Salvador Jauregui Ortiz

Localization and tracking of nodes in a wireless sensor network based on information fusion and context awareness

Dissertação apresentada ao Programa de Pós-Graduação em Informática da Pontifícia Universidade Católica do Paraná como requisito parcial para obtenção do título de Doutor em Informática.

Curitiba
2013

Salvador Jauregui Ortiz

Localization and tracking of nodes in a wireless sensor network based on information fusion and context awareness

Dissertação apresentada ao Programa de Pós-Graduação em Informática da Pontifícia Universidade Católica do Paraná como requisito parcial para obtenção do título de Doutor em Informática.

Área de Concentração: Computer Science

Orientador: Dr. Mario Ángel Siller González Pico
Co-orientador: Dr. Edson Emilio Scalabrin

Curitiba
2013

Salvador, Jauregui Ortiz
Localization and tracking of nodes in a wireless sensor network based on information fusion and context awareness. Curitiba, 2013.

Dissertação - Pontifícia Universidade Católica do Paraná. Programa de Pós-Graduação em Informática.

1. Wireless Sensor Network 2. Location & Tracking 3. Information Fusion & Context Awareness I.Pontifícia Universidade Católica do Paraná. Centro de Ciências Exatas e Tecnologia. Programa de Pós-Graduação em Informática II - t

I lovingly dedicate this thesis to my wife Leticia and my parents Beatriz y José, who supported me each step of the way.

Acknowledgements

Foremost, Dr. Mario Siller and Dr. Edson Scalabrin have been the ideal thesis supervisors. Their sage advice, insightful criticisms, and patient encouragement aided the writing of this thesis and several articles in innumerable way. I express my gratitude and deep appreciation to them. I could not have imagined having a better advisor and co-advisor for my Ph.D study.

I would also like to thank Dr. Félix Ramos whose steadfast support during my Ph.D research was greatly needed and deeply appreciated. My sincere thanks also goes to Dr. Evangelos Kranakis and Dr. Michel Barbeau for sharing me their expertise and leading me working on an exciting project.

I thank my fellow labmates of Centro de Investigación y de Estudios Avanzados, Pontifícia Universidade Católica do Paraná, and Carleton University for their pleasant company and friendship.

I graciously acknowledge the final support received from National Council of Science and Technology (CONACYT) and Emerging Leaders in the Americas Program (ELAP) to conduct my Ph.D study and research.

Last but not the least, I would like to thank my family. My wife Leticia and my parents Beatriz and José for their support and unconditional love.

Contents

List of Figures

List of Tables

List of Algorithms

List of Symbols

X	The hidden states
Y	Observation process
dis	The distance between nodes
μ_0	The hypothesis population mean
σ	The population standard deviation
ρ	The Obstacle Level Indicator
z	Statistical value
sm	The sample mean
sn	The sample size
ra	The region of acceptance
l_i	Sequential locating point from the main system
Y_i	Observations
i	Instance time
$E[x_i]$	Expected value of the subsystem
(x, y)	Two dimensional Cartesian space
θ_0	Initial orientation
$speed_0$	Initial speed
thr	Threshold
$speed_i$	Speed at instance time i
$N(\mu_s, \sigma_s^2)$	Normal distribution for speed
ξ_i	Noise in Observations
$N(\mu_Y, \sigma_Y^2)$	Normal distribution for observations' noise
ψ_i	Gaussian noise for l_i
$p(x_i\|Y_{0:i})$	The posterior probability distribution
w_0^p	The probability for each particle
x_i^p	Particle
ϕ_i	Gaussian noise for particles

List of Abbreviations

RSSF	*Rede de Sensores Sem Fio*
ASH-LN	*Arquitetura Subsumption Hierárquica para Localização de Nós*
RSI	*Red de Sensores Inalámbrica*
HSA-NL	*The Hierarchical Subsumption Architecture for Node Localization*
WSN	*Wireless Sensor Network*
NLA	*Node Localization Algorithm*
AoA	*Angle of Arrival*
RSSI	*Received Signal Strength Indicator*
ToA	*Time of Arrival*
TDoA	*Time Different of Arrival*
GPS	*Global Position System*
APIT	*Approximate Point-In-Triangulation*
CL	*Centroid Localization*
DV-Hop	*Distance Vector-Hop*
WCL	*Weighted Centroid Localization*
TCL	*Triangular Centroid Localization*
DOI	*Degree Of Irregularity*
ICL	*Improved Centroid Localization Algorithm*
LQI	*Link Quality Indicator*
ABS	*Active Badge System*
SIR	*Sequential Importance Resampling*
VL	*Vectorial Localization*
ECL	*Energized Centroid Localization*
ID	*IDentification number*
SzL	*Subzone Localization*
OLI	*Obstacle Level Indicator*
UD	*Unknown Distribution*
SBNs	*Smart Beacon Nodes*
HT	*Hypothesis Testing*

LPN	*Logical Position of Nodes*
AVU	*Area/Volume of Uncertainty*
SEA-NL	*The Smart Environmental Architecture for Node Localization*
CAA-NL	*The Context-Aware Architecture for Node Localization*
PRMM	*Probabilistic Random Mobility Model*
CTR	*Center Turning Radius*
SoU	*Space of Uncertainty*
LBS	*Localization Based Services*

Resumo

Uma RSSF (*Rede de Sensores Sem Fio*) é formada por vários dispositivos (nós) capazes de coletar informações do ambiente tais como temperatura, vibração, umidade, som, luz e movimento. Além disso, a sua comunicação sem fio permite implantar rapidamente uma rede composta de centenas de nós. Em desastres naturais, uma RSSF pode ajudar as equipes de resgate na busca por sobreviventes. Uma RSSF também pode ser empregada na detecção de incêndios florestais. A capacidade de localizar nós automaticamente é o fator chave de muitas aplicações móveis de RSSF. Outras aplicações incluem nós móveis como veículos, trens, robôs, pedestres e animais. O rastreamento de nós abertos permite uma grande variedade de aplicações em domínios como exploração, navegação, agricultura, militar e de saúde.

Esta tese aborda o problema de localização de nós na perspectiva de fusão de informações e reconhecimento do contexto. Em particular, esta tese aborda (i) a localização do nó em RSSF para o interior e exterior, bem como (ii) o rastreamento do nó móvel em áreas sombreadas. Em (i), a ASH-LN (*Arquitetura Subsumption Hierárquica para Localização de Nós*) é proposta a qual realiza a fusão de algoritmos de localização de nós para melhorar a estimativa por meio de sua execução sequencial. Além disso, um conjunto de seis algoritmos de localização de nós é também proposto de modo a alimentar o ASH-NL. Em (ii), nós propomos um Modelo Móvel Aleatório Probabilístico para geração de caminhos de qualquer veículo, bem como um algoritmo para rastreamento de nós móveis em áreas sombreadas baseado no reconhecimento do contexto e filtragem de partículas.

Palavra-chaves: (Rastreamento de nós, reconhecimento do contexto, fusão da informação, modelo de mobilidade, filtragem de partículas.)

Resumen

Una RSI (*Red de Sensores Inalámbrica*)) está formada por varios dispositivos (nodos) capaces de recoger información del ambiente tales como temperatura, vibraciones, humedad, sonido, luz, y movimiento. Además, su comunicación inalámbrica nos permite desplegar rápidamente una red compuesta por cientos de nodos. En los desastres naturales, una RSI puede ayudar a los equipos de rescate a encontrar sobrevivientes. Un RSI también puede emplearse para la detección oportuna de incendios forestales. La capacidad de localizar automáticamente los nodos es un factor clave para muchas aplicaciones basadas en una RSI. Otras aplicaciones importantes de localización incluyen nodos móviles tales como vehículos, trenes, robots, peatones, y animales. El rastreo de nodos abre una amplia gama de aplicaciones en el dominio de la exploración, navegación, agricultura, militar y salud.

