when combined with our iterative algorithms. The energy requirements of the initialization algorithm are negligible when we consider the communication costs and the computational complexity of the iterative schemes. Moreover, the benefits of the LM algorithm are apparent since it provides a better initial point that in turn reduces the

overall number of iterations with a resulting reduction on energy consumption.

5.5 Chapter Conclusions

In this chapter we explored the use of a stopping criterion as a means to control the energy consumption of the distributed iterative algorithms. Our main objective is to achieve the lowest localization error with the lowest energy (or power) expenditure in order to maximize the lifetime of the network. Given the iterative nature of the schemes, our objective was to minimize the number of transmission/receptions.

To provide an objective quantitative comparison among algorithms, we performed an energy analysis that describes the algorithms at the level of computing cycles and single bit transmission. An energy model was derived based on other well known model, and evaluated through simulations over 20 test WSNs. We found that the quality of the initial position estimated plays a crucial role on reducing the overall number of wireless messages. Hence, using the LM algorithm instead of the BL algorithm provides slightly better RMSE for a given energy budget, despite the fact that the LM scheme is about six times more expensive than LS in CPU cycles. This result is more apparent for the scheme using sub-local objective functions. Additionally, both algorithms showed to be very robust to poor initial estimates, providing significant improvements for the case of multilateration LS which showed an initialization RMSE of 24 meters.

Moreover, we found that DSCL-L gives the best energy-RMSE tradeoff for $\tau < .25m$. It is relevant to note that for low values of τ the energy consumption can be reduced to a third of the largest energy budget ($\tau = 0$), while losing less than one meter of accuracy. Given the measurement errors found in GPS system (around 5 meters under optimal line of sight conditions), we found our localization errors quite competitive (less than 4.5 meters on the average).

Our best, but most expensive, localization error was about 3.25 meters on the average. The obvious question to ask next is if we can achieve this error rate at lower energy consumption. We show in the next two chapters, post-processing or refining schemes that target this objective.

Chapter 6

Improvements of Localization Accuracy using Anchor Convex Hulls

Up to this point we have introduced a localization process consisting of an initialization stage and a refinement stage. The latter stage consists of a variation of multilateration. The algorithm is an iterative collaborative process where all sensors solve a spatially-constrained optimization problem to determine position updates which are broadcasted to continue position refinement over the next iteration. We introduced two types of objective functions: local and sub-local. In the local case, each sensor tries to minimize the mean absolute range error with all its neighbors simultaneously. In the sub-local case, the objective function is actually a set of objective functions where the range error is satisfied between a node an each one of its neighbors separately. A final solution is found by averaging the sub-local solutions.

Our iterative algorithm is characterized by a spatial constraint that limits the solution space to some region around the current position estimate. This constraint allows all the nodes to update their position simultaneously while achieving convergence. In general, experimental results show the local objective function performs better than the sub-local case. We note that the proposed approach has computational characteristics that allow its deployment on real mote hardware. Furthermore, in Chapter 5, we introduced a simple stopping criterion based on a threshold τ that relaxes the absolute distance change between position updates. Each node stops updating its position once τ is exceeded. We performed an extensive evaluation of the tradeoff between localization accuracy and energy consumption. This is a key metric for an iterative scheme as we want to minimize the number of wireless transmissions/receptions (the most energy expensive operations on a WSN) while providing the best localization accuracy. Obviously these are con-

flicting goals; as τ increases, the cost of wireless communication decreases while the localization RMSE increases. We found that there is a strong dependence on the initialization scheme and that the local objective function provides a better accuracy-energy tradeoff.

As discussed before, in WSN deployments, anchors are scattered at lower densities than regular nodes. For instance, in [25, 62] a .5% to 2% anchor proportion is assumed. In our previous experiments we have assumed a 24 to 1 relationship over a 100×100 area. Based on the later premise, we will typically have many sensor nodes lying inside of the contour formed by at least three non-collinear anchors. We define each one of these contours as anchor convex hull (ACH). A relevant behavior between an ACH and the sensor nodes observed over in our simulations was that those nodes located inside ACH tend to be well localized over a shorter number of iterations when compared to those nodes located outside the ACH. A similar observation has been reported in [25, 7, 33].

Hence, we could visualize an additional step to our localization scheme where each sensor determine its membership to an ACH and performs the localization process using the corresponding anchors and ACH member nodes. This step should hypothetically lead to an improvement of localization accuracy (i.e., RMSE) over a large scale network that forms a super-grid of ACHs. In addition, it will be of interest if we can develop additional refinement steps based on the use of an ACH to select which nodes can be further refined (i.e., we remove those nodes who bias the localization towards higher RMSEs).

In this chapter we explore strategies that can be used to increase the node position accuracy based on ACH membership. The drawback of this strategy is that any sensor needs to determine the ACH to decide if it belongs or not to the ACH. However, a naive approximation to the ACH can be used for any sensor in order to reduce processing. We have divided this chapter in four main sections. Section 6.1 validates through extensive simulations our assessment on the significance of the ACH on localization accuracy. In Section 6.2, we present an ACH refinement method that can improve sensor location at the expense of higher energy requirements. Finally in Section 6.3 we present a non-iterative refinement stage that shows significant improvement on accuracy and energy performance by solving an unconstrained optimization problem with the

LM algorithm.

6.1 Impact of Anchor Convex Hull on Localization

In this section we show through extensive simulations how our iterative algorithms provide better position estimates for *inner* nodes than *outer*. Without loss of generality, we use the BL and DSCL-SL algorithms in our evaluation; we found similar behavior for other combinations of algorithms. In our test WSNs, described in Section 4.5.1, each network is composed by four anchors where the RSS ranging technique is used to estimate distances between pairs of neighboring nodes. In our simulations, we use propagation parameters $\eta_p = 2.6, R = 30m$ and $\sigma_{dB} = 6$, and equation (4.7) is used by each node to bound noisy estimate distances from neighboring nodes.

6.1.1 Effects of Centered ACHs in the Accuracy Performance

This subsection is focused on testing localization accuracy for *inner* nodes based on the position of the nodes with respect to the ACH. We are using the maximum number of iterations (i.e., equal to 100) as the only stopping criterion in our simulations. We are using strategically square-centered ACHs, as shown Figure 6.1, for testing each one of the 20 networks.

Each ACH is evaluated on each one of the 20 WSNs. The BL algorithm is used as the first stage to provide initial estimates followed by our DSCL-SL algorithm which is run for 100 iterations. On each network we have two groups: the *inner* and *outer* nodes from which we obtain the localization RMSE separately. We obtain the RMSE for each group of nodes, so 20 RMSE values are obtained per group in each ACH set. We plot the mean RMSE of each group in Figure 6.2 at different ACH sizes. Figure 6.2 shows that *outer* nodes present higher RMSE on the estimated positions than *inner* nodes for both the BL and DSCL-SL algorithms. Also, we plot the overall (*inner* and *outer* nodes) mean RMSE as a reference to the results shown in Figure 4.12(b).

Table 6.1 presents more detailed information in the sense that we can see how the position and size of the ACH affects the initial position estimates. Here, Group 1 corresponds to the *inner*

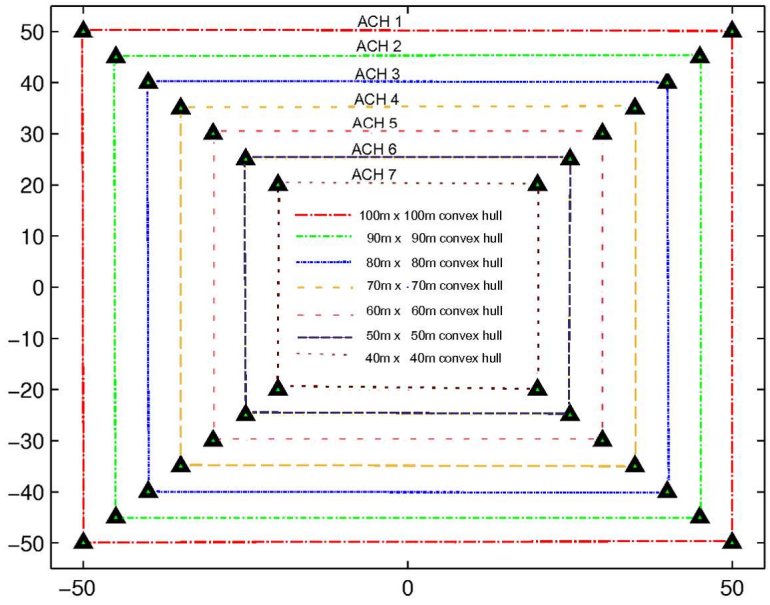


Figure 6.1: Different Convex Hulls formed by four anchors located at the corners.

nodes and Group 2 represents the *outer* nodes. The results confirm that better initial estimated positions are obtained for *inner* nodes whose differences with respect to *outer* nodes are more notorious as the size of the ACH decreases.

Similarly, mean RMSE values for the DSCL-SL approach are summarized in Table 6.2. Here, the DSCL-SL algorithm shows a good improvement on the final position estimates (at least 3:1 with respect to the initial position estimates) for any ACH set. Furthermore, in this table we can observe that much better position estimates are obtained for *inner* nodes than *outer* nodes as shown the relation between Group 1 and Group 2, respectively.

6.1.2 Effects of Non-Centered ACHs in the Accuracy Performance

In last subsection 20 different WSNs were tested using seven centered ACHs. For each test the ACH was centered and gradually reduced in steps of $10mx10m$ starting with an area of $100mx100m$ (ACH1) and finishing with an area of $40mx40m$ (ACH7) as shown Figure 6.1. Since

Table 6.1: Estimated positions based on decreasing and centering a convex hull set inside of WSNs (Bilateration approach).

	Nodes Average		Overall RMSE				RMSE (Group 1)				RMSE (Group 2)			
ACH Size	Group 1	Group 2	min	mean	max	SD	min	mean	max	SD	min	mean	max	SD
ACH 1 (100mx100m)	100	0	13.77	15.13	16.41	0.77	13.77	15.13	16.41	0.77	-	-	-	-
ACH 2 (90mx90m)	82.05	17.95	12.77	14.07	15.49	0.69	12.36	13.86	15.24	0.77	13.06	15.01	17.72	1.26
ACH 3 (80mx80m)	66	34	11.83	12.94	14.90	0.78	11.45	12.34	13.92	0.71	11.47	13.93	19.68	1.85
ACH 4 (70mx70m)	50.6	49.4	10.38	12.23	13.45	0.78	8.36	10.68	12.69	1.09	11.31	13.58	15.15	0.98
ACH 5 (60mx60m)	36.85	63.15	9.89	11.42	13.23	0.93	7.12	8.98	10.66	0.83	10.88	12.60	14.41	1.07
ACH 6 (50mx50m)	25.1	74.9	10.55	11.36	12.59	0.56	6.44	7.86	10.50	1.09	11.13	12.31	13.52	0.65
ACH 7 (40mx40m)	16.3	83.7	10.89	12.12	15.39	1.18	4.48	5.98	7.12	0.63	11.41	12.98	16.50	1.35

Table 6.2: Estimated positions based on decreasing and centering a convex hull set inside of WSNs (DSCL-SL approach).

	Nodes Average		Overall RMSE				RMSE (Group 1)				RMSE (Group 2)			
ACH Size	Group 1	Group 2	min	mean	max	SD	min	mean	max	SD	min	mean	max	SD
ACH 1 (100mx100m)	100	0	3.46	4.44	5.43	0.64	3.46	4.44	5.43	0.64	-	-	-	-
ACH 2 (90mx90m)	82.05	17.95	3.55	4.45	5.33	0.55	2.78	3.84	5.09	0.56	3.57	6.43	8.82	1.36
ACH 3 (80mx80m)	66	34	3.36	4.26	6.31	0.80	2.42	3.03	3.75	0.33	3.92	5.95	9.85	1.59
ACH 4 (70mx70m)	50.6	49.4	3.48	4.72	8.05	0.96	2.35	2.91	3.99	0.45	3.86	6.02	10.23	1.39
ACH 5 (60mx60m)	36.85	63.15	3.37	4.73	6.71	0.92	1.82	2.61	4.15	0.53	4.14	5.61	7.94	1.12
ACH 6 (50mx50m)	25.1	74.9	3.95	4.60	5.52	0.41	1.66	2.29	3.07	0.44	4.17	5.14	6.26	0.49
ACH 7 (40mx40m)	16.3	83.7	3.81	4.88	6.43	0.77	1.44	1.87	2.74	0.34	4.00	5.27	7.25	0.89

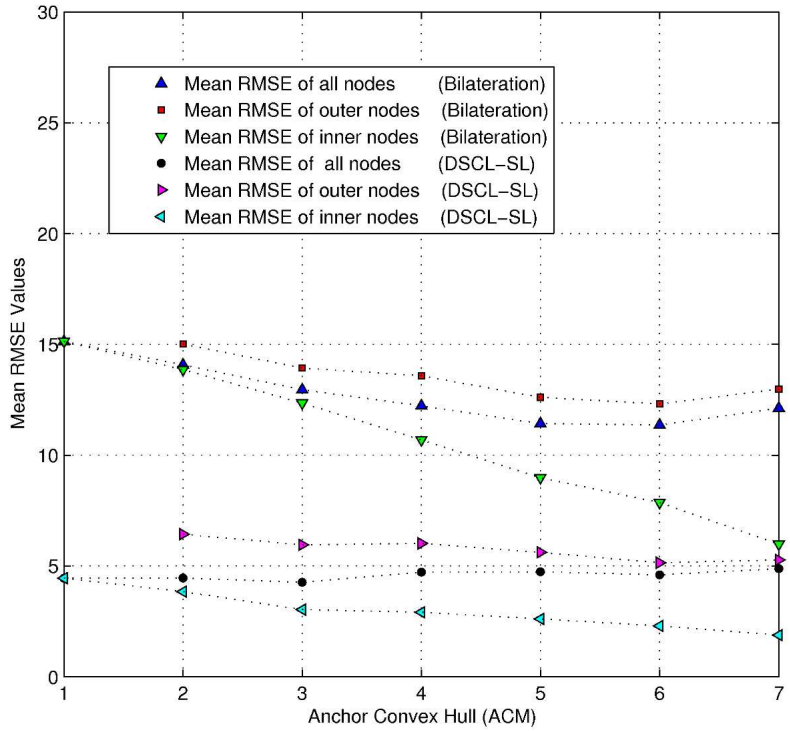


Figure 6.2: mean RMSE values for nodes inside and outside of different convex hull sets.

nodes that have neighbor sensors in all directions commonly tend to be better estimated than nodes that are located at network boundaries, using centered ACHs may introduce a bias that leads us to an erroneous assessment. To assess the impact on having a centered ACH, we repeated the tests of Subsection 6.1.1 using a new set of ACHs as shown Figure 6.3. Here, each ACH with anchor positions $(\mathbf{x}, \mathbf{y}, \mathbf{z}, and \mathbf{w})$ is tested over same 20 WSNs. The key point for this analysis is based on the idea of gradually reducing and moving the ACH to a corner which biases the neighboring anchors to one side of the 100×100 region.