Esta tesis aborda el problema de la localización de nodos desde la perspectiva de la fusión de información y reconocimiento del contexto. En particular, la presente investigación doctoral está enfocada en (i) la localización de nodos en una RSI para interiores y exteriores, así como (ii) el seguimiento de un nodo móvil en zonas de sombra. En (i), se propone HSA-NL (*The Hierarchical Subsumption Architecture for Node Localization*), la cual fusiona los algoritmos de localización simples con el fin de mejorar la estimación a través de su ejecución secuencial. Además, se propone un conjunto de seis algoritmos de localización para alimentar a la arquitectura HSA-NL. En (ii), se propone un modelo de movilidad probabilístico aleatorio para la generación de trayectorias de cualquier vehículo, así como un algoritmo para el rastreo de un nodo móvil en zonas de sombra basado en el reconocimiento del contexto y filtrado de partículas.

Palabras claves: (Localización, rastreo, reconocimiento del contexto, fusión de la información, modelo de mobilidad, filtro de particulas)

Abstract

A WSN (*Wireless Sensor Network*) is formed by various devices (nodes) capable of collecting information from the environment such as temperature, vibration, humidity, sound, light, and motion. Besides, their wireless communication allows us to quickly deploy a network composed of hundreds of nodes. In natural disasters, a WSN can help rescue teams to find survivors. A WSN can be also employed for an early detection of forest fires. The capability of automatically locating nodes is a key factor in many WSN applications. Other major localization applications include mobile nodes such as vehicles, trains, robots, pedestrians, and animals. The node tracking opens up a wide range of applications in exploration, navigation, agriculture, military and health domains.

This thesis deals with the node localization problem from the perspective of information fusion and context awareness. In particular, this thesis addresses (i) the node localization in a WSN for indoors and outdoors as well as (ii) the mobile node tracking in shaded areas. In (i), the Hierarchical Subsumption Architecture for Node Localization (HSA-NL) is proposed which fuses node localization algorithms in order to improve the estimation through their sequential execution. Besides, a set of six node localization algorithms are also proposed in order to fuel HSA-NL. In (ii), we propose a Probabilistic Random Mobility Model for generating paths of any vehicle as well as an algorithm for tracking a mobile node in shaded areas based on context awareness and particle filtering.

Keywords: (Localization, tracking, context awareness, information fusion, mobility model, particle filtering)

Chapter 1

Introduction

1.1 Localization and tracking of nodes

A WSN (*Wireless Sensor Network*) consists of various devices (nodes) capable of collecting information from their environment through sensors incorporated such as temperature, humidity, sound, light, and motion, among others. Some of the main characteristics of a WSN include rapid deployment, scalability, application oriented configuration, and in most cases are low in cost. A WSN can be used in several applications such as habitat monitoring, e-health, helping rescue teams in events of disasters, military surveillance, etc. A WSN can be used for early detection of forest fires [1, 2]. The forest is considered one of the most important and vital natural resource, besides it plays an important role in the ecological balance of the earth. An early detection of a forest fire could reduce the damaged area considerably and the wildlife could be preserved.

The monitoring of crop fields is not recent; it can be realized by the analysis of images taken from satellites or an aircraft [3] or by using infrared and visible-light images [4]. However, a WSN can be used for a fine monitoring, because it can detect nutrient deficiencies, diseases, water deficiency or surplus, water flow, weed infestations, insect damage, hail damage, wind damage, herbicide damage, and plant populations, among others. In a specific case, if an agriculturist wants to know the air temperature and humidity, the soil temperature and moisture, the CO2 concentration, and the illumination intensity, the best solution might be a WSN application such as [5], which is based on Greenhouse [6]. In the latter paper, an optimal environment can be achieved with a Greenhouse, it involves continuous monitoring and activation of different units, such as heating, cooling, lighting, water/soil gradient and so on, in order to maintain the most favorable environment and protect crops from external climate. A WSN plays an important role

from an economic perspective respect to satellites or an aircraft. Besides, captured data can be more specific such as the CO2 concentration.

Natural disasters such as avalanches, earthquake, floods, tornadoes, tsunamis, blizzard, and so forth usually destroy the communication network infrastructure. A WSN can be used for create a rapid structure to sense environmental data and prevent hazards [7]. WSN is not except to be damaged by natural disasters; nevertheless it can recover itself and transmit urgent data effectively [8]. Furthermore, the WSN can work undersea [8] for data collecting, pollution monitoring, assisted navigations, exploration of natural underwater sea resources, and so on.

We have highlighted that WSN applications can be quite useful in diverse domains. Undoubtedly the node position plays an important role. In the case of forest fires, we would like to know as soon as possible where the fire is starting, specifically, the geographical position of the node that reports the incident. Where is the node? This is the fundamental question that motivates to the community of researchers that are working in the area of localization of nodes. Nowadays, there are numerous Node Localization Algorithms (NLAs) with different accuracy and approach. Despite the high relative precision of some algorithms, this is still an opened research area due to: (i) physical phenomena such as the attenuation, reflection, diffraction, and scattering. These constraints are discussed in greater detail in [9]; (ii) hardware constraints such as the storage capacity, the energy consumption, the low computational power availability, and the radiation pattern of antennas; (iii) the kind of environment such as indoors, outdoors, or both; and (iv) Environment's factors such as temperature, humidity, and obstacles. The inclusion of all previous factors into a single NLA (*Node Localization Algorithm*) might not possible or feasible to achieve. Some solutions for the localization of nodes are range-based NLA in which distances or angles among nodes are employed. Some NLAs of this kind include the AoA (*Angle of Arrival*) [10, 11, 12, 13], RSSI (*Received Signal Strength Indicator*) [11], ToA (*Time of Arrival*)[11], TDoA (*Time Different of Arrival*) [11], PinPtr [14], the SpotON [15], and GPS (*Global Position System*). The latter is not operational indoors because the line-of-sight is required and sometimes it is not the best solution from an economical perspective. On the other hand, the range-free NLAs use only contents of received messages. Example of this kind of NLAs are the APIT (*Approximate Point-In-Triangulation*) [16, 11], CL (*Centroid Localization*) [17, 11], Bounding box [18, 11], and DV-Hop (*Distance Vector-Hop*) [19, 11]. On the other hand, there are NLAs that can be classified as range-based and range-free i.e., hybrid NLAs. For instance, the WCL (*Weighted Centroid Localization*) [20], TCL (*Triangular Centroid Localization*) [21], and algorithm presented in [22].

Generally speaking the accuracy performance of NLAs is better under specific conditions. For instance CL, the authors evaluated their simple-idealized radio model in a real scenario. Experimental results show different error performance for the outdoor and indoor scenario. For the former, the error never exceeds the 2 meters while for the latter the maximum error is 22.3 meters. In [16], is presented via simulation other accuracy performance of the CL considering the variation of: (i) the node density, (ii) beacons heard, (iii) the beacon node range ratio, and (iv) the pattern irregularity of the signal. Again, simulation results show that the CL has different error performance. The minimum average error of the CL is $0.1R$ and the maximum average error is $3.1R$ such that R is the units of node radio range.

The integration of all factors of a given complex environment into a simple NLA might be not possible and feasible to achieve. Some factors represent an authentic challenge but itself, and the benefit in node localization may be poor. The obtaining of an optimal equation is not an easy task, if the environment status changes the accuracy of the NLA also changes. We want to highlight that there is no a universal NLA which obtains the same high accuracy anywhere. Instead of, some authors have created NLAs to be executed in specific places such as the RADAR [23] which was developed in a fully-observable environment (second floor of a three-storey building). The authors included a user location in four orientations: north, south, east, and west. Besides, they considered the number of walls, whereof they incorporated the wall attenuation factor into the floor attenuation factor propagation model [24], hereof they proposed the wall attenuation factor model.

In order to increase the application range of NLAs a composition of them might be required. In [25, 26] a composition of NLAs is proposed in a hierarchical framework considering two main sequential execution phases: (i) range-based NLAs and (ii) range-free NLAs. The execution order from this approach can be explained based on the premise that range-based NLAs are normally more accurate than range-free one. On this basis alone, the estimated position of nodes cannot be improved through the execution phases.

1.2 Motivation

A WSN has been used in several applications as stated earlier. Undoubtedly, the node localization is a key factor and it is directly related with the profit. A high accuracy in node localization is demanded by some critical and non-critical applications. We are motivated by the idea that some NLAs are able to work together in order to provide a flexible solution since there is no a universal NLA. It means a NLAs fusion might be

required. We believe that a NLA should be able to react to an environment as a context-aware application in order to improve its node localization estimation. The NLA fusion and context awareness seem to be a flexible and appropriate solution for critical and non-critical applications in a WSN.