The mean RMSE values are plotted in Figure 6.4. We observe again for all cases that *inner* nodes present better position estimates than *outer* nodes. However, we can also observe that estimated positions for this analysis in general present higher errors in comparison with the centered ACHs implying that ACH location directly affects the localization process. Nonetheless, our observation about the ACH is confirmed.

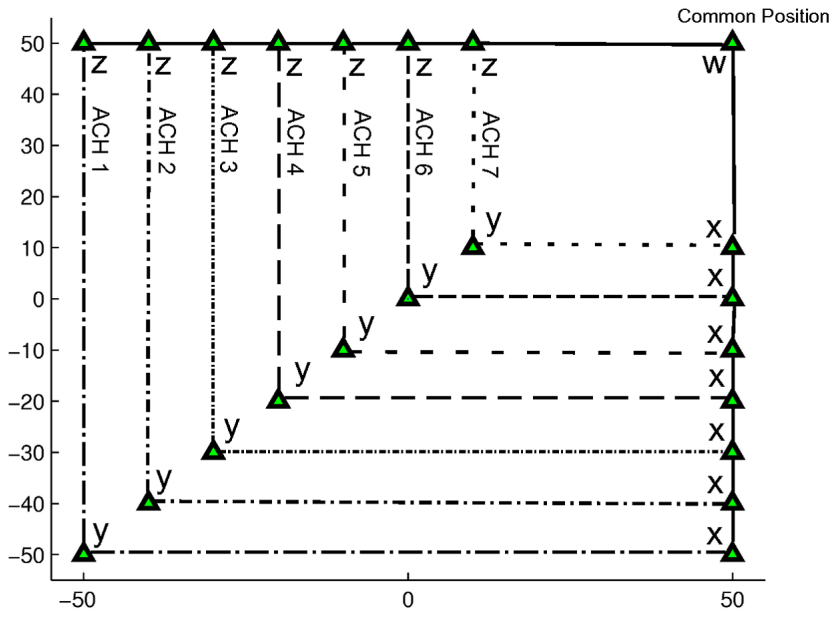


Figure 6.3: Different Convex Hulls formed by four anchors forming a bounding box.

In summary, *inner* nodes have much better estimated positions than *outer* nodes. We belive that using a process of two stages can increase the accuracy performance of the entire network. For example, after the initialization step, if we re-estimate only *inner* nodes with an iterative approach, we could expect an increment on accuracy position estimates in *inner* nodes. Later in a second stage, an overall re-estimating process, i.e., using the same iterative algorithm should promote that *outer* nodes with neighboring *inner* nodes improve their position estimates, so we believe that repeating the two-stage process until satisfies certain stopping criterion should gradually improve the overall RMSE of the entire network. The next section explores this idea.

Table 6.3 and Table 6.4 present a detailed information for both localization algorithms at different ACH sets. Clearly the RMSE values show that the best estimated positions (in both algorithms) are closely related to the number of nodes that lie inside of an ACH, and also error differences between both groups (*inner* and *outer* nodes) are more remarkable.

Table 6.3: Estimated positions based on decreasing and not centering a convex hull set in WSNs (Bilateration approach).

	Nodes Average		Overall RMSE				RMSE (Group 1)				RMSE (Group 2)			
ACH Size	Group 1	Group 2	min	mean	max	SD	min	mean	max	SD	min	mean	max	SD
ACH 1 (100*m*x100*m*)	100	0	13.62	15.1	16.35	0.67	13.62	15.1	16.35	0.67	-	-	-	-
ACH 2 (90*m*x90*m*)	82.05	17.95	12.78	14.18	16.18	0.83	11.66	13.82	15.69	1.01	11.12	15.54	18.94	2.07
ACH 3 (80*m*x80*m*)	66	34	12.45	13.34	14.62	0.74	11.58	12.86	14.73	0.87	12.12	14.54	16.88	1.31
ACH 4 (70*m*x70*m*)	50.6	49.4	11.52	13.42	14.63	0.84	8.99	11.20	13.85	1.10	12.08	15.33	19.04	1.58
ACH 5 (60*m*x60*m*)	36.85	63.15	11.94	14.22	17.67	1.36	7.97	10.97	14.20	1.26	12.92	15.79	19.87	1.61
ACH 6 (50*m*x50*m*)	25.1	74.9	14.62	17.38	22.22	1.87	8.19	11.09	15.24	1.90	15.52	18.97	23.91	2.11
ACH 7 (40*m*x40*m*)	16.3	83.7	18.74	22.43	27.71	2.57	9.84	13.22	19.46	2.5	19.73	23.80	30.27	2.90

Table 6.4: Estimated positions based on decreasing and not centering a convex hull set in WSNs (Sub-Local approach).

	Nodes Average		Overall RMSE				RMSE (Group 1)				RMSE (Group 2)			
ACH Size	Group 1	Group 2	min	mean	max	SD	min	mean	max	SD	min	mean	max	SD
ACH 1 (100*m*x100*m*)	100	0	3.83	4.64	5.21	0.36	3.83	4.64	5.21	0.36	-	-	-	-
ACH 2 (90*m*x90*m*)	82.05	17.95	3.55	4.72	6.41	0.71	3.05	3.90	5.22	0.54	3.74	7.28	11.25	1.99
ACH 3 (80*m*x80*m*)	66	34	3.71	4.82	6.24	0.68	2.75	3.60	4.81	0.53	4.67	6.57	9.5	1.27
ACH 4 (70*m*x70*m*)	50.6	49.4	3.67	5.29	6.59	0.83	2.57	3.47	4.83	0.57	4.27	6.62	8.62	1.32
ACH 5 (60*m*x60*m*)	36.85	63.15	4.13	5.56	7.99	1	2.5	3.65	4.6	0.72	4.70	6.43	9.78	1.39
ACH 6 (50*m*x50*m*)	25.1	74.9	5.57	7.45	11.16	1.58	2.36	4.04	6.18	0.93	6.07	8.25	12.29	1.79
ACH 7 (40*m*x40*m*)	16.3	83.7	5.66	9.74	13.84	2.09	3.68	5.53	6.98	0.98	5.99	10.37	15.10	2.3

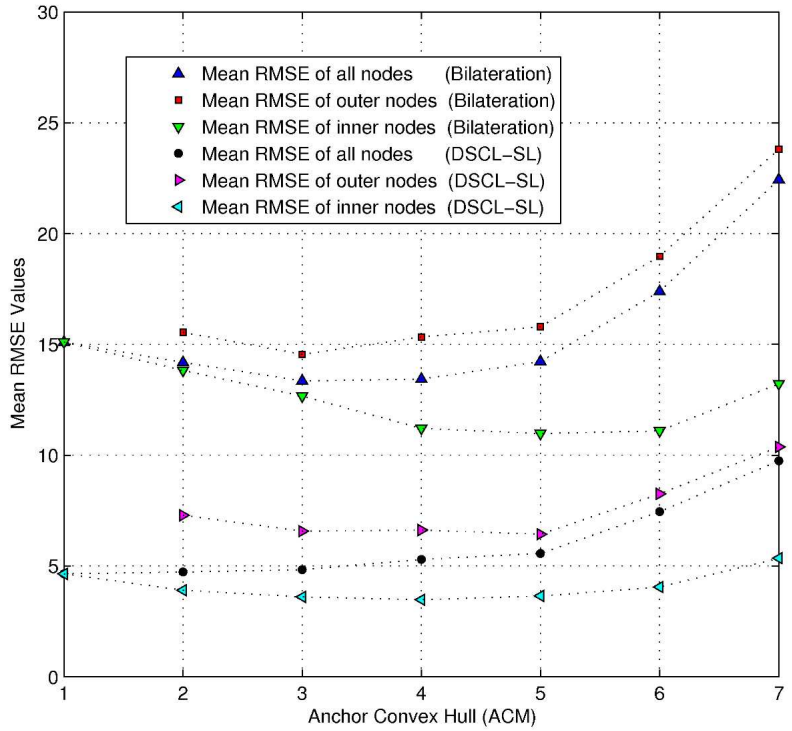


Figure 6.4: Mean RMSE values for nodes inside and outside of different convex hull sets.

6.2 Multistage Localization Based on ACH Membership

6.2.1 Approximation to the ACH

Computation of the convex hull is a fundamental operation in computational geometry. For example, it can serve as a first preprocessing stage for geometric algorithms. Also, it is important in optimization problems. If the convex hull is found or provided, the problem can be solved with linear programming relaxation using interior point methods. The convex hull concept also has been employed in practical applications like pattern recognition, image processing, statistics, data mining, and recently in WSNs [22].

Although there are complex algorithms that can be used by a sensor node, s_i, to determine the exact form of the ACH [53], the number of required computations are extremely expensive

for constrained systems (i.e., nodes) so in order to reduce processing time in a sensor node we have decided to use a simple approximation to the convex hull. For example, once we obtain initial estimates, computed by any initialization algorithm described in Chapter 2, each sensor node s_i with position $\mathbf{p}_i = [p_x, p_y]^T$ creates a square box using the coordinates of anchor positions $\mathbf{q}_k = [x_k, y_k]^T$ with $k = 1, 2, ..., M$, by finding a bounding box $(x_{min}, y_{min}), (x_{max}, y_{max})$ as

$$x_{min} = \min(x_1, x_2, x_3, x_4) \tag{6.1a}$$

$$x_{max} = \max(x_1, x_2, x_3, x_4) \tag{6.1b}$$

$$y_{min} = \min(y_1, y_2, y_3, y_4) \tag{6.1c}$$

$$y_{max} = \max(y_1, y_2, y_3, y_4) \tag{6.1d}$$

This is a rough approximation of the trapezoid formed by the anchors into a rectangular box. This is a simple scheme that is easily implementable by each node. We will show that this scheme provides an effective way for each node to self classify as being inside or outside the ACH. A sensor s_i decides to be within the ACH if the following two equations are satisfied:

$$x_{min} \leq p_x \leq x_{max} \tag{6.2a}$$

$$y_{min} \leq p_y \leq y_{max}. \tag{6.2b}$$

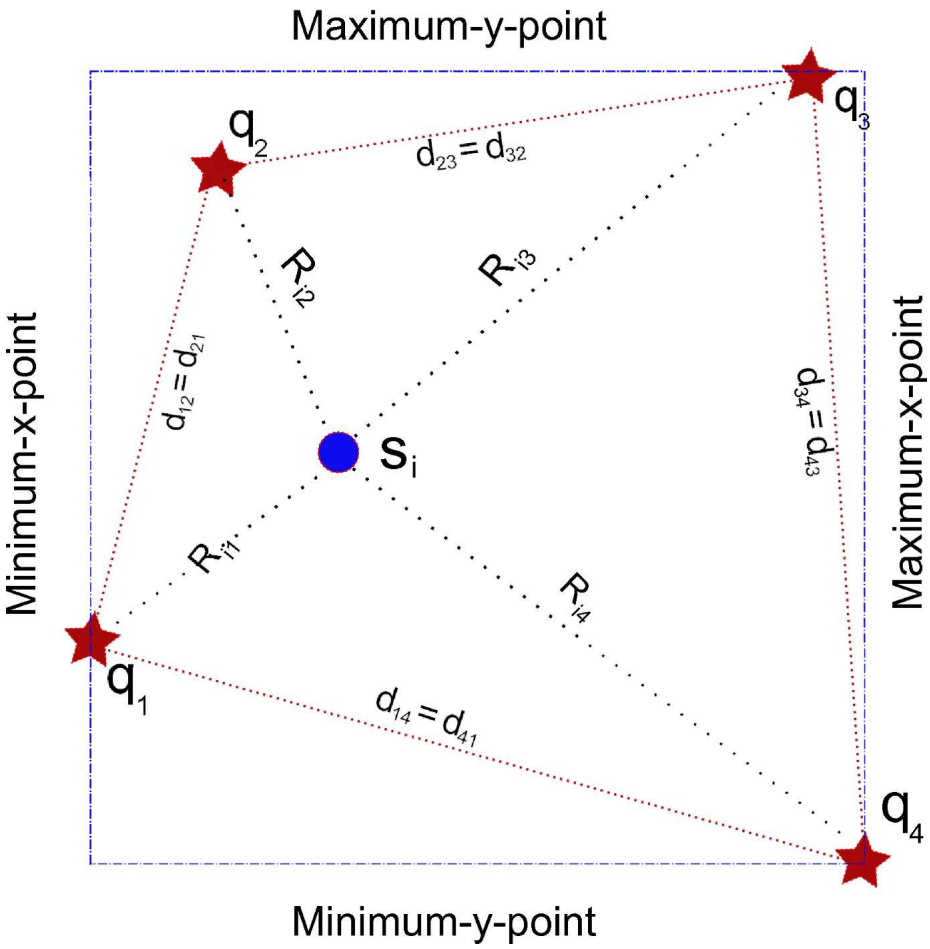


Figure 6.5: Quadrant Restriction formed by four anchors.