On the other hand, many algorithms and applications use GPS as standard for outdoor usage. But it cannot performance correctly for shaded areas such as tunnels, canyons, and near large buildings. Other localization algorithms and reference information might be then required for aiding GPS. In this sense, we are also motivated by the idea that a mobile node is able to keep tracking by itself even in shaded areas.

1.3 Problem description

The main node localization and tracking problems are well-defined in the node localization community so that each problem opens up a vast research subarea. They are listed below.

- The type of antenna.
- The WSN's density.
- The node's mobility.
- Physical phenomena (diffraction, reflection, reflation, and scattering).
- The diversity of places (either indoors or outdoors).
- Diversity of obstacles (persons, walls, and any possible object).
- High performance and low cost.
- The wave propagation irregularity.
- The distance estimation among nodes.
- Noise in sensors.
- Energy consumption and Complexity.

Some context problems have been considered slightly and suggested to be explored. For instance, the node position on the user, either on his belt or in his shirt pocket. Maybe, this is a very fine problem, but it was already taken into account [27]. The importance of each of the problems described above depends entirely on the application environment

and their importance can vary from one environment to another. This research focuses on the problems of the diversity of places and obstacles.

1.4 General objectives

The aim of this doctoral work is not to solve the node localization and tracking problem with a universal solution. The objective is proof that Information Fusion and Context Awareness are useful and flexible for both localization in a WSN and tracking of a node in shaded areas.

1.5 Specific objectives

Formulate and develop an architecture that allows the sequential fusion of node localization algorithms and uses then context awareness for improving the estimated position of nodes. Formulate and develop an architecture that allows tracking a node even in shaded areas. In order to accomplish these objectives the following duties are required:

- Review the state of the art of the localization and tracking of nodes, context awareness, information fusion, distance estimation, and particle filtering.
- Formulate and develop several single NLAs in order to fuel the proposed architectures for node localization problem in a WSN.
- Validate the two architectures through accuracy and performance comparisons.

1.6 Hypothesis

We have one only hypothesis which involves all the work performed.

- Information Fusion and Context Awareness working together are able to provide a flexible solution for the node localization in a WSN as well as tracking of a node in shaded areas.

1.7 Document organization and main contributions

1.7.1 Organization

The document is organized into nine chapters as follow: The section 2 shows the related work and background of localization and tracking of nodes bearing in mind information fusion and context awareness. The section 3 describes the seven proposed NLAs. The section 4 details four proposed architectures for node localization in WSN. The section 5 shows the test scenarios and experimental results. The section 6 finalizes the first thesis' part with ongoing work and conclusion. In the second thesis' part node tracking is addressed. The section 7 details the proposed architecture. The section 8 shows the test scenarios and experimental results. Finally, the section 9 concludes the second thesis' part with ongoing work and conclusion.

1.7.2 Main contributions

The main contributions of Part I (LOCALIZATION) are:

- Three architecture for node localization in a Wireless Sensor network:

  The Logical Position of Nodes.

  Smart Node Architecture for Node Localization.

  The Hierarchical Subsumption Architecture for Node Localization.

- Six node localization algorithms:

  The Vectorial Localization Algorithm.

  The Energized Centroid Localization.

  The Subzone Localization algorithm.

  The Triangular Centroid Localization.

  The Smart Beacon Nodes.

  The Logical Position of Nodes.

- A novel solution for node localization considering the variations of the energy consumption of nodes.

- A research subarea focused on developing NLAs with an execution stamp n.

- The use of context awareness for improving the estimated position of a node.

- We proved that for the studied NLAs the number of required beacon nodes can be highly reduced by the use of context awareness without affecting their average accuracy (it is believed that this is also true for similar NLAs to the studied ones).
- We successfully introduced the subsumption principle to the area of context aware computing for a localization process.

The main contributions of Part II (TRACKING) are:

- An architecture for locating of a mobile node during shaded areas.
- We successfully introduced the suppress principle of the subsumption architecture [28] into our proposed architecture as a priority-selective control between l_i and $E[x_i]$.
- A Probabilistic Random Mobility Model for generation of paths of any vehicle [29].
- We successfully introduced the Flexible SoU upon the area of Particle Filtering and generation of paths.
- The use of the CTR and status of a mobile node, in combination, to reduce and adapt the space of uncertainty.

Chapter 2

Background

2.1 Node localization

Hereafter, the term *beacon nodes* refers to those nodes for which their location is known and *unknown nodes* for those nodes for which their location is unknown. The next subsections describe several related NLAs according to the wide classifications: range-free, range-based, or hybrid NLAs, where all equations presented in this chapter belong to the state of the art.

2.2 Range-free node localization

Range-free NLAs are usually less accurate than range-based NLAs. However, they can obtain reasonable performance such as the CL [17]. This NLA has a low complexity and it is easy to implement. In the CL algorithm each unknown node i estimates its physical position P_i in a two-dimensional Cartesian space (x, y) by averaging the positions of n heard beacon nodes B_j. It is defined as shown in (2.1).

$$P_i(x, y) = \frac{1}{n} \sum_{j=1}^{n} B_j(x, y) \tag{2.1}$$

The experimental results from [17] show different error performance for outdoor and indoor scenarios. For the former, the error never exceeded 2 meters while for the latter varied widely from 4.6 meters to 22.3 meters. According to the authors the error fluctuation depends on: the amount of walls, the node location, and the node orientation. The CL algorithm was developed assuming perfect spherical radio propagation and an identical transmission range, unlike the Approximate Point-In-Triangulation algorithm

[16]. The APIT algorithm uses the DOI (*Degree Of Irregularity*) equal to 0.1 and 0.2. The DOI is defined as an indicator of the ratio pattern irregularity, e.g. if DOI=0.3, the radio range r in each direction takes a random value from $0.7r$ to $1.3r$. In the APIT process, an unknown node selects arbitrarily three heard beacon nodes in order to form a triangle (beacon triangle); if the unknown node is inside the beacon triangle, the beacon triangle can be considered a containing area. The process is executed for all heard beacon nodes or until a certain accuracy is achieved. The unknown node assumes the centroid of the intersection center of all containing areas as its position as shown in Figure 2.1. The density of beacon nodes plays an important role for this kind of approaches. This is also true for the DV-Hop [19] and the amorphous position algorithms [30].

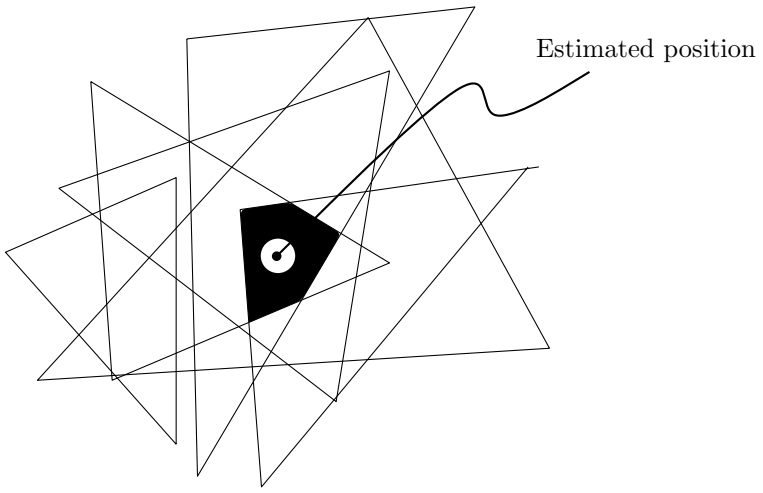


Figure 2.1: An example of APIT performance.