6.2.2 Two-Stage Localization

In this section, we describe a methodology for improving the accuracy on the position estimates. We assume that all nodes have already found their initial position estimates using any algorithm of initialization. Our method is based on two stages:

- In the first stage, only *inner* nodes participate in the refining process using an iterative algorithm. Nodes decide to re-estimate their positions based on their initial position estimates which should satisfy equation (6.1). Nodes that do not participate in the first stage (*outer* nodes) will be ignored by active nodes (*inner* nodes) during their re-estimation stage, so an active node s_i consider only neighboring *inner* nodes that are inside of its radio range R to re-estimate its position.

- Once *inner* nodes have refined their positions, an overall refining process (i.e., *inner* and *outer* nodes) starts at as second-stage using the same iterative approach.

Consider the 20 test WSNs described in Subsection 4.5.1 where connectivity and noisy distances among sensors remain without change, using the RSS ranging technique with $\eta_p = 2.6, R = 30m$, and $\sigma_{dB} = 6$ as typical parameters. We use the scheme already mentioned in last sections where for each centered ACH (See Figure 6.1) we apply the bilateration followed by the two stages approach using the DSCL-SL approach as iterative algorithm. We should take into account that the priority of this methodology is the accuracy in the position estimates rather than the energy consumption in sensor nodes. For each ACH, we run the bilatreation followed by the two-stage approach which is run for 100 iterations per stage. We obtain 20 RMSE values for both groups of nodes: *inner* and *outer* nodes. Table 6.5 summarizes the mean RMSE values obtained for each ACH set.

Comparing the mean RMSE values obtained by the DSCL-SL approach shown in Table 6.2 with the RMSE values obtained by the two-stage approach presented in Table 6.5, we observe that better position estimates are obtained using the two-stage approach using the same scenarios and same initial conditions. Furthermore, when the neighboring density is 4:1 in favor of *inner* nodes (ACH 2 case), the differences on the accuracy performance are more remarkable.

Table 6.5: Estimated positions based on the two-stage approach.

	Nodes Average		Overall RMSE				RMSE (Group 1)				RMSE (Group 2)			
ACH Size	Group 1	Group 2	min	mean	max	SD	min	mean	max	SD	min	mean	max	SD
ACH 1 (100mx100m)	100	0	2.63	4.06	6.14	0.89	2.63	4.06	6.14	0.89	-	-	-	-
ACH 2 (90mx90m)	82.05	17.95	2.85	3.62	4.59	0.51	2.71	3.11	4.20	0.36	2.86	5.17	8.26	1.48
ACH 3 (80mx80m)	66	34	2.48	3.88	5.99	0.75	1.92	2.91	3.67	0.47	2.96	5.21	8.27	1.28
ACH 4 (70mx70m)	50.6	49.4	2.85	4.26	5.90	0.88	1.82	2.56	4.09	0.55	3.41	5.46	8.21	1.32
ACH 5 (60mx60m)	36.85	63.15	3.00	4.24	5.40	0.76	1.63	2.24	2.94	0.35	3.50	5.07	6.59	1.04
ACH 6 (50mx50m)	25.1	74.9	3.11	4.42	5.88	0.81	1.50	2.02	3.08	0.37	3.44	4.97	6.66	0.92
ACH 7 (40mx40m)	16.3	83.7	3.13	4.78	6.57	1.04	1.46	1.85	2.35	0.26	3.32	5.16	7.13	1.17

Figure 6.6 shows a typical behavior for one network, the continuous line represents the RMSE values obtained by the DSCL-SL approach at 100 iterations; it starts at 14.55m of initial error and finishes at 3.29m. On the other hand, dotted lines represent the RMSE values provided by the the two-stage algorithm. For this approach, in first-stage only *inner* nodes participate in the refining process starting at 14.55m and finishing at 9.97m of error using 100 iterations. In the second-stage, all nodes participate in the process and clearly the estimated positions are improved until reach a RMSE of 2.64m. Clearly, the localization process based on two stages improves the accuracy in the estimated positions at 0.65m for this specific case. However, it also requires too many extra iterations which imply more energy consumption in the nodes (wireless transmissions). This is a drawback of this approach, and also it is opposite to our goals. In the next section we test this approach under more realistic scenarios.

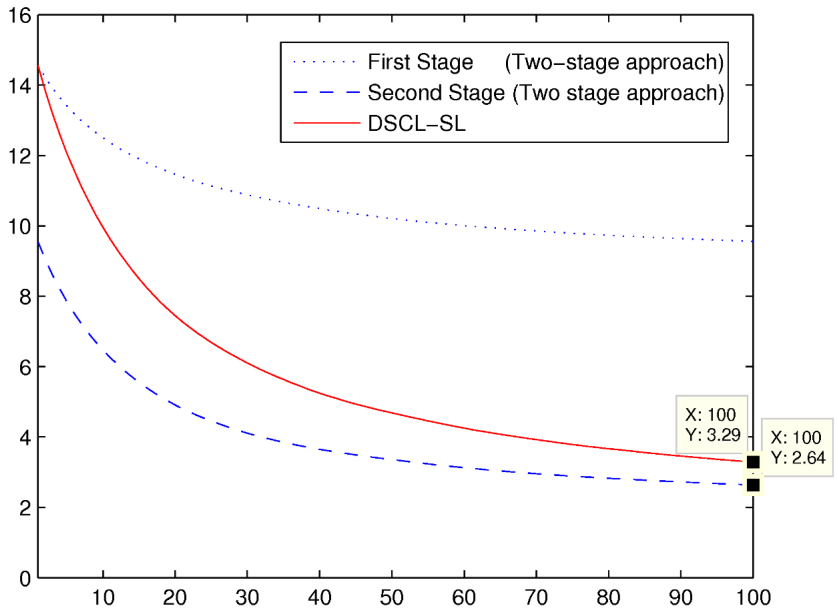


Figure 6.6: RMSE mean values for nodes inside and outside of different convex hull sets.

6.2.3 Localization for Non-Uniform ACHs

In this section, we use the 20 WSNs presented in Section 4.5.1 where the convex hull of anchors are forming polygons instead of ideal square boxes as described in last subsection. Each sensor s_i using ranging and position information of anchors should be able to estimate its own initial position using the bilateration algorithm, which gives the ability to each sensor s_i to decide if it belongs or not to the ACH using equation (6.1) as described before.

Let us define a refinement to the process of running the two-stage approach. Now considering that each node can re-estimate if it still belongs or not to an ACH after finishing one refinement process, each node s_i can request a new refinement process if it has changed its state from the group of *inner* to *outer* nodes or vice versa. In this way, we will show through simulations that position estimates gradually decrease and converge after few refinements, and also *inner* and *outer* nodes tend to not change their states after few refinements.

Table 6.6 summarizes the errors on the estimated positions after five refinements, for each one of the 20 networks, using the two stage approach. Clearly, we can see that in each refinement process the RMSE values becomes closer and closer implying that all sensors will no longer change in future refinements. Also, this table shows that the two stage procedure overcomes the estimated positions provided by the DSCL-SL approach (third column) for all cases. For example, after five refinements, the two stage approach improves the estimated positions, in average, by $1.38m$ with respect to the DSCL-SL approach. However, the two-stage approach greatly increases the number of iterations or wireless transmissions, making this approach not suitable to be implemented in constrained resources like nodes as was mentioned before.

From the RMSE results shown in Table 6.6, we can conclude that it is necessary to develop a better strategy that can be used to reduce errors in the estimated distances without spending many wireless transmissions. The next section describes a much better approach that can be used not only to improve position estimates but also to save energy in the nodes. This methodology takes the idea of the post-processing stage described in the last section.

Table 6.6: Estimated positions after a few refinements.

	Bilateration RMSE (m)	DSCL-SL One-Run RMSE (m)	Two-Stage Approach									
			Refinement 1 RMSE (m)		Refinement 2 RMSE (m)		Refinement 3 RMSE (m)		Refinement 4 RMSE (m)		Refinement 5 RMSE (m)	
Network	Initial Estimates	Final Estimates	Initial	Final	Initial	Final	Initial	Final	Initial	Final	Initial	Final
1	14.30	4.97	14.3	4.55	4.55	4.03	4.03	3.91	3.91	3.88	3.88	**3.94**
2	12.32	5.14	12.32	4.58	4.58	3.83	3.83	**3.71**	3.71	3.68	3.68	3.67
3	14.34	5.24	14.34	4.82	4.82	3.78	3.78	3.38	3.38	**3.20**	3.20	3.11
4	14.33	5.82	14.33	5.70	5.70	4.78	4.78	4.31	4.31	**3.86**	3.86	3.58
5	12.51	4.52	12.51	3.98	3.98	3.34	3.34	3.20	3.20	**3.14**	3.14	3.10
6	12.66	5.14	12.66	5.30	5.30	4.11	4.11	3.83	3.83	**3.73**	3.73	3.68
7	13.25	5.73	13.25	4.91	4.91	4.04	4.04	3.81	3.81	**3.76**	3.76	3.75
8	13.90	5.97	13.90	5.66	5.66	5.28	5.28	5.13	5.13	4.94	4.94	4.91
9	11.72	5.72	11.72	5.75	5.75	5.09	5.09	**4.68**	4.68	4.65	4.65	4.63
10	13.87	5.52	13.87	4.80	4.80	4.07	4.07	3.57	3.57	**3.22**	3.22	3.03
11	13.19	4.71	13.19	4.39	4.39	3.45	3.45	**3.23**	3.23	3.18	3.18	3.16
12	11.96	3.70	11.96	3.14	3.14	3.02	3.02	**3.03**	3.03	3.05	3.05	3.06
13	12.09	3.63	12.09	3.36	3.36	2.54	2.54	2.43	2.43	**2.39**	2.39	2.38
14	12.80	4.20	12.80	3.83	3.83	3.42	3.42	**3.34**	3.34	3.33	3.33	3.33
15	12.62	4.63	12.62	4.06	4.06	3.71	3.71	3.63	3.63	**3.64**	3.64	3.65
16	12.73	4.49	12.73	3.77	3.77	2.78	2.78	**2.57**	2.57	2.49	2.49	2.46
17	12.46	4.57	12.46	4.10	4.10	3.80	3.80	**3.65**	3.65	3.57	3.57	3.56
18	12.43	4.76	12.43	4.11	4.11	3.50	3.50	3.45	3.45	3.56	3.56	**3.58**
19	13.73	6.09	13.73	5.45	5.45	5.20	5.20	5.06	5.06	5.04	5.04	**5.03**
20	11.99	4.36	11.99	3.85	3.85	3.47	3.47	3.58	3.58	3.68	3.68	**3.72**

6.3 A Post-Processing Stage to Improve Localization

On the last section, we presented a technique to improve localization accuracy albeit at a very large energy and computational cost. In this section we develop a novel refinement method that requires minimal additional resources while achieving significant improvements. We base our method on two of our results from the previous sections and chapters. First, we have just shown that we can further refine positions through the application of DSCL over two stages; this indicates that the use of a refinement stage is a valid scheme. Second, in Chapter 2 we presented algorithms for initialization using multilateration, including a novel bilateration algorithm. These algorithms solve an unconstrained problem using the location and range measurements of anchors. In particular, we found in Chapter 5 that the combination of LM/DSCL-L algorithm has provided the best accuracy-to-energy tradeoff. Furthermore, it is well known that the LM algorithm is a unconstrained method with q-quadratic convergence given that a *good initial point is provided* [48]. Looking at the general multilateration problem, each node will solve the program

$$\min_{\mathbf{p}_i} f(\mathbf{p}_i) = \min_{\mathbf{p}_i} \left(\frac{1}{2} \mathbf{R}(\mathbf{p_i})^T \mathbf{R}(\mathbf{p_i}) \right) \tag{6.3}$$

where $\mathbf{R}(\mathbf{p}_i)$ is the residual vector described in equation (2.17), and a solution could be provided with the LM algorithm as described by Algorithm 1.

From these two observations, we provide the following scheme:

- Solve the localization problem using any of the combinations described in Chapter 5. In particular we will use the LM/DSCL-L algorithm for the rest of this section.
- Identify the ACH *inner* nodes.
- For each node in the ACH, set a multilateration minimization problem with its ACH neighbors.
- Solve this problem with the LM algorithm using the positions obtained with LM/DSCL-L as an *initial point*.

- The solution becomes the new position estimate for the node.

To evaluate this scheme, we use as a reference the LM/DSCL-L results from Section 5.1. Let us consider a scenario where a WSN is deployed over a $100m \times 100m$ area that contains 4 anchors with full connectivity, and the remaining sensor nodes have a radio range of 30m. Additionally, noisy ranges among sensors are modeled according to the RSS scheme with $\eta = 2.6$ and $\sigma_{SH} = 6_{dB}$. We replicate the localization results on Figure 6.7 where we show the average RMSE, over 20 networks, as a function of τ, including the overall localization error, and the errors for the *inner* and *outer* nodes. As expected, the *inner* nodes have an average lower localization error of about two meters with respect to the *outer* nodes. Also, for reference, we show the number of transmissions as a function on τ in Figure 6.8. We recall that for our simulations, $\tau = 0$ means that the DSC-L algorithm runs for 100 iterations implying 9600 transmissions (e.g., 96 nodes). Also, in the following discussion, we do not account for wireless receptions; these have been discussed in detail on Chapter 5.

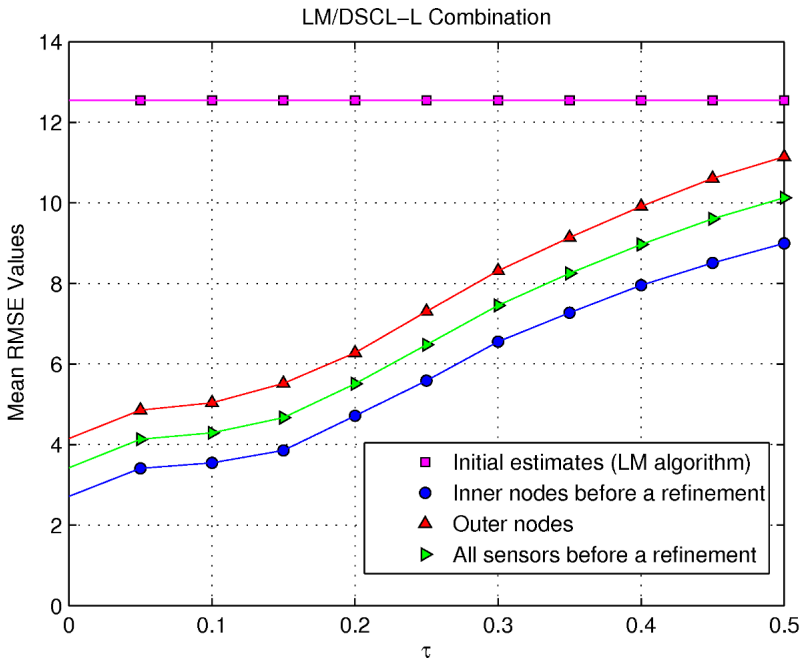


Figure 6.7: Mean RMSE values for the overall, *inner*, and *outer* nodes at different stopping criteria considering the LM-based initialization algorithm and the DSCL-L iterative approach.

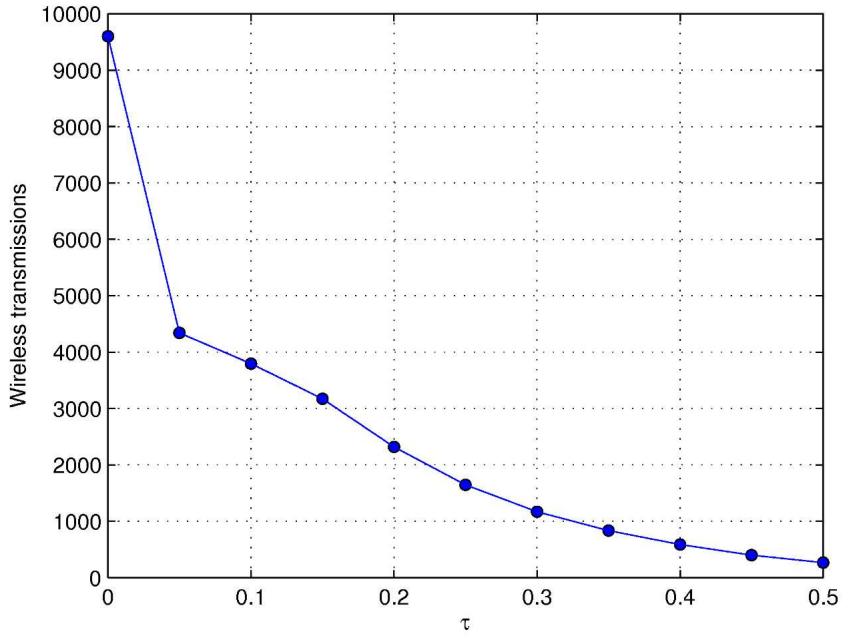


Figure 6.8: Wireless transmission required by the LM/DSCL-L combination at different stopping criteria.

Let us to consider the hypothetical scenario shown in Figure 6.9. Here, the sensor s_i after finishing its iterative process (e.g., DSCL-L) determines if it belongs or not to the convex hull formed by three anchors a_1, a_2, and a_3. Given that each node s_i also has the knowledge of its neighbor positions, it can also determine which of them are inside the ACH. Using this subset of node positions and the anchor positions, s_i sets the residual equation (2.17) which is used to solve problem (6.3) using the LM algorithm to produce a refined position. All *inner* nodes follow the same process simultaneously and produce a refined set of positions. We should keep in mind that on the average, the number of neighboring nodes for any sensor s_i is around $N = 21$ as shown table 4.3 and $M = 4$ anchors. Hence, equation will use $M + N \cong 25$ reference positions which will increase the computational cost for the gradient and Jacobian approximations needed by the LM algorithm (see Chapter 2. Nonetheless, it is very important to remark that this procedure is executed internally by each node without the need of additional transmissions or receptions. Since the LM algorithm is applied internally without the need for further broadcasts, we can

expect substantial energy savings as the energy requirements of the LM algorithm are negligible compared to radio utilization. These energy costs were discussed in detail on Chapter 5.

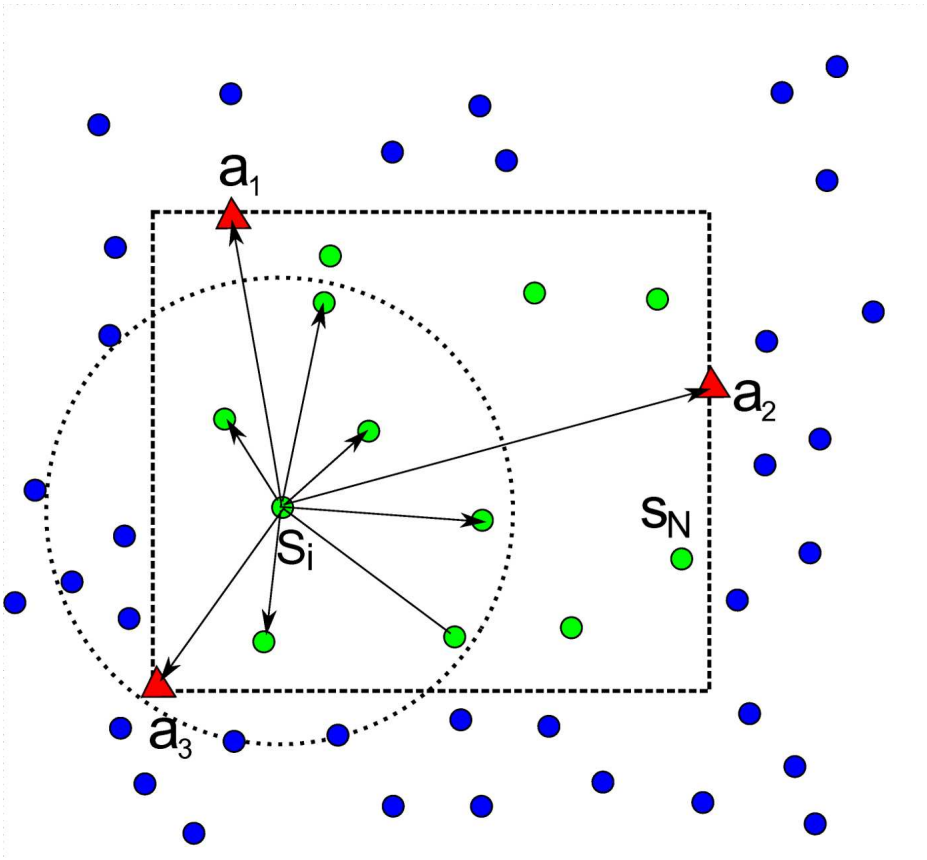


Figure 6.9: An *inner* sensor node s_i refining its own position after the LM/DSCL-L combination.

The results for the refinement stage are plotted in Figure 6.10. The Figure shows how *inner* node position estimates are greatly improved by solving (6.3) with the LM algorithm. This is a significant result for larger values of τ which require a lower number of DSCL-L iterations, and in consequence less energy. As an example, consider $\tau = 0.5m$ where the DSCL-L algorithm gives an *inner* node average RMSE of $8.99m$ which is improved to $5.97m$ after refinement. Similarly, the overall average RMSE goes from $10.1m$ to $8.62m$. This result becomes more relevant by comparing the number of transmissions required to achieve the same localization performance when no refinement is used. From Figure 6.7, we see that a RMSE of $8.62m$ is obtained with $\tau = 0.375$. This performance level requires 445 additional wireless transmissions for the entire

network as can be found in Figure 6.8. 5.

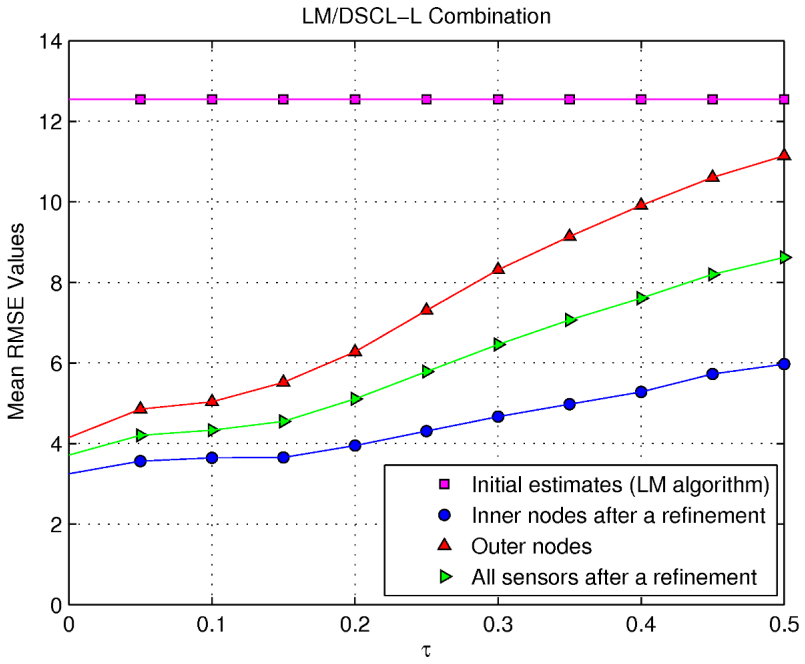


Figure 6.10: RMSE results by applying a post-processing step to only *inner* nodes after the LM/DSCL-L combination.

Although our objective has been the improvement of localization accuracy based on membership to the ACH. There is no reason to limit the LM-based refinement stage to the *inner* nodes. In particular, it is clear from Figures 6.7 and 6.10 that *outer* nodes generate a large bias on the overall RMSE. Hence, we decided to apply the LM post-processing stage to all position estimates (*inner* and *outer*) obtained from the DSCL-L output. In summary, the localization scheme is given by the following steps

- Solve the localization problem using any of the combinations described in Chapter 5. In particular we will use the LM/DSCL-L algorithm for the rest of this section.
- For each node in the network set a multilateration minimization problem using its neighbors.

- Solve this problem with the LM algorithm using the positions obtained with LM/DSCL-L as an *initial point*.
- The solution becomes the new position estimate for the node.

.

The results are shown in Figure 6.11.