In Bounding Box the coverage range of each beacon node is considered as a box. The center of the intersection of all heard beacon nodes is the estimated position of the non-beacon node. The boxes' size are based on hop count radio range from beacon nodes to the non-beacon node as shown in Figure 2.2. The box of each node can be expressed with a lower position $B^i_{LOWER}(x, y)$ and the upper position $B^i_{UPPER}(x, y)$. The height is then $h = |B^i_{LOWER} - B^i_{LOWER}|/2$. The equations 2.2 and 2.3 shows the lower position P_{LOWER} and upper position P_{UPPER} of the parallelepiped formed by the intersection of the boxes.

$$P_{LOWER} max[B^i_{LOWER}(x), B^i_{LOWER}(y)] \tag{2.2}$$

$$P_{UPPER} max[B^i_{UPPER}(x), B^i_{UPPER}(y)] \tag{2.3}$$

The estimated position of the node is the average of coordinates lower and upper of the parallelepiped as is shown in equation 2.4

$$P_i(x,y) = (\sqrt{(P_{UPPER}(x) - P_{LOWER}(x))^2}, \sqrt{(P_{UPPER}(y) - P_{LOWER}(y)^2}) \quad (2.4)$$

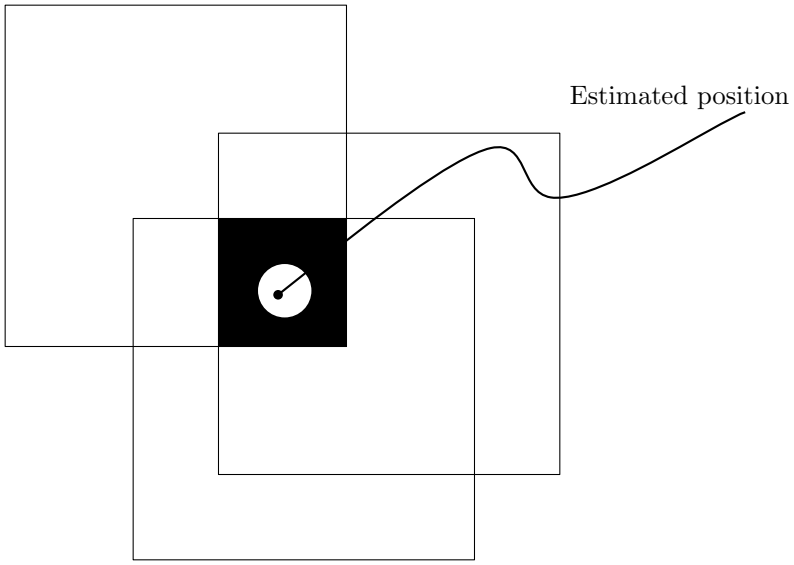


Figure 2.2: An example of DV-Hop performance.

2.3 Range-based node localization

Range-based NLAs usually perform better than range-free, however, in some cases additional hardware is employed. Another issue to consider is the calibration process and time that might be required before actual location estimation can be executed. For instance, the RADAR [23] is a radio-frequency based system for locating and tracking persons in indoor scenarios; it uses the signal strength from three beacon nodes inside a building. Based on empirical measurements the user location is inferred by triangulation. The authors consider the wall attenuation based on a floor attenuation factor propagation model. Nevertheless, RADAR requires an extensive effort and time for the generation of the signal database based on previous survey measurements. A similar work is presented in [31] where a wireless signal strength map is generated by using a ray-tracing approach in order to include absorption and reflection characteristics of various obstacles (home environment). The locations of users are computed using Bayesian Filtering on sample sets which are derived by Monte Carlo Sampling. Authors' results show a sub-room precision ($\sim 1m$). However, the training effort requires time because they have to measure

walls, windows, and door length of all rooms as well as identification of major obstacles.

SpotON from [15] is also a radio-frequency based NLA. It employs tagging technology and the radio signal strength for 3D location estimation. In this system beacon nodes are connected to a server using serial ports and Ethernet interfaces. The available interfaces are incremented if required by using a microwebserver. The SpotON algorithm estimates the distance between nodes by using the equation 2.5, where SS is the signal strength and d is the distance in meters. The values 0.629 and 4.71 were obtained in an empirical form:

$$SS = 0.236 \times d^2 - 0.629 \times d + 4.71 \tag{2.5}$$

The PinPtr algorithm [14] is a sniper-detecting system based on acoustic measurements of shots. In other words, this is founded in the fact that when a projectile is fired, it produces a spherical sound wave. The shot trajectory is taken into account and points within it are then selected. From these points a cone-shape is formed and the system then computes the angles among them. Based on the sound velocity, the computed angles and the trajectory, a shooter position is then estimated. This algorithm is limited by a line-of-sight requirement. Other algorithms limited for such factor include Time of Arrival based NLAs such as GPS and [32, 33]. However, the former is still not an option for indoor environments. In a different approach Angle of Arrival NLAs [19, 34, 18] detect the angle of arrival of messages by using microphones as shown in Figure 2.3. Each node must have one speaker and many microphones. Both cost and accuracy are the main issues of this approach.

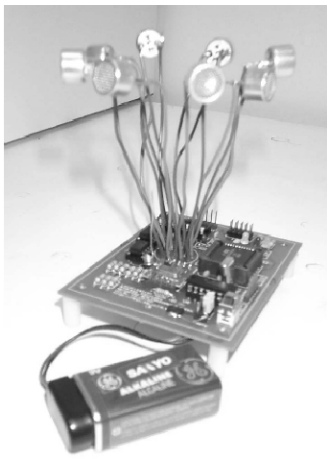


Figure 2.3: The experimental node Medusa.

Time Difference of Arrival techniques [35] are based on the propagation time dif-

ference among sound and electromagnetic signals. TDoA NLAs require for each node one speaker and microphone. The strategy for node localization is as follows: (a) the beacon node sends a broadcast message and it expects a fixed time F_t to produce a chirp with its speaker. (b) The unknown node receives the message at time T_m and the chirps at time T_c, where $T_c > T_m$. (c) The unknown node employs these three variables and the corresponding propagation speed of the sent message and the sound to calculate the distance between it and the beacon node. This is expressed in the equation 2.6, where the variable S_m is the message propagation speed and S_c is the chirp propagation speed.

$$distance = S_m - S_c \times T_c - T_m - F_t \tag{2.6}$$

The accuracy for both TDoA and AoA NLAs is limited by hardware and line-of-sight. On the other hand, the distance estimation can be also performed based on the Received Signal Strength Indicator [**Benkic:2008**, 36, 37]. In [36] the RSSI measurements are filtered using the standard deviation and considering packet loss limits. On the other hand, the study presented in [**Benkic:2008**] concludes that the RSSI is a "bad" factor for distance estimation. Multi-path fading, background, interference, and irregular signal propagation can explain the latter. Finally in Table 2.1 synopsis of some NLAs is presented.

Table 2.1: NLA's performance.

NLA	Accuracy	Nodes	Overload	Architecture	Environment	Range
AoA	High	> 10	Low	Distributed	Outdoors	Based
RSSI	Fair	> 10	Fair	Centralized	Indoors	Based
ToA	High	> 10	Low	Distributed	Outdoors	Based
TDoA	High	> 10	Low	Distributed	Outdoors	Based
Amorphous	Good	>16	Large	Distributed	Outdoors	Free
APIT	Good	>16	Low	Distributed	Outdoors	Free
Centroid	Fair	>10	Low	Distributed	Both	Free
WCL	Fair	>10	Low	Distributed	Both	Free
Bounding Box	Fair	>16	Low	Distributed	Both	Free
Dv-Hop	Good	>16	Large	Distributed	Outdoors	Free
MDS	Good	> 16	Large	Centralized	Outdoors	Both

2.4 Hybrid node localization

Hybrid NLAs use both the content of the received messages and distances/angles. Some of these algorithms includes Triangular Centroid Localization [21], the Weighted

Centroid Localization [20], ICL (*Improved Centroid Localization Algorithm*) [38], and EDIPS [39]. The TCL algorithm employs two beacon nodes and the estimated position of the unknown node to form a triangle. Two beacon nodes form the base and the height is given by the estimated position of the unknown node (vertex B) which is provided by the CL algorithm. On the order hand, the distance between the unknown and both beacon nodes is estimated by using the variations of the RSSI or the LQI (*Link Quality Indicator*). The vertex B is then displaced in a defined process until the triangle cathetus matches with the RSSI/LQI estimated distances. The distance estimation in WCL is also performed by using the RSSI/LQI. The unknown node position is estimated by the equations 2.7 and 2.8.