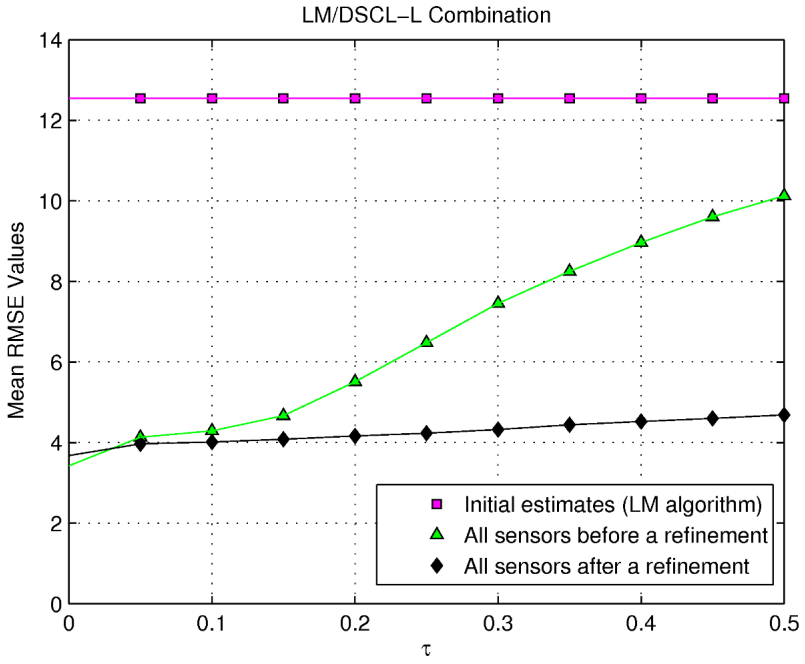


Figure 6.11: RMSE results by applying a post-processing step to all sensors after the LM/DSCL-L combination.

The mean RMSE results show a great improvement on the final position estimates. In this case we can observe that much better results are obtained for the larger values of τ. For example, for τ=0.5 DSCL-L provides an overall mean RMSE of 10.13m, and the proposed scheme reduces this error to 4.68m. In fact, we see that the curve for the refinement step is a straight line with a very small slope. Hence, the increment on RMSE from $\tau = 0$ to $\tau = 0.5$ is about one meter. If we consider the RMSE of 4.68m in Figure 6.7, we see that it is obtained with $\tau = .15$. This value for τ requires 3174 wireless transmissions (plus related wireless receptions) as shown Figure 6.8.

On the other hand, with $\tau = 0.5$, the number of transmissions is less than 500. This improvement comes only from CPU processing which requires much less energy than wireless transmissions. Hence, for this example we have reduced transmissions six-fold while the error is within one meter of the best (but extremely energy expensive) localization performance.

This refinement strategy provides an excellent tradeoff between position accuracy and energy-consumption. We can attribute this improvement to the optimal solutions obtained by the LM algorithm. As described before, the LM algorithm solves a non-linear unconstrained problem based on the Gauss-Newton method. Also, another important characteristic added to the LM algorithm is the use of a descent direction condition (Armijo's condition) which guarantees at least local minima for the position estimates. These characteristics make the LM algorithm convergences almost q-quadratic. This is clearly observed in Figure 6.11 where we achieve a remarkable reduction on RMSE for the larger values of τ.

These results have two main implications:

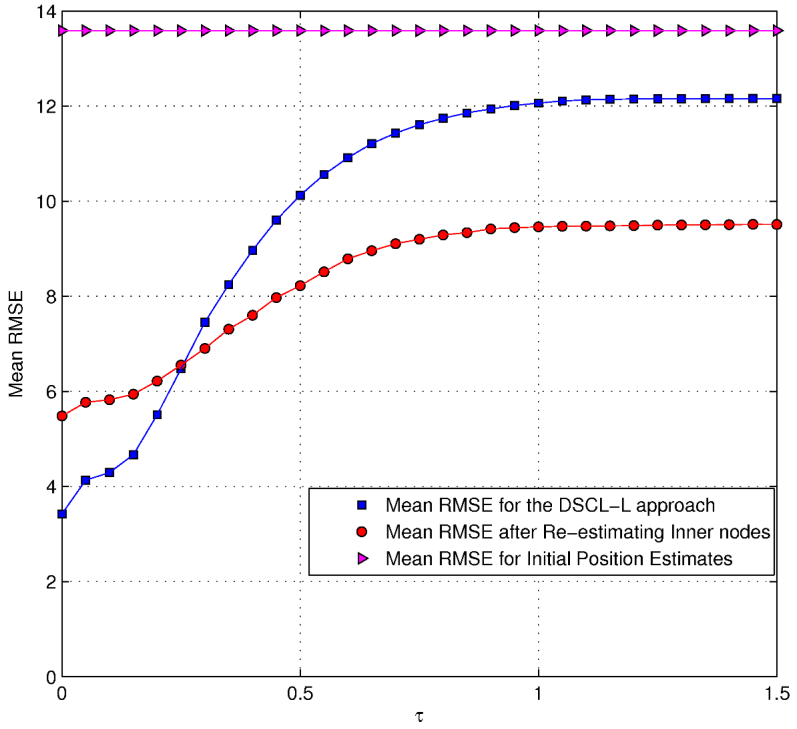
1. We can use a relaxed value for τ which will move the set of position estimates to a "good" initial point within the solution region of the LM algorithm. Then the LM algorithm will find an optimal solution given the set of measured ranges and position estimates. Hence, this strategy greatly increases accuracy without spending energy over additional wireless transmissions.

2. For $\tau \leq .1$ meters, we see that there is no big improvement on RMSE. In fact, for $\tau = 0$, there is an increase on the refinement stage with respect to the DSCL-L result. This suggests that DSCL-L without stopping criteria is finding position estimates which cannot be improved by LM. In fact the LM algorithm is a general solver to an unconstrained minimization problem which does not integrate additional information about the problem. In the case of DSCL-L and DSCL-SL, the spatial constraint implicitly includes information about the localization problem, providing a more meaningful solution in the process (i.e., we know the solution is moving over a 2-D region over a sensor field).

Now, we are ready to use the distributed/iterative LM scheme as a post-processing stage over

the results provided by the LM/DSCL-L combination scheme shown in Figure fig:LM-Local.

First, the idea is to re-estimate *inner* nodes until a stopping criterion be reached. Also, it is important to determine the number of required iterations for *inner* nodes to find their optimal position estimates. Figure 6.12 and 6.13 show the accuracy performance and energy performance (i.e., transmissions) respectively after applied the distributed/iterative LM algorithm.



b

Figure 6.12: The distributed/iterative LM scheme re-estimating *inner* nodes after the LM/DSCL-L combination.

From the figures we observe that for $\tau < 0.25$, the refined *inner* nodes have actually worst performance than the DSCL-L scheme. We present values up to $\tau = 1.5$ where we see that the there is an improvement after refinement. However we note that this improvement is rather modest when compared to the results from 6.11. For example, if we compare the RMSEs for $\tau = 0.5$, we can see that there is a slight improvement from 8.6 meters to approximately 8.2 meters. Another important aspect regarding this iterative approach is that the *total* number of wireless

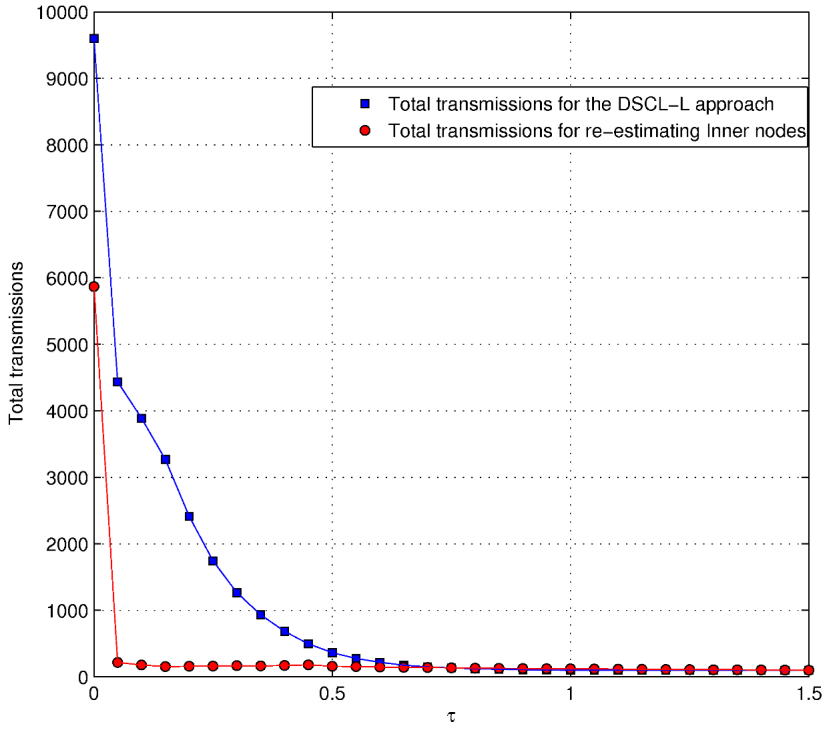


Figure 6.13: Number of required iterations of the distributed/iterative LM scheme to re-estimate *inner* nodes.

transmissions needed by the refinement stage is very low for $\tau > 0$, as shown in Figure 6.13. In other words, position refinements are found with very few runs of the the LM algorithm per sensor node. This is a very important and desirable property that maintains the energy consumption low.

As on our previous analysis, we can perform the refinement stage over all the sensor nodes on the network rather than just on the *inner* nodes. We can see in Figure 6.14 that there is an excellent improvement with respect to just refining *inner* nodes.

From the Figure we can appreciate that final position estimates provided by LM/DSCL-L at a $\tau = 1.5m$ provides a mean RMSE around 12.15 meters with 96 transmissions while the refinement stage improves such position estimates down to a mean RMSE of 3.97 meters with and additional 210 transmissions as shown in Figure 6.15. This implies that each node on the WSN needed between three to four transmissions on the average. These last results satisfy our

original strategy of improving accuracy performance without a substantial increase on wireless transmissions.

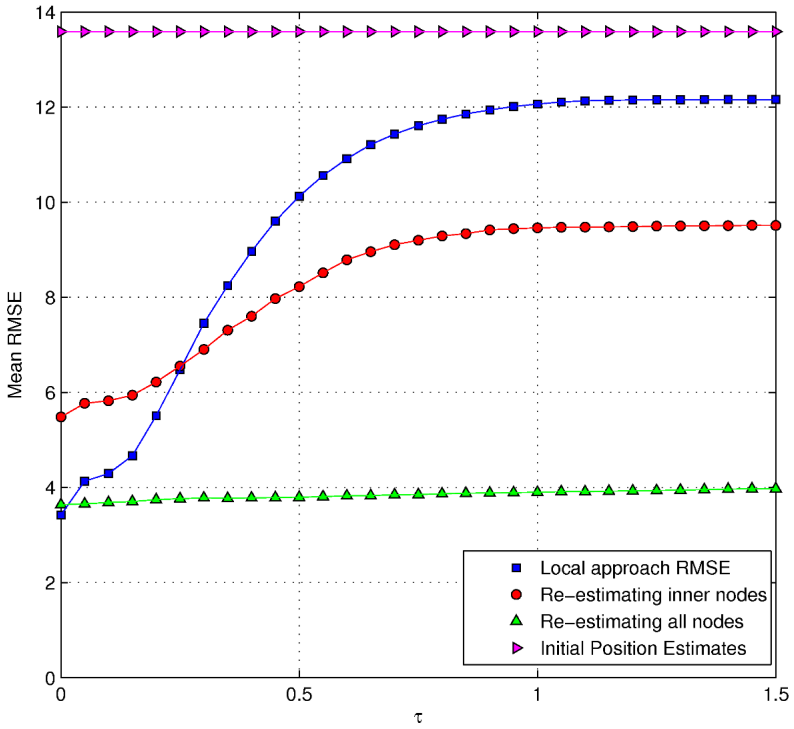


Figure 6.14: The distributed/iterative LM scheme re-estimating all nodes after the LM/DSCL-L combination.

In summary, in this chapter our initial attempt was to exploit the use of the ACH as a way to identify the *inner* nodes which could provide better position estimates and could be localized first. Then, all nodes could be updated using anchor and *inner* node positions. We showed that accuracy can be improved with this scheme albeit at a significant increase in computational and communications cost. Hence, the approach is not feasible for deployment on a real WSN. From this work, we found a novel method where a significant improvement on accuracy is obtained with a minimal increase on complexity and without the need to differentiate between *inner* and *outer* nodes. The proposed method uses the LM/DSCL-L algorithm as before, and adds a refinement stage where each sensor solves a multi-lateration least-squares problem using the LM

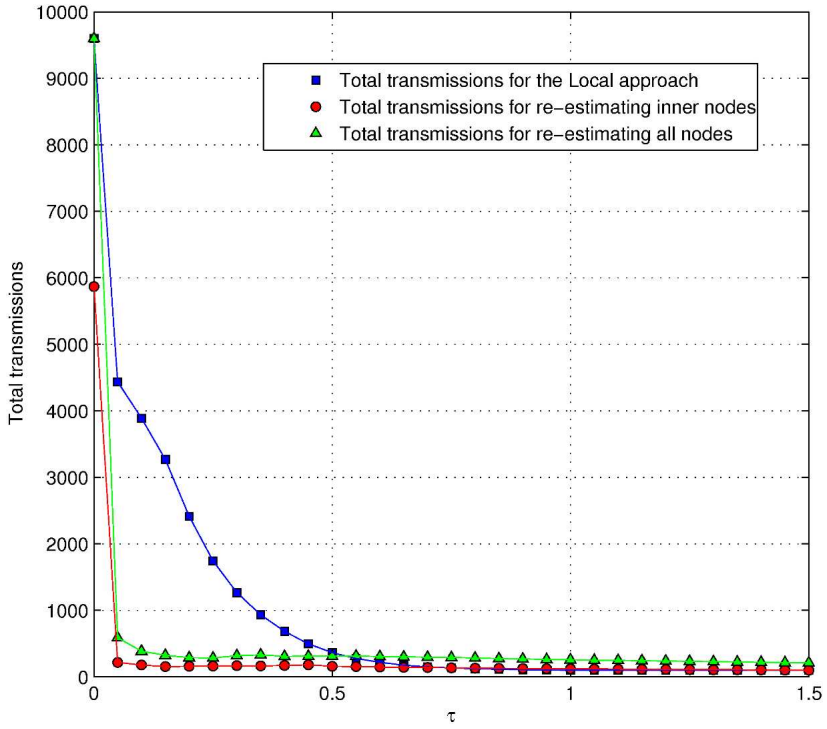


Figure 6.15: Number of required iterations of the distributed/iterative LM scheme to re-estimate all nodes.

algorithm. The convergence property of the LM algorithm allows for excellent localization accuracy (less than four meters on the average) with large stop thresholds (e.g., $\tau = 1.5$ meters). This combination of features provides excellent performance in terms of accuracy and low energy consumption, making the method suitable for real WSN applications.

Chapter 7

Conclusions

This dissertation presents research results on the important issue of node localization in wireless sensor networks (WSNs) . We have divided our conclusions in two main sections. In the first Section 7.1, we present the major contributions of this research. We developed novel localization algorithms that can be realistically deployed over real WSNs, and they can provide good accuracy performance and low energy consumption. The second section discusses future lines of work that use our results as a starting point. Subsection 7.2.1 discusses potential solutions to the case where node to anchors range estimates need to be derived using multi-hop methods (e.g., DV-hop). In Subsection 7.2.2 we discuss the use of realistic irregular radio propagation models to develop localization algorithms that are robust to such phenomena. Our algorithms seem to be well suited to account for radio irregularities. Finally, Subsection 7.2.3 presents a promising iterative/distributed scheme that exploits the power and properties of the Levenberg-Marquardt algorithm. In this subsection, we present preliminary results and potential applications for this scheme.

7.1 Contributions

The major contributions of this research consist of three distributed algorithms for node localization and a model used to characterize the energy consumption of the localization process. These contributions are explained in detail next.

Our first algorithm is a distributed scheme, which can be used as an initialization stage to find an initial set of locations. The other two algorithms are iterative and distributed schemes where each sensor node re-estimates its own location based on updates provided by its neighbors. In

summary the algorithms are described as follows:

a) Most initialization algorithms demand very high computing power to provide a set of initial estimates for an N-node WSN. Our proposed initialization algorithm is based on a bilateration argument. The algorithm is capable to provide competitive initial estimates at low processing power. This approach is basically formed by two stages. The first stage consists of finding all circle intersections formed by anchor positions and their respective range estimates to a sensor node, obtained by ranging techniques like *ToA* or *RSS*. The great advantage of this approach is to use "closed-formulas" to find all circle intersections (i.e., candidate positions) using two anchors at time. In the second stage, the algorithm uses a sorting algorithm to find the cluster of candidate positions that tend to be closer to each other around the true location. The cluster with the nearby candidate positions is averaged to finally obtain the initial location. This scheme can be used by any WSN localization algorithm that needs initial approximations. Also, it can be implementable in constrained devices with low processing and memory capabilities (i.e., motes). Experimental results show that this initialization algorithm is well behaved in comparison with other well known algorithms like LS methodologies.

b) The refinement algorithms enable each sensor node to dynamically or iteratively re-estimate its own position until some stop criteria is met. This stage is based on the idea of splitting a complex global objective function into what we identify as "local" and "sub-local" optimization problems that can be solved locally by each node in the network. We identify these algorithms as distributed spatially-constrained localization (DSCL) and further qualify them as DSCL-L and DSCL-SL according to the use of local (L) or sub-local (SL) objective functions.

   For DSCL-L (i.e., use of local objective functions) a given sensor s_i establishes a one-to-K localization problem where K is the number of neighboring nodes. Using the known position and range estimates of neighboring nodes, the node s_i creates a discretized *spatially-constrained* search region centered at its current position estimate. Next, s_i finds a position

update inside of the spatially-constrained search area which minimizes the error distance with its K neighboring nodes. Finally, all active sensors broadcast the updated positions to repeat the full refinement step. Each sensor has the ability to stop its refinement process once a stopping criterion is locally satisfied.

For DSCL-SL (i.e., use of sub-local objective functions), a given sensor s_i establishes a one-to-one localization problem with each of its K neighbors. Using the known position and range estimates of neighboring nodes, the sensor s_i finds a set of new K candidate positions over the same number of spatially-constrained search areas. Finally, the sensor s_i updates its position by averaging over these K candidate positions. Sensors can then broadcast the updated positions to repeat the full refinement step. Each sensor has the ability to stop its refinement process once a stopping criteria is locally satisfied.

Both iterative/distributed algorithms have been tested using ToA and RSS measurements. Extensive evaluations over radio range and channel parameters are presented in which we found that our algorithms are robust over a wide gamut of values. In particular, our algorithms show excellent RMSE performance for the RSS case. Besides the RMSE values, our algorithms have the property of being computationally efficient (i.e., the number of operations needed is a function of the number of neighboring nodes), and they can be implemented with the basic resources provided by the sensor node. Also, given the distributed nature of the scheme, our proposed algorithms are scalable as long as there is way to generate an initial set of positions from anchor information.

Additionally, we introduce a realistic energy model that models consumption at the processor cycle and bit transmission levels. The model characterizes energy consumption for the localization process over the complete network. Also, with the goal of saving energy during the iterative localization process, we introduce a stopping criterion on the iterative algorithms. Experimental results show that, under certain network conditions, it is easy to determine the best tradeoff between energy and accuracy performance for a given energy budget. In this regard, we found that the combination between an initialization scheme using the LM algorithm and the DSCL-L

approach provides the best tradeoff between energy and accuracy. In addition, we found that even though the LM-based algorithm is more computationally expensive than the bilateration algorithm, the improved initialization of former provides a significant energy reduction on the iterative algorithms to offset the larger energy costs of initialization.

Finally, we developed schemes for location refinement based on the observation that nodes inside the convex hull formed the anchors tend to provide position estimates with smaller error. This work lead to a novel refinement step that applies to all the nodes in the network, and it achieves an excellent tradeoff between accuracy and energy consumption. The refinement step takes the output of one of our previous algorithms, like LM/DSCL-L, and solves a new multilateration problem using the LM algorithm. Evidence showed that even for large values of τ (the stop criteria threshold), the solution provided by the LM algorithm provides significant improvements on accuracy with minimal computational cost (i.e., energy cost). The origin of this performance seems to be founded on the globalized nature and fast convergence of this solver. Additional improvements were obtained when this refinement step was applied iteratively. Further work on this topic is underway as described on the next section.

7.2 Future Work

Although localization over WSNs has been extensively studied, it remains a challenging problem. There are many applications on military, medical and commercial settings that are taking advantage of wireless communications. For all of them, some type of localization service is required. The research presented in this dissertation will be expanded over the next years along a few lines of work. Next we describe some promising topics of research to explore in the future.

7.2.1 Localization Under Multi-Hop Communication Scenarios

Up to this point we have studied the case where anchors have a large radio range. The next scenario is to consider the case where both anchors and nodes have the same radio range. Under this perspective, the availability of sufficient anchor neighbors cannot be ensured for all nodes.

Hence, the initialization stage developed in Chapter 2 has to be revisited.

Collaborative or multi-hop communications among sensors becomes the most attractive way to overcome RF range limitations. There are two collaborative approaches that can be used to estimate distance between anchors and nodes. The first scheme is range-based localization, such a DV-dist [46], where sensors have the ability to estimate ranges with neighboring nodes (under a limited RF range) using ToA and RSS techniques. Using these range estimates, each sensor node can collaborate with other motes to estimate *the shortest-path* or *route* to at least three anchors. Once these ranges are available, the sensor can compute its initial position estimate as described in Chapter 2. To obtain the shortest path from a node to anchors a distributed algorithm similar to DV-dist [46] can be used.

On the other hand, if ranging hardware is not available on the sensor nodes, a range-free scheme should be used. In this scenario, the presence or absence of neighbor sensors is the only available information gathered by each sensor node [25, 63]. Using this information, each node indirectly estimates its distance to a set of anchors using the minimum number of hops needed to establish a connection with them. A formula to determine the range estimate R_{ik} between an anchor node a_k and a sensor node s_i is

$$R_{ik} = h_{i,k} \cdot r_k, \tag{7.1}$$

where $h_{i,k}$ is the shortest number of hops between the two nodes, and r_k is the average hop size (in meters). The hop size can be computed as [45, 66]

$$r_k = \frac{\sum_\ell \|q_k - q_\ell\|}{\sum_\ell h_{k,\ell}}, \tag{7.2}$$

where $\mathbf{q}_k = [x_k, y_k]^T$ refers to the anchor position for a_k, $\mathbf{q}_\ell = [x_\ell, y_\ell]^T$ represents ℓ known anchors, and $h_{k,\ell}$ represents the shortest number of hops from a_k to each a_ℓ neighboring anchor. Next, the value r_k is flooded by anchor a_k throughout the network. In this way, any sensor can use this value to convert the number of hops to anchor a_k to an actual range. Once a node obtains a range

to at least three anchors, it can determine an initial position estimate using any of the initialization algorithms described in Chapter 2.

7.2.2 Impact of Irregular Radio Propagation on Localization

Another interesting topic for future work consists on evaluating the impact of more realistic radiation patterns on the localization process. In real scenarios, the ideal spherical radiation pattern model for sensor nodes is unacceptable [25]. We are conscious that *wireless-connections* or *links* among nodes are neither permanent nor uniform, and broken *links* (i.e., node failures) will occur during the network lifetime. Recently, models like the Radio Irregularity Model (RIM) [75] have been introduced to understand the effect irregular propagation on WSN performance. The impact of irregular radio propagation on the MAC layer and different routing protocols has been studied in detail.

Consider the irregular radio pattern shown in Figure 7.1. R_I and R_T represent the maximum non-uniform and uniform radio ranges respectively [34, 25, 75]. For example, consider the case for node s_j which has a set $\mathbf{S}$ of N connected nodes, and consider that s_j has the radiation pattern shown in the Figure 7.1. Let $\mathbf{S}_T$ be a subset of neighbor nodes which fall inside of the uniform radio range R_T, implying they have permanent *wireless connectivity* with the sensor s_j. On the other hand, let us define the subset $\mathbf{S}_I = \mathbf{S} - \mathbf{S}_T$ as the neighbor sensors of s_j which fall within R_T and R_I. The subset N_I will have intermittent *wireless connectivity* with the sensor s_j. The natural question to ask is what would be the effect of this intermittent or irregular connectivity on the localization process?

As part of our future work, we want to evaluate the effects of irregular radiation patterns on the accuracy of the localization process and on the impact of the energy required to achieve a given level of accuracy. For instance, the neighbor density of the DSCL algorithms could show significant variations across iterations. This could translate into a slower convergence rate, requiring more iterations (and energy) to achieve good accuracy. In other words, we should take into account factors like multi-path fading, scattering, path loss, and noise [59] that generate

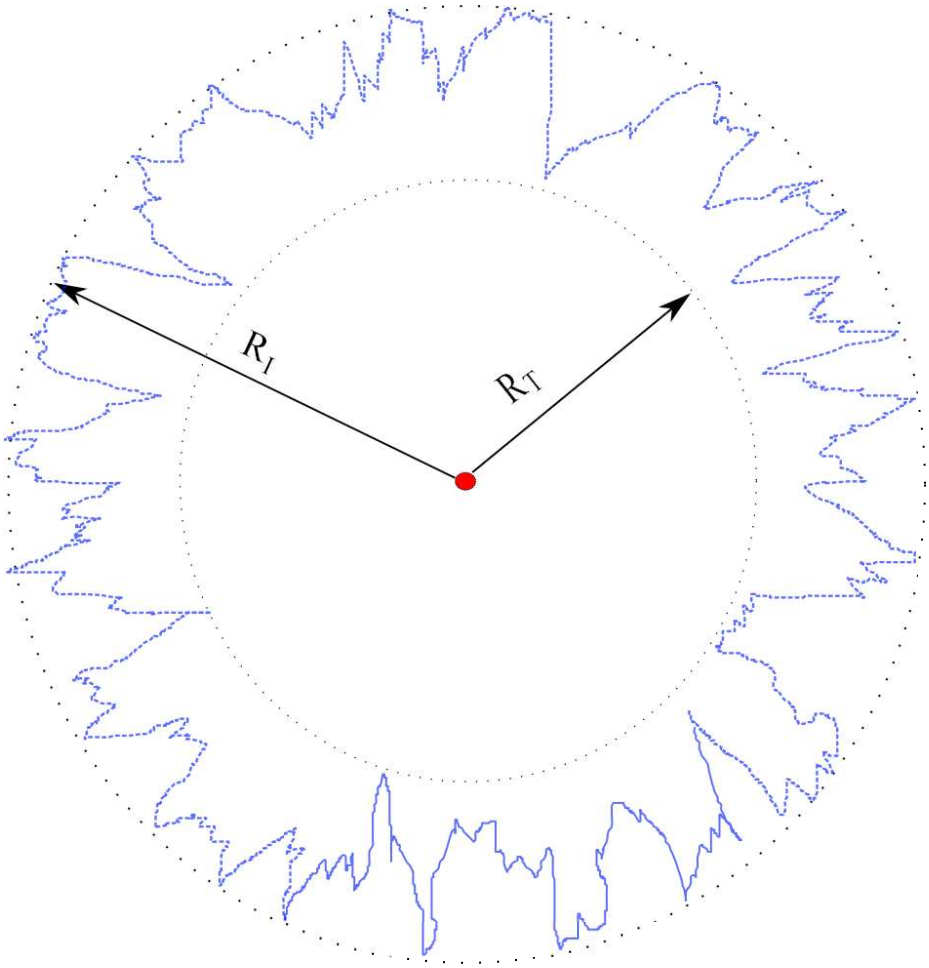


Figure 7.1: Example of an irregular radiation pattern.

large variability of the transmitted signal along all directions. We expect to obtain important conclusions under these considerations in regards to the minimum node density requirements to achieve good localization performance.