$$P_i(x,y) = \frac{\sum_{j=1}^{n} W_{ij} \times B_j(x,y)}{\sum_{j=1}^{n} W_{ij}} \tag{2.7}$$

$$W_{ij} = \frac{1}{d_{ij}^{g}} \tag{2.8}$$

Where $P_i(x,y)$ represents the position of the unknown node i and B_j the position of a heard beacon node. For each B_j a weight W_{ij} is assigned and calculated using the equation 2.8. This weight represents the wave attenuation factor. The distance d_{ij} is the distance between nodes i and B_j and it is estimated from the LQI variations. The variable g takes integer values ranging from one to five in order to determine the optimal value. In the experimental results from [20] the WCL the best performance under two settings: (i) $g = 1$ and the coverage range is 10 meters; and (ii) $g = 3$ and the coverage range is 30 meters. The distance estimation in the ICL algorithm [38] is also performed by using (2.7) and (2.8), but the variable g is fixed to 1. The ICL algorithm is based on APIT [16] and the quality of perpendicular bisector. Three perpendicular bisectors divide the beacon triangle formed by the APIT algorithm into six small cells. A unknown node assigns itself to a cell from by the RSSI from heard beacon nodes. Finally, the centroid of the assigned cell is assumed as the position of the unknown node.

The EDIPS [39] algorithm matches with both hybrid NLAs and context-aware applications because it uses the signal strength from reference points and reacts regarding the environment status. EDIPS is explained in greater detail in the next subsection.

2.5 Context awareness

The general mechanism for computing location contextual information can be broken into three steps:

1. Information collection. Collecting information on the user's physical, informational or emotional state.

2. Context extraction. This involves analyzing the information in relation to other information such as location, time, environment, social and mental state, etc.

3. Pattern recognition. This entails predicting patterns in the consumer's behavior, over time.

All the aspects by themselves require expertise and thus demand individual teams to collaborate and come up with an efficient solution. Issues that may surface in this regard are likely to be that of privacy and spam. The node localization is employed as an essential data on the context-aware computing. In other words, the node localization is a priori knowledge. The important aspects of a context are: where you are, who you are with, and what resources are nearby. In our approach, we use the context awareness in order to improve the node localization estimation made by a fair node localization algorithm for instance the Centroid Localization Algorithm. The context encompasses more than just the user's location, because other things of interest are also mobile and changing. Context includes lighting, noise level, network connectivity, communication costs, communication bandwidth, and even the social situation.

Applications that react regarding environment status are named context-aware applications such as in [40], if the user is in a bookstore, s/he receives offers and suggestions of books in his/her personal devices. Nowadays, this kind of applications have a wide area of use such as airports [40], education [41, 42], and hospitals [43, 44], just to appoint a few. Node localization is an essential input for context-aware applications. These applications assume the use of GPS or employ a high amount of beacon nodes with a short-range coverage. Therefore node localization is not considered an issue. The ABS (*Active Badge System*) [27] is a suitable example. The system provides a service for locating users within a building using badges with infra-red signal. In addition, ABS displays the likelihood of each user to stay in a certain place. The drawback of this system is based on the fact that the infra-red signal cannot travel through walls and badges have a coverage range of six meters. A similar application is EDIPS [39] which locates and tracks users indoors by using the signal strength from reference points. Once users are located, this system helps them to arrive with others users by displaying in his/her personal device the corresponding path. EDIPS can be divided in two parts: (i) a method for locating/tracking nodes and (ii) a service that helps users to find other users regarding their positions and the environment status.

Node localization plays an important input for the previous context-aware applications. However in scenarios where beacon nodes have a high-coverage range the node estimation can be compromised by attenuation, reflection, diffraction, and scattering issues which are discussed in greater detail in [9]. As far as we know, there is not a universal node localization algorithm that presents the same accuracy for all possible scenarios. We believe that a NLA should be able to react to the environment as a context-aware application in order to improve the node localization instead of including several factors from the environment.

2.6 Mobile node tracking

According to Yaakov [45], tracking is the processing of measurements obtained from target in order to maintain an estimate of its current state which typically consists of:

- Kinematic components (position, velocity, acceleration, etc.)
- Other components (radiated signal strength, spectral characteristics, "feature" information, etc.).
- Constant or slowly-varying parameters (couping coefficients, propagation velocity, etc.)

Measurements are noise-corrupted observations related to the state of a target, such as:

- Direct estimate of position.
- Range and/or azimuth (bearing) from a sensor.
- Time of arrival difference (obtained by cross-correlating data) between two sensors.
- Frequency of narrow-band signal emitted by target.
- Observed frequency difference (due to Doppler shift) between two sensors.
- Signal strength.

The measurements of interest in multi-target applications are usually not raw data points, but rather the outputs of signal processing and detection subsystems, as shown in Figure 2.4.

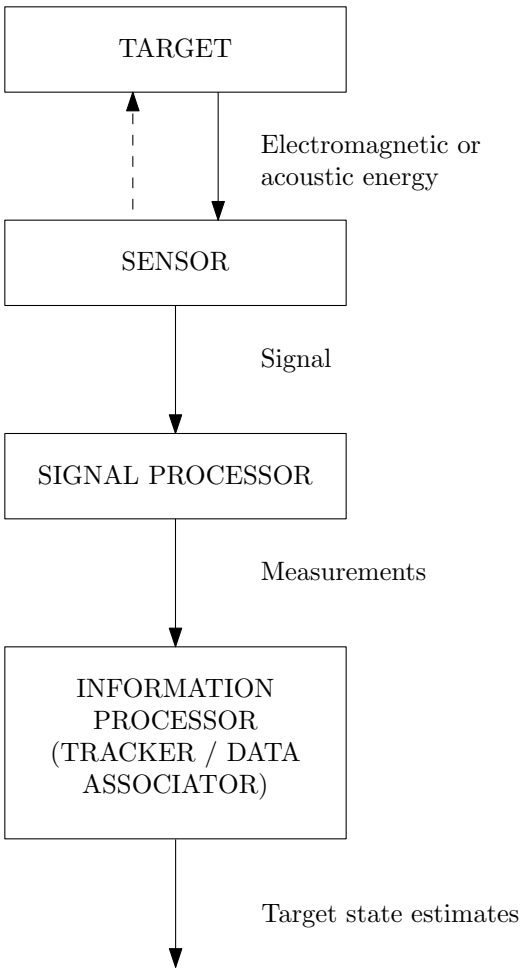


Figure 2.4: Components of a typical tracking system.

The track is a state trajectory estimated from a set of measurements that have been associated with the same target. The crucial point of the multi-target tracking problem is to carry out this data association (or data correlation) process for measurements whose origin is uncertain due to:

- Random false alarms in the detection process.
- Clutter due to spurious reflectors or radiators near the target of interest.
- Inferring targets.
- Decoys or other countermeasures.

The applications of tracking are numerous, ranging from undersea surveillance and space-age weapon systems to bubble-chamber experiments and image processing.

Some of the earliest examples involved radar and sonar systems, where manual tracking of blips on video displays by human operators evolved in to computer-controlled

tracking algorithms. Radar and sonar applications continue to abound. Military uses include land, air, sea, and space surveillance involving a large variety of sensors, targeting and control of individual weapons and weapon systems, and overall battle management. Civilian uses include air traffic control, collision avoidance, and navigation.

Sensors based on infrared, laser, and other technologies and new applications such as machine vision require novel approaches to data association and tracking. The sophistication and complexity of the algorithms grow to keep pace with the ever-expanding frontiers of computer technology.

In Particle Filtering approach external information is referred as observations and they can be captured for example by GPS, video cameras, odometers, beacon nodes, and so forth. Observations are the only information that helps to keep and weight particles, obviously the hardware quality plays an important roll, but also physical phenomena that affect the wave propagation and measurements. Information fusion [46] is commonly used as a solution to merge observations coming from several and diverse sources in order to improve the node localization estimation such as [47, 48, 49, 50, 51].

The objective of a particle filter is to estimate the posterior density of the state variables given the observation variables. The particle filter is designed for a hidden Markov Model, where the system consists of hidden and observable variables. The observable variables (observation process) is related to the hidden variables (state-process) by some functional form that is known. Similarly the dynamical system describing the evolution of the state variables is also known probabilistically.