7.2.3 Further Development of Distributed-Iterative Localization Schemes

In this subsection, we present preliminary results and potential applications for the distributed/iterative multi-lateration scheme which solves a set of non-linear least-squares (NLLS) problems using the Levenberg Marquardt (LM) algorithm. Chapter 6 presented results where the LM algorithm plays a crucial roll on producing position estimates with high accuracy and low energy consumption.

Following the work from Chapter 6, a natural step was to consider a scheme where the initialization and iterative stages are implemented as the solution to NLLS problems solved with the LM algorithm. The localization method is still distributed and collaborative. As soon as the LM algorithm produces a position estimate, the node broadcast this value to its neighbors, and receives updated values from them. As before, each node repeats the process until the stopping criteria is met.

An initial evaluation of RMSE vs. τ using the average of the 20 networks described in Section 4.5.1 is shown in Figure 7.2. We can observe a great improvement from initial to final estimates, from $13.5m$ of initial error to practically $3.68m$ in average. Also, it is easy to deduce that this iterative algorithm is not affected by τ since it provides the same results independently of the imposed threshold. Figure 7.3 shows the number of iterations spent by the iterative LM algorithm at each threshold. Clearly, we can observe that better tradeoffs between accuracy and energy consumption are obtained by increasing τ which in turn decreases the number of wireless transmissions without affecting the mean RMSE. Hence for our simulation set up, the best accuracy-energy tradeoff is obtained with a $\tau = 0.2m$. For example, $\tau = 0.2$ requires around 500 iterations to provide a mean RMSE of $3.71m$. This RMSE is very close to the lowest RMSE of $3.62m$ obtained with a $\tau = 0$ (9600 wireless transmissions). Moreover, our proposed DSCL-L approach provides a mean RMSE $3.42m$ at the same number of wireless transmissions (i.e., 9600). However, the main point to highlight for this analysis is that these results are obtained by the application of the LM algorithm to the solution NLLS multi-lateration problems.

Even though the distributed LM algorithm requires more processor cycles than the DSCL-L approach, clearly the former scheme can provide a best tradeoff between energy and accuracy

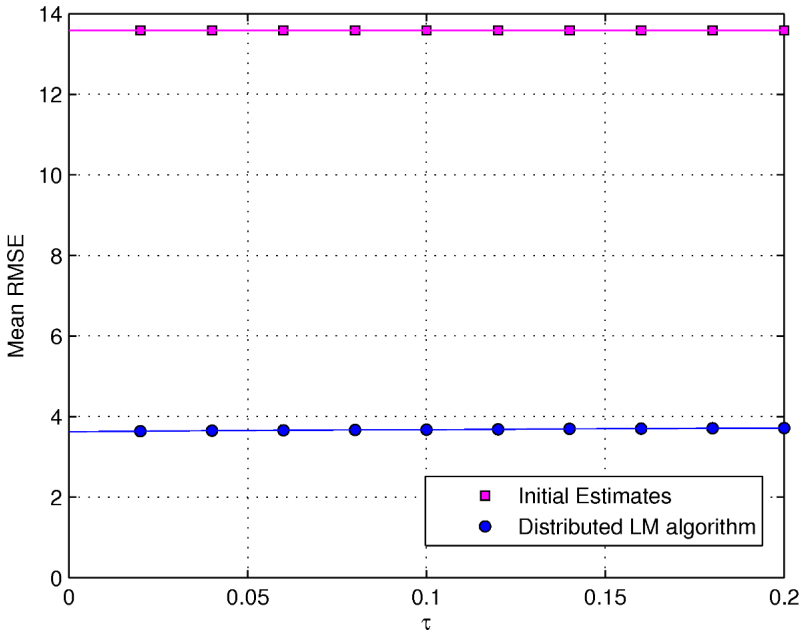


Figure 7.2: Mean RMSE results obtained by the distributed LM algorithm over 20 networks at different thresholds.

since it requires less iterations on the sensor nodes to reach optimal position estimates.

To expand on the ability of this scheme, we also found that we can improve the accuracy of the position estimates through the use of DSCL-L as a post-processing stage. Figure 7.5 shows that accuracy is improved, but the price is to use more wireless transmissions as shown Figure 7.5. As a conclusion, this scheme (distributed LM-DSCL-L) can provide a good tradeoff between accuracy performance and energy consumption. The current research shows promising results for this scheme. An in-depth analysis of this scheme will follow the work concluded on this dissertation.

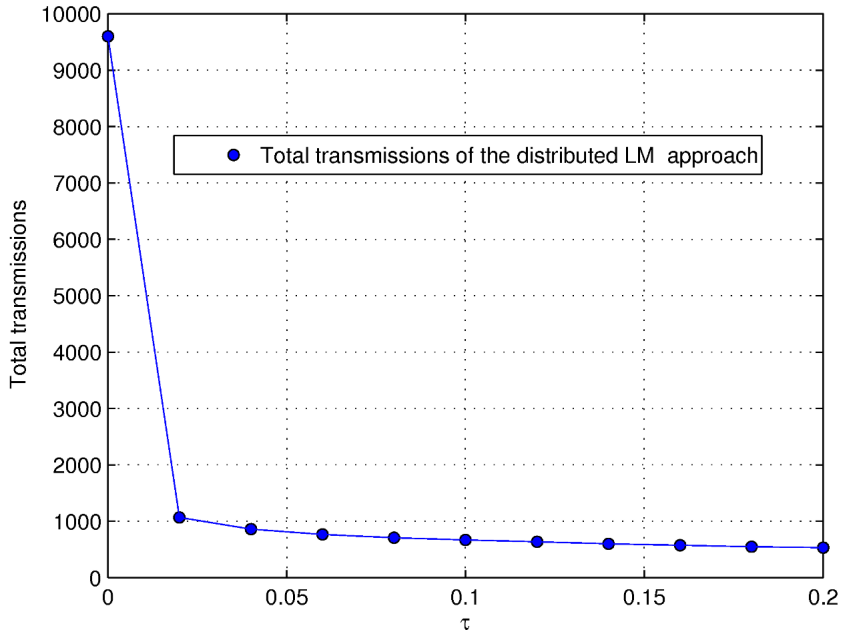


Figure 7.3: Wireless transmissions used by the distributed LM algorithm to provide final positions estimates.

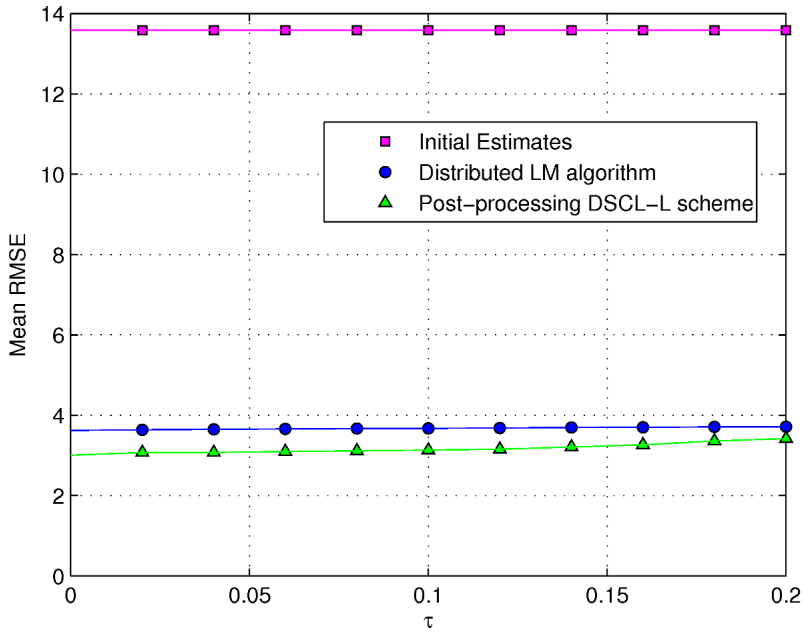


Figure 7.4: Mean RMSE results obtained by the distributed LM algorithm followed by the DSCL-L approach.

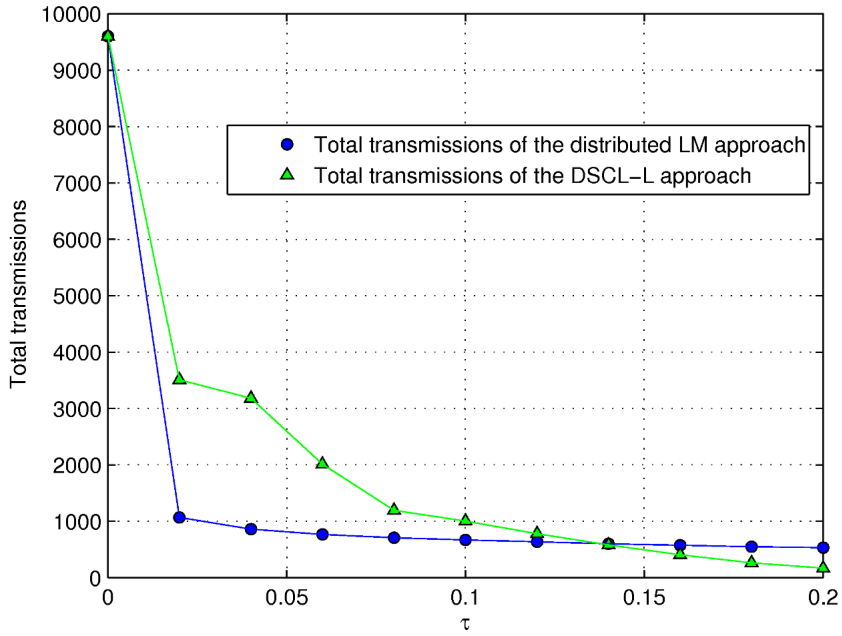


Figure 7.5: Wireless transmissions used by the distributed LM algorithm and the DSCL-L approach to find position estimates at different thresholds.

References

[1] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. A survey on sensor networks. *IEEE Communications magazine*, 40(8):102–114, 2002.

[2] J.B. Andersen, T.S. Rappaport, and S. Yoshida. Propagation measurements and models for wireless communications channels. *IEEE Communications Magazine*, 33(1):42–49, 1995.

[3] J. Aspnes, D. Goldenberg, and Y.R. Yang. On the computational complexity of sensor network localization. *Algorithmic Aspects of Wireless Sensor Networks*, pages 32–44, 2004.

[4] J.A.R. Azevedo. Sistemas de Última Generación para la Observación, Predicción y Vigilancia Activa de Espacios Naturales Forestales en la Macaronesia. 2008.

[5] J.A.R. Azevedo and F.E. Santos. Signal Propagation Measurements with Wireless Sensor Nodes. 2007.

[6] P. Biswas, T.C. Lian, T.C. Wang, and Y. Ye. Semidefinite programming based algorithms for sensor network localization. *ACM Transactions on Sensor Networks (TOSN)*, 2(2):220, 2006.

[7] C. Bohm and H.P. Kriegel. Determining the convex hull in large multidimensional databases. *Data Warehousing and Knowledge Discovery*, pages 294–306, 2001.

[8] Paul Bourke. Intersection of two circles, 1997.

[9] M.W. Carter, H.H. Jin, M.A. Saunders, and Y. Ye. Spaseloc: An adaptive subproblem algorithm for scalable wireless sensor network localization. *SIAM Journal on Optimization*, 17(4):1102–1128, 2007.

[10] B. Champagne, S. Bédard, and A. Stéphenne. Performance of time-delay estimation in the

presence of room reverberation. *IEEE Transactions on Speech and Audio Processing*, 4(2): 148–152, 1996.

[11] KW Chan and HC So. Accurate sequential weighted least squares algorithm for wireless sensor network localization. In *Proc. Euro. Signal Processing Conf.(EUSIPCO)*. Citeseer.

[12] H. Chen, K. Sezaki, P. Deng, and H.C. So. An improved DV-Hop localization algorithm for wireless sensor networks. In *Industrial Electronics and Applications, 2008. ICIEA 2008. 3rd IEEE Conference on*, pages 1557–1561. IEEE, 2008.

[13] B.H. Cheng, L. Vandenberghe, and K. Yao. Distributed algorithm for node localization in wireless ad-hoc networks. *ACM Transactions on Sensor Networks (TOSN)*, 6(1):1–20, 2009.

[14] C.Y. Chong, SP Kumar, and B.A. Hamilton. Sensor networks: Evolution, opportunities, and challenges. *Proceedings of the IEEE*, 91(8):1247–1256, 2003.

[15] C.W. Commander, M.A. Ragle, and Y. Ye. Semidefinite programming and sensor network localization problem, SNLP. 2008.

[16] T.H. Cormen. *Introduction to algorithms*. The MIT press, 2001. ISBN 0262032937.

[17] F. Delbos, D. Sinoquet, J.C. Gilbert, and R. Masson. A trustregion Gauss–Newton method for reflection tomography. *KIM Annual Report, Institut Français du Pétrole, Rueil-Malmaison, France, http://consortium. ifp. fr/KIM*, 2001.

[18] J.E. Dennis and R.B. Schnabel. *Numerical methods for unconstrained optimization and nonlinear equations*. Society for Industrial Mathematics, 1996. ISBN 0898713641.

[19] H. Dietz, B. Dieter, R. Fisher, and K. Chang. Floating-point computation with just enough accuracy. *Computational Science–ICCS 2006*, pages 226–233, 2006.

[20] L. Doherty, K.S.J. Pister, and L. El Ghaoui. Convex position estimation in wireless sensor networks. In *IEEE INFOCOM*, volume 3, pages 1655–1663. Institute of electrical engineers inc. (IEEE), 2001.

[21] T. Eren, OK Goldenberg, W. Whiteley, YR Yang, AS Morse, BDO Anderson, and PN Belhumeur. Rigidity, computation, and randomization in network localization. In *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, volume 4, 2004.

[22] M. Fayed and H.T. Mouftah. Localised convex hulls to identify boundary nodes in sensor networks. *International Journal of Sensor Networks*, 5(2):112–125, 2009. ISSN 1748-1279.

[23] S. Gezici. A survey on wireless position estimation. *Wireless Personal Communications*, 44(3):263–282, 2008.

[24] L. Girod, V. Bychkovskiy, J. Elson, and D. Estrin. Locating tiny sensors in time and space: A case study. In *2002 IEEE International Conference on Computer Design: VLSI in Computers and Processors, 2002. Proceedings*, pages 214–219, 2002.

[25] T. He, C. Huang, B.M. Blum, J.A. Stankovic, and T. Abdelzaher. Range-free localization schemes for large scale sensor networks. In *Proceedings of the 9th annual international conference on Mobile computing and networking*, pages 81–95. ACM, 2003.

[26] B. Hendrickson. The molecule problem: Exploiting structure in global optimization. *SIAM Journal on Optimization*, 5(4):835–857, 1995.

[27] J.L. Hennessy, D.A. Patterson, and D. Goldberg. *Computer architecture: a quantitative approach*. Morgan Kaufmann, 2003. ISBN 1558607242.

[28] Y. Huang, J. Benesty, G.W. Elko, and RM Mersereati. Real-time passive source localization: A practical linear-correction least-squares approach. *Speech and Audio Processing, IEEE Transactions on*, 9(8):943–956, 2002. ISSN 1063-6676.

[29] Y. Huang, J. Benesty, J. Chen, and SpringerLink (Online service). *Acoustic MIMO signal processing*. Springer, 2006. ISBN 3540376305.

[30] IPv6. 6lowpan, 2011. `http://en.wikipedia.org/wiki/6LoWPAN#Adapting_the_packet_sizes_of_the_two_networks`.

[31] N. Patwari J. A. Costa and Alfred O. HeroIII. Distributed Weighted-Multidimensional Scaling for Node Localization in Sensor Networks. *ACM Transactions on Sensor Networks*, 2(1):39–64, Feb 2006.

[32] CT Kelley. *Iterative methods for optimization*. Society for Industrial Mathematics, 1999.

[33] U.A. Khan, S. Kar, and J.M.F. Moura. Distributed sensor localization in random environments using minimal number of anchor nodes. *Signal Processing, IEEE Transactions on*, 57(5):2000–2016, 2009. ISSN 1053-587X.

[34] L. Kleinrock and J. Silvester. Optimum transmission radii for packet radio networks or why six is a magic number. In *Proceedings of the IEEE National Telecommunications Conference*, volume 4, pages 1–4, 1978.

[35] B. Krishnamachari. *Networking wireless sensors*. Cambridge Univ Pr, 2005.

[36] N. Krislock, V. Piccialli, and H. Wolkowicz. Robust semidefinite programming approaches for sensor network localization with anchors. *Report, Department of Combinatorics and Optimization, University of Waterloo, Waterloo*, 2006.

[37] O. Landsiedel, K. Wehrle, and S. Gotz. Accurate prediction of power consumption in sensor networks. In *Proceedings of The Second IEEE Workshop on Embedded Networked Sensors (EmNetS-II)*. Citeseer, 2005.

[38] V.D. Le, V.H. Dang, S. Lee, and S.H. Lee. Distributed Localization in Wireless Sensor Networks based on Force-Vectors. In *Intelligent Sensors, Sensor Networks and Information Processing, 2008. ISSNIP 2008. International Conference on*, pages 31–36, 2008.

[39] X. Li, B. Hua, Y. Shang, and Y. Xiong. A robust localization algorithm in wireless sensor networks. *Frontiers of Computer Science in China*, 2(4):438–450, 2008. ISSN 1673-7350.

[40] J. Liu, Y. Zhang, and F. Zhao. Robust distributed node localization with error management. In *Proceedings of the 7th ACM international symposium on Mobile ad hoc networking and computing*, pages 250–261. ACM, 2006. ISBN 1595933689.

[41] D. Lymberopoulos, Q. Lindsey, and A. Savvides. An empirical characterization of radio signal strength variability in 3-d ieee 802.15. 4 networks using monopole antennas. *Wireless Sensor Networks*, pages 326–341, 2006.

[42] R. Mansi. Enhanced Quicksort Algorithm.

[43] C. Meng, Z. Ding, and S. Dasgupta. A semidefinite programming approach to source localization in wireless sensor networks, 2008.

[44] R.L. Moses, D. Krishnamurthy, and R.M. Patterson. A self-localization method for wireless sensor networks. *EURASIP Journal on Applied Signal Processing*, pages 348–358, 2003.

[45] D. Niculescu and B. Nath. Ad hoc positioning system (APS) using AOA. In *IEEE INFOCOM*, volume 22, pages 1734–1743. Citeseer, 2003.

[46] D. Niculescu and B. Nath. DV based positioning in ad hoc networks. *Telecommunication Systems*, 22(1):267–280, 2003.

[47] J. Nie. Sum of squares method for sensor network localization. *Computational Optimization and Applications*, 43(2):151–179, 2009.

[48] J. Nocedal and S.J. Wright. *Numerical Optimization*. Springer, 2006.

[49] A.O. HeroIII Patwari, N.S. Correal M. Perkins, and R.J. ODea. Relative location estimation in wireless sensor networks . *IEEE trans signal processing*, 51(8):2137–2148, August 2003.

[50] N. Patwari. Wireless sensor network localization measurement repository, 2003. `http://www.eecs.umich.edu/~hero/localize/`.

[51] N. Patwari. *Location estimation in sensor networks*. PhD thesis, Citeseer, 2005.

[52] N. Patwari, J.N. Ash, S. Kyperountas, AO Hero, R.L. Moses, and N.S. Correal. Locating the nodes. *IEEE Signal processing magazine*, 22(4):54, 2005.

[53] F.P. Preparata and M.I. Shamos. *Computational geometry: an introduction*. Springer, 1985. ISBN 0387961313.

[54] Y. Qi, H. Kobayashi, and H. Suda. Analysis of wireless geolocation in a non-line-of-sight environment. *Wireless Communications, IEEE Transactions on*, 5(3):672–681, 2006.

[55] V. Raghunathan, C. Schurgers, S. Park, and M.B. Srivastava. Energy-aware wireless microsensor networks. *Signal Processing Magazine, IEEE*, 19(2):40–50, 2002. ISSN 1053-5888.

[56] T.S. Rappaport et al. *Wireless communications: principles and practice*, volume 207. Prentice Hall PTR New Jersey, 1996.

[57] M. Rudafshani and S. Datta. Localization in wireless sensor networks. In *Proceedings of the 6th international conference on Information processing in sensor networks*, page 60. ACM, 2007.

[58] A. Savvides, C.C. Han, and M.B. Strivastava. Dynamic fine-grained localization in ad-hoc networks of sensors. In *Proceedings of the 7th annual international conference on Mobile computing and networking*, pages 166–179. ACM New York, NY, USA, 2001.

[59] P.M. Shankar. *Introduction to wireless systems*. John Wiley & Sons, 2002.

[60] A. Sinha and A.P. Chandrakasan. Energy aware software. In *VLSI Design, 2000. Thirteenth International Conference on*, pages 50–55. IEEE, 2000. ISBN 0769504876.

[61] A.M.C. So and Y. Ye. Theory of semidefinite programming for sensor network localization. *Mathematical Programming*, 109(2):367–384, 2007.

[62] R. Stoleru and J.A. Stankovic. Probability grid: A location estimation scheme for wireless sensor networks. In *Sensor and Ad Hoc Communications and Networks, 2004. IEEE SECON 2004. 2004 First Annual IEEE Communications Society Conference on*, pages 430–438. IEEE, 2004. ISBN 0780387961.

[63] R. Stoleru, T. He, and J.A. Stankovic. Range-free localization. *Secure Localization and Time Synchronization for Wireless Sensor and Ad Hoc Networks*, pages 3–31, 2007.

[64] R. Stoleru, J.A. Stankovic, and S.H. Son. Robust node localization for wireless sensor networks. In *Proceedings of the 4th workshop on Embedded networked sensors*, page 52. ACM, 2007.

[65] TI-Algorithms. Optimized sort algorithms for dsp, 2011. `http://processors.wiki.ti.com/index.php/Optimized_Sort_Algorithms_For_DSP`.

[66] R. Verdone, D. Dardari, G. Mazzini, and A. Conti. *Wireless sensor and actuator networks: technologies, analysis and design*. Elsevier, 2008.

[67] V. Vivekanandan and V.W.S. Wong. Ordinal MDS-based localisation for wireless sensor networks. *International Journal of Sensor Networks*, 1(3):169–178, 2006.

[68] A. Wang and A. Chandrakasan. Energy-efficient DSPs for wireless sensor networks. *Signal Processing Magazine, IEEE*, 19(4):68–78, 2002. ISSN 1053-5888.

[69] Z. Wang, S. Zheng, S. Boyd, and Y. Ye. Further relaxations of the SDP approach to sensor network localization. *SIAM J. Optim*, 19(2):655–673, 2008.

[70] Application Note XST-AN019a. Xbee and xbee-pro oem rf module antenna considerations, Sep 2005. `http://www.digi.com`.

[71] Z. Yang and Y. Liu. Quality of trilateration: Confidence-based iterative localization. *IEEE Transactions on parallel and distributed systems*, pages 631–640, 2009. ISSN 1045-9219.

[72] N. Ye. *The handbook of data mining*. Lawrence Erlbaum, 2003. ISBN 0805840818.

[73] W. Ye, J. Heidemann, and D. Estrin. An energy-efficient MAC protocol for wireless sensor networks. In *IEEE INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings*, volume 3, 2002.

[74] Kegen Yu and Y. Jay Guo. Robust Localization in Multihop Wireless Sensor Networks. *IEEE Xplore*, 2008.

[75] G. Zhou, T. He, S. Krishnamurthy, and J.A. Stankovic. Impact of radio irregularity on wireless sensor networks. In *Proceedings of the 2nd international conference on Mobile systems, applications, and services*, pages 125–138. ACM, 2004.

Curriculum Vita

Juan de Dios Cota Ruiz was born on July 26, 1975. His birthplace is Navojoa, Sonora, Mexico; He received the BS in Electrical Engineering degree from Sonora Institute of Technology, Sonora, México, in 1998; and the MS in Electrical Engineering from the Chihuahua Institute of Technology, Chihuahua, Mexico, in 2003. He obtained his Ph.D. in Electrical and Computer Engineering at the University of Texas at El Paso (UTEP), in 2011. He has worked on the manufacturing industry for 3 years as a Test Engineer. Dr. Juan de Dios Cota Ruiz received for three years the PROMEP scholarship and the Chihuahua-UTEP scholarship for two years.

Currently, he works as a full research professor at the Autonomous University of Ciudad Juárez (UACJ), since 2003. He has national and international publications on the fields of electrical engineering, and is holder of a national patent. His current research interests include Instrumentation, Optimization, Wireless Sensor Networks, and Digital Signal Processing.

Permanent Address: Privada Andalucia #2215-5
Ciudad Juárez, Chih. México
C.P. 32470