The objective of the particle filter is to estimate the values of the the hidden states X, given the values of the observation process Y. The particle filter aims to estimate the sequence of hidden parameters, X_k for $k = 0, 1, 2, 3, ...$, based only on the observed data Y_k for $k = 0, 1, 2, 3, ...$ All Bayesian estimates of X_k follow from the posterior distribution $P(X_k|Y_0, Y_1, ..., Y_k)$.

A generic particle filter estimates the posterior distribution of the hidden states using the observation measurement process. Consider a state-space shown in the diagram Figure 2.5.

The general particle filtering approach includes three steps: (i) Initialization, (ii) Prediction, and (iii) Filtering. In (i), Particles are dropped from the proposal distribution. In (ii), the prior distribution is often used as importance function because it is easier drop particles and perform subsequent importance weight calculations. In (iii), filtering based on connection, exact range, bounded ranging and weighting is usually employed [31, 52, 53, 54, 55, 56]. SIR (*Sequential Importance Resampling*) is a very commonly used particle filtering algorithm which is shown in Algorithm 1.

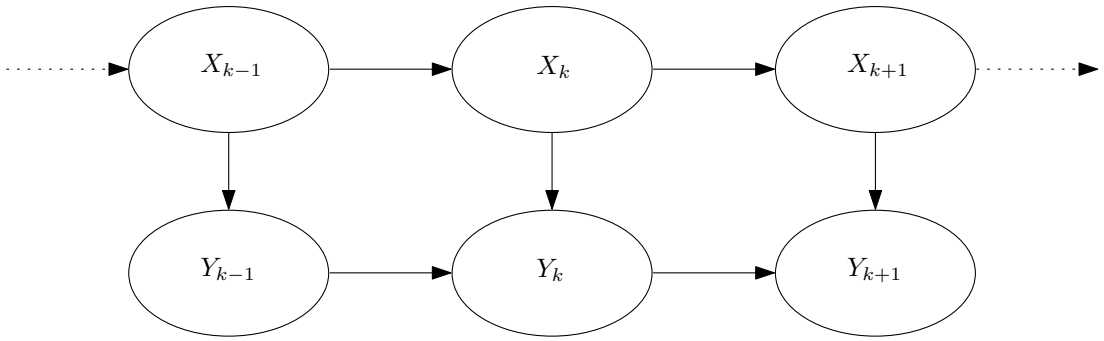


Figure 2.5: An example of the State-Space.

Algorithm 1 The generic Sequential Importance Sampling algorithm.

{At time 1}
for $i = 1$ to N **do**
  Sample $X_1^i \sim q(\cdot)$
  $W_1^i \propto \frac{\pi(X_1^i)}{q_1(X_1^i)}$
end for
Resample $\{X_1^i, W_1^i\}$ to obtain $\{\bar{X}_1^i, 1/N\}$
{At time $n \geq 1$}
for $i = 1$ to N **do**
  Set $N_{1:n-1}^i = X_{1:n-1}^{i^-}$
  Sample $X_n^i \sim q_n(\cdot\,|X_{1:n-1}^i)$
  Set $W_n^i \propto \frac{\pi_n(X_{1:n}^i)}{q_n(X_n^i|X_{1:n-1}^i)\pi_{n-1}(X_{1:n-1}^i)}$
end for
Resampling $\{X_{1:n}^i, W_n^i\}$ to obtain $\{\bar{X}_{1:n}^i, 1/N\}$

2.7 Conclusion

Wireless sensor networks are tremendously being used in different environments to perform various monitoring tasks such as search, rescue, disaster relief, target tracking and a number of tasks in smart environments. In many such tasks, node localization is inherently one of the system parameters. Node localization is required to report the origin of events, assist group querying of sensors, routing and to answer questions on the network coverage. One of the fundamental challenges in wireless sensor network is node localization. In this chapter a comprehensive review of the related work was shown. The importance localization and tracking of nodes was highlighted for indoors and outdoors. How the node localization in a WSN plays an important roll for both critical and non-critical applications. A common technical in target tracking in a WSN is that individual homogeneous sensors only measure their distances to the target whereas the state of the target composes of its position and velocity in the Cartesian coordinates. That is, the senor measurements are nonlinear in the target state. However, for those applications

where there is not a target or reference point an array of sensor might be required. The array of sensors have to be capable of collecting information valid during the displacement of the mobile node. Normally these measurements contain background noise and data processing before use is necessary. Particle Filtering approach seems to be appropriate to address the mobile node localization during shaded ares because it is able to handle the location uncertainty during shaded areas and improves the node location estimation over time.

PART I

LOCALIZATION

Chapter 3

The proposed Node Localization Algorithms

3.1 Introduction

There are diverse approaches for estimating the distance between nodes by using the RSSI. On this basis alone, various approaches have been developed to estimate the distance among nodes such as [**Benkic:2008**, 36, 37]. In [36] the authors improve the estimation by filtering the measured RSSI through optimized standard deviation and packet loss limits. On the other hand, in [**Benkic:2008**] is presented a study of the RSSI, where the authors conclude that the RSSI is bad distance estimator in buildings; as a matter of fact, the indoor node localization is a hard task for NLAs based on the RSSI. In general, these kinds of methods suffer problems of the multi-path fading, the background, the interference, and the irregular signal propagation. We have development our own method to estimate the distance among nodes. We used the TinyOS [57] to program the functionality of nodes and the network simulator AvroraZ [58] to simulate the WSN along with the obstacles.

3.2 Distance estimation process

The measurement point $MesurementPoint_n$ for each scenario is defined by the distance dis between two nodes MicaZ (see appendix B). The variable is a three-tuple such that $MesurementPoint_n = \{dis, \mu_0, \sigma\}$ where μ_0 is the hypothesis population mean and σ is the population standard deviation. The variable dis ranging from 1 to 18 because the maximum coverage range of each node is about 18 meters. A total of 100 running simulations were achieved for each different scenario, 18, by changing the obstacle level at random. The details for each scenario are as follow:

- A propagation model for indoors based on the Rayleigh and lognormal distribution.
- The Obstacle Level ρ ranging at random from 0.01 to 1, where ρ represents the number of people per square meter in the coverage range of each node.
- 242 packets received per node.
- 12.312 seconds per simulation.
- 1600 m^3 of volume

The Table 3.1 shows the resulting condensate of the 18 scenarios i.e., the 1800 simulations. The results are also presented in Figure 3.1 to make them clearer.

Table 3.1: The simulation results of the distance estimation process.

dis (m)	**RSSI**	**SDev**	**LQI**	**SDev**	**dis (m)**	**RSSI**	**SDev**	**LQI**	**SDev**
1	144.18	0.8	102.28	4.69	10	214.2	0.8	79.66	5.96
2	235.41	0.82	102.54	4.67	11	212.87	0.79	75.61	6.09
3	230.07	0.79	102.73	4.67	12	211.83	0.79	71.04	5.17
4	226.1	0.78	102.89	4.6	13	210.86	0.82	64.9	5.54
5	223.3	0.79	103.77	4.61	14	209.77	0.78	62.38	7.14
6	220.87	0.79	102.85	4.61	15	209.23	0.79	59.87	5.28
7	219.01	0.79	102.77	4.71	16	208.34	9.5	57.01	5.15
8	217.14	0.81	95.76	6.45	17	197.32	41.61	56.09	11.96
9	215.65	0.83	83.35	7.3	18	105.17	102.9	45.59	29.09

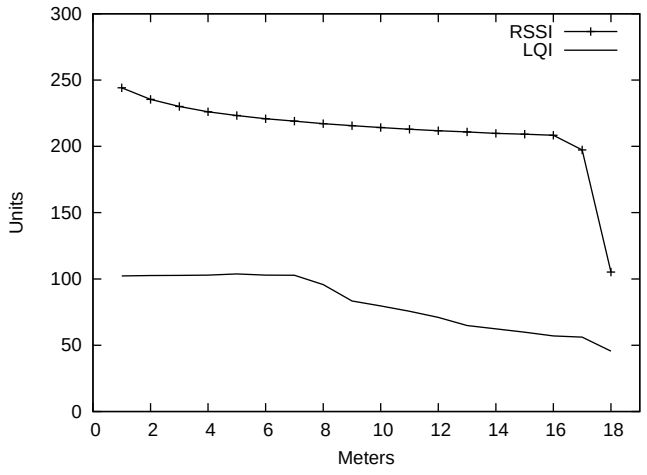


Figure 3.1: The distance estimation process: the simulation results.

In Figure 3.1,the RSSI distance has a correlation factor of -0.69809601, thus the data tend to be strongly related and LQI-distance has a correlation factor of -0.96185265.

That is, the data tend to be perfectly related.The values for RSSI and LQI are referred as "Units" since the values varies from one vendor to another.

The Table 3.2 shows the acceptance region and confidence level of the statistical value z of the RSSI and the LQI. The values were defined empirically through the captured data. In hypothesis testing, the test procedure partitions all the possible sample outcomes into two subsets (on the basis of whether the observed value of the test statistic is smaller than a threshold value or not). The subset that is considered to be consistent with the null hypothesis is called the acceptance region; another subset is called the rejection region. If the sample outcome falls into the acceptance region, then the null hypothesis is accepted. If the sample outcome falls into the rejection region, then the null hypothesis is rejected (i.e. the alternative hypothesis is accepted). Confidence level represents the portion of the data that is considered.

Table 3.2: The acceptance region and the confidence level for distance estimation process.

	RSSI	**LQI**
Acceptance region	$1.96 > z > -1.96$	$3.91 > z > -3.91$
Confidence level	95%	99.9%

Algorithm 2 with complexity $O(n)$ shows the technique for estimating the distance between a beacon node and a non-beacon node by using hypothesis testing. Where, sm is the sample mean, sn is the sample size, ra is the region of acceptance.

Algorithm 2 Distance Estimation. $O(n = 18)$

Require: sm, μ_0, σ, sn, ra
Ensure: $MesurementPoint_n\{dis\}$
1: **for** $MesurementPoint_n = 1$ **to** $MesurementPoint_n = 18$ **do**
2: $z \leftarrow \frac{sm-\mu_0}{\sigma\sqrt{sn}}$
3: **if** $MesurementPoint_n = 18$ **then**
4: **return** $MesurementPoint_n\{dis\}$
5: **else if** $ra > z > -ra$ **then**
6: **return** $MesurementPoint_n\{dis\}$
7: **end if**
8: **end for**

The complexity of Algorithm 2 could be reduced to O(log n), however space exploration is only 18 elements so that it is negligible O(log n) and we are interested in the smallest value dis that satisfied the condition of z in the Algorithm 2.

3.3 Vectorial Localization (VL)

The node VL (*Vectorial Localization*) algorithm performances better for 2D and 3D outdoor environments such as all range-based NLAs and it has order of n^2 time complexity. The Algorithm 3 shows the pseudo-code. The VL needs at least the identifiers of two heard beacon nodes B_i, the coordinates of their physical positions and the previous estimated position S of the non-beacon node. As first step, the VL algorithm creates one vector with origin in the heard beacon node B_i and destiny in the previous estimated position (Lines 1 and 2). As second step, the magnitude M of the vector is decreased (lines 3-8) or increased (lines 9-15) until it is close enough to the estimated distance ED (lines 4 and 10) calculated by Algorithm 2. Finally, the coordinates of the destiny of each vector are averaged for obtaining the new estimated position NE_i of non-beacon node (line 22).

Algorithm 3 The Vectorial Localization. $O(n^2)$.

Require: S, B_i, M, ED
Ensure: NE
1: $C_i \leftarrow \sqrt{{S_i}^2 + {B_i}^2}$
2: $M \leftarrow \sum \sqrt{{C_i}^2}$
3: **if** $M > ED$ **then**
4: **while** $M > ED$ **do**
5: **if** $S_i \neq 0$ **then**
6: $C_i \leftarrow |C_i - 0.001|$
7: **end if**
8: **end while**
9: **else**
10: **while** $M > ED$ **do**
11: **if** $S_i \neq 0$ **then**
12: $C_i \leftarrow |C_i + 0.001|$
13: **end if**
14: **end while**
15: **end if**
16: **if** $B_i < S_i$ **then**
17: $NE_i \leftarrow B_i + C_i$
18: **else**
19: $NE_i \leftarrow B_i - C_i$
20: **end if**
21: {Back to 1: for the next heard beacon node}
22: {Averaging each NE_i}

In Figures 3.2 and 3.3 a simple example of the VL algorithm is shown. The estimated distance ED is assumed as accurate in order to show the VL principle. The black dots represent beacon nodes B_i, the happy face denotes the real position RP of the non-beacon node, and the star represents the estimated position S by Centroid Localization

algorithm [17]. In the step 1, the vectors among the B_i and the S are formed. In step 2, each C_i is increased or decreased in 0.1 meters until the magnitude of the vector M is close enough to ED. In this example, the error is reduced from 1.41 meters to 0.5 meters i.e., 64.54%.

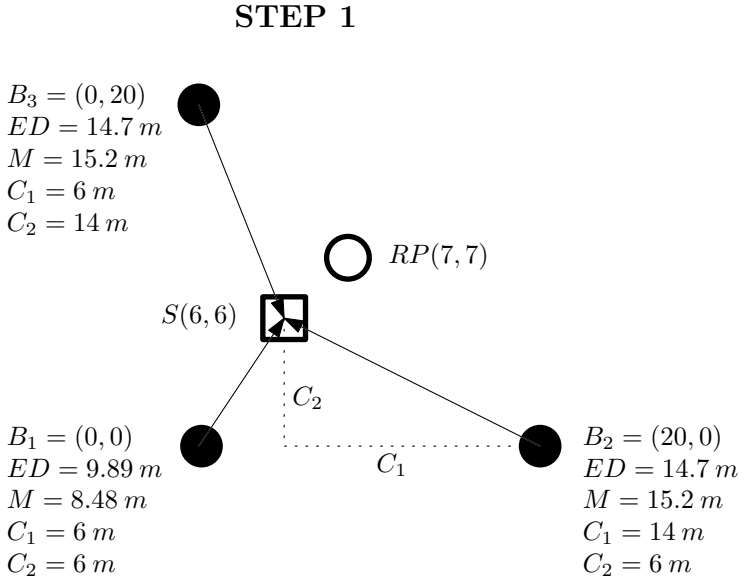


Figure 3.2: VL example. The vectors are created.

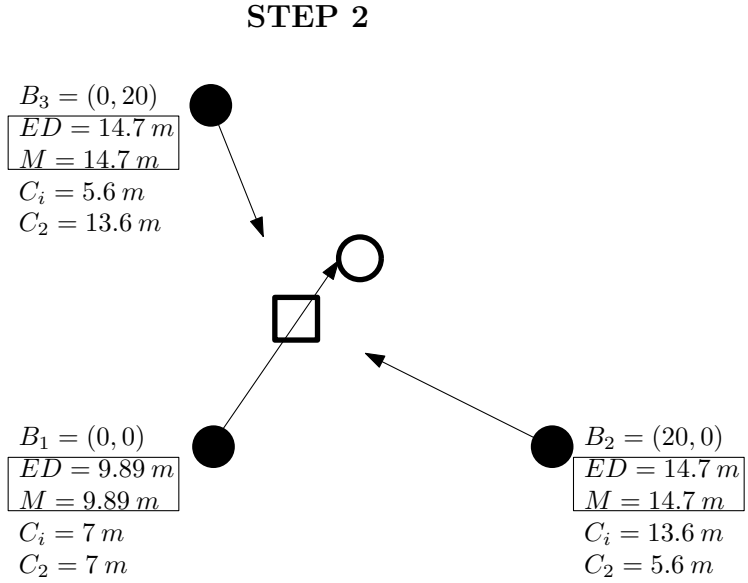


Figure 3.3: VL example. The vectors are adjusted.

Since the VL algorithm requires a previous estimated position of the non-beacon node as an input data; it only can be performed with an execution stamp higher than 1. The use and meaning of the stamp will cover in Chapter 4.

3.4 Energized Centroid Localization (ECL)

The terms *pattern* and *feature* are included in this subsection. The *patterns* are physical representations of objects such as signals, images or values' tables. The *features* are measured, or primitive attributes derived from measurements of the patterns and they can be useful for characterization.

The most commonly employed patterns in the range-based NLAs are: the receive signal strength, the link quality, the time of arrival of signals/sounds, and the packet traffic. As far as we know, there is not a single node localization algorithm that employs the energy consumption information of beacon nodes. Instead of, this pattern has been addressed to the energy saving. The ECL (*Energized Centroid Localization*) algorithm is able to determine the delimited area where the non-beacon node is throughout the energy consumption information of the heard beacon nodes. The ECL was developed as an algorithm to be performed with the execution stamp number 1. However, it can also be executed in isolated form by finding the centroid of the delimited area.

3.4.1 Scenario

We used the TinyOS [57] to program the functionality of nodes and the network simulator AvroraZ [58] to simulate the WSN along with the obstacles. The ECL algorithm was developed throughout the energy consumption analysis of the beacon nodes while they sent and received messages. The Figure 3.4 shows the scenario where the ECL was developed. In each intersection of the imaginary lines there is a beacon node positioned. The topology of the beacon nodes was established according to one employed in the Centroid Localization. The coverage range of all nodes (continuous lines) is approximately 18 meters because of the wave propagation irregularity. The crossing of the coverage ranges make possible the formation of thirteen areas which are identified with a letter (A-M).

In our simulations, a simple non-beacon node is located in each zone at random 100 times. A total of 1300 simulations have achieved. Each beacon node starts the transmission by sending 30 packets. The non-beacon node for each packet that has received sends an acknowledgement packet. The gathered data were: delay, jitter, inter-arrival time, corrupted packets, beacon node' ID (*IDentification number*), position of beacon nodes, number of sent packets, number of received packets, lost packets, the RSSI, the LQI, and the energy consumption information. The data has been treated according to the steps of data mining [59]:Feature selection, Pre-processing, Data transformation, and

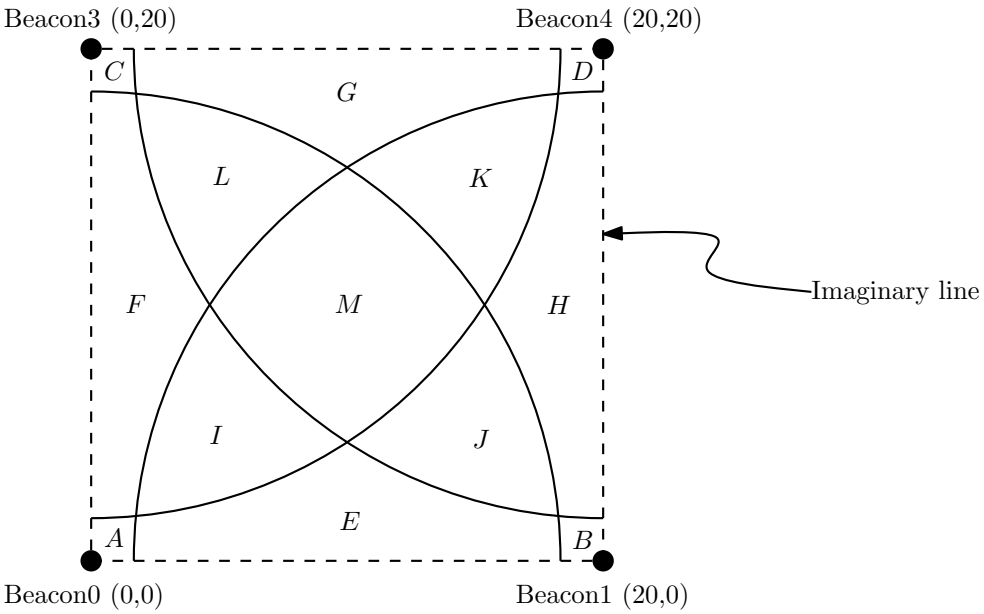


Figure 3.4: The experimental environment for ECL.

Model generation. We used for this propose Weka [60], TinyOS, and AvroraZ. See the appendix B to obtain greater detail.

3.4.2 Feature Selection

In this step is determined which is the most important set of features or significant to describe patterns and the irrelevant ones are cast aside, thus improving the classification and reduces downtime processing.

- The packets' delay.
- The packets' jitter.
- The packet's interarrival time.
- The corrupt packets rate per node beacon.
- The corrupt packets rate across the network.
- The estimated position of node-beacon (random data).
- The beacon node position.
- The packets sent per node.
- The packets sent cross the network.

Through this step the valid patterns are, as shown in Table 3.3:

- The packets sent/received per node.
- The lost packets of beacon nodes.
- The beacons' RSSI.
- The beacons' LQI.
- The beacon nodes energy consumption.

Table 3.3: Databases after feature selection.

Node ID	**Packets received**	**Packets lost**	**RSSI**	**LQI**	**Energy consumption**	**Area**
0	30	19	243.03	73	44.58	A
1	0	24	-	-	35.09	A
2	0	24	-	-	34.29	A
3	1	25	-	-	34.09	A
⋮	⋮	⋮	⋮	⋮	⋮	⋮
0	124	18	209.51	56.64	47.57	M
1	124	21	209.58	56	47.47	M
2	124	21	209.64	56.03	47.27	M
3	124	14	209.72	57.57.41	47.27	M

The energy consumption for each node was record through a energy monitory available in AvroraZ.

3.4.3 Pre-processing

In some cases the data is amended for correcting deficiencies caused by limitations of the sensors. Since we are working with simulator's results might is not feasible amended the data. In this step, we use the evaluator of feature CfsSubsetEval from Weka in order to figure out the best valid pattern. In Table 3.4 the results of CfsSubsetEval are shown.

The Table 3.4 clearly shows that packets received per node is the best pattern. However, this pattern has already been widely used in NLAs. On the order hand, the energy consumption of beacon nodes is the second best pattern. One of the objectives of this doctoral work is to proof that the energy consumption of nodes is a useful patter to be used in the node localization.

Table 3.4: The CfsSubsetEval's resuts.

	CfsSubsetEval	
Research method	**The best pattern**	**The second best pattern**
Best First	The packets received per node	The energy consumption
FCBF	N/A	N/A
Genetic Search	The packets received per node	The energy consumption
Greedy Stepwise	The packets received per node	The energy consumption
Linear Forward Selection	The packets received per node	The energy consumption
Race search	N/A	N/A
andom Search	The packets received per node	N/A
Scatter Search	The packets received per node	The energy consumption
Subset Size Forward	The packets received per node	The energy consumption

3.4.4 Data transformation

We discovered that each heard beacon node needs received in average 30 acknowledgement packets in order to determine the area where the non-beacon nodes is. The energy consumption of beacon nodes fluctuates between 43 to 58 joules when a non-beacon node is inside its coverage range. On the other hand, the consumption of a beacon node when there is no a non-beacon node inside its coverage range fluctuates between 27 and 40 joules. A threshold was established at 41.5 joules. Should a beacon node passes the threshold it is set to 1, otherwise 0. Table 3.5 shows the transformation of the energy consumption characteristics of each beacon node.

Table 3.5: The data transformation according to the threshold 41.5 joules.

The energy consumption of beacons nodes				
Beacon0	**Beacon1**	**Beacon3**	**Beacon4**	**Area**
0	0	0	1	D
0	0	1	0	C
0	0	1	1	G
0	1	0	0	B
0	1	0	1	H
0	1	1	1	K
1	0	0	0	A
1	0	1	0	F
1	0	1	1	L
1	1	0	0	E
1	1	0	1	J
1	1	1	0	I
1	1	1	1	M

3.4.5 ECL Model

A decision tree is characterized by their robustness respect to the data's noise [61]. A decision tree can be used for classification of discrete values (high, medium, low) and continuous values. The latter require to be discretized. By using the data from Table 3.5, the binary decision tree model is generated which is depicted in Figure 3.5 where δ represents the binary energy consumption of the beacon node n. At this moment, the decision tree model is restricted to our experimental environment. However, the same process can be realized for other environments.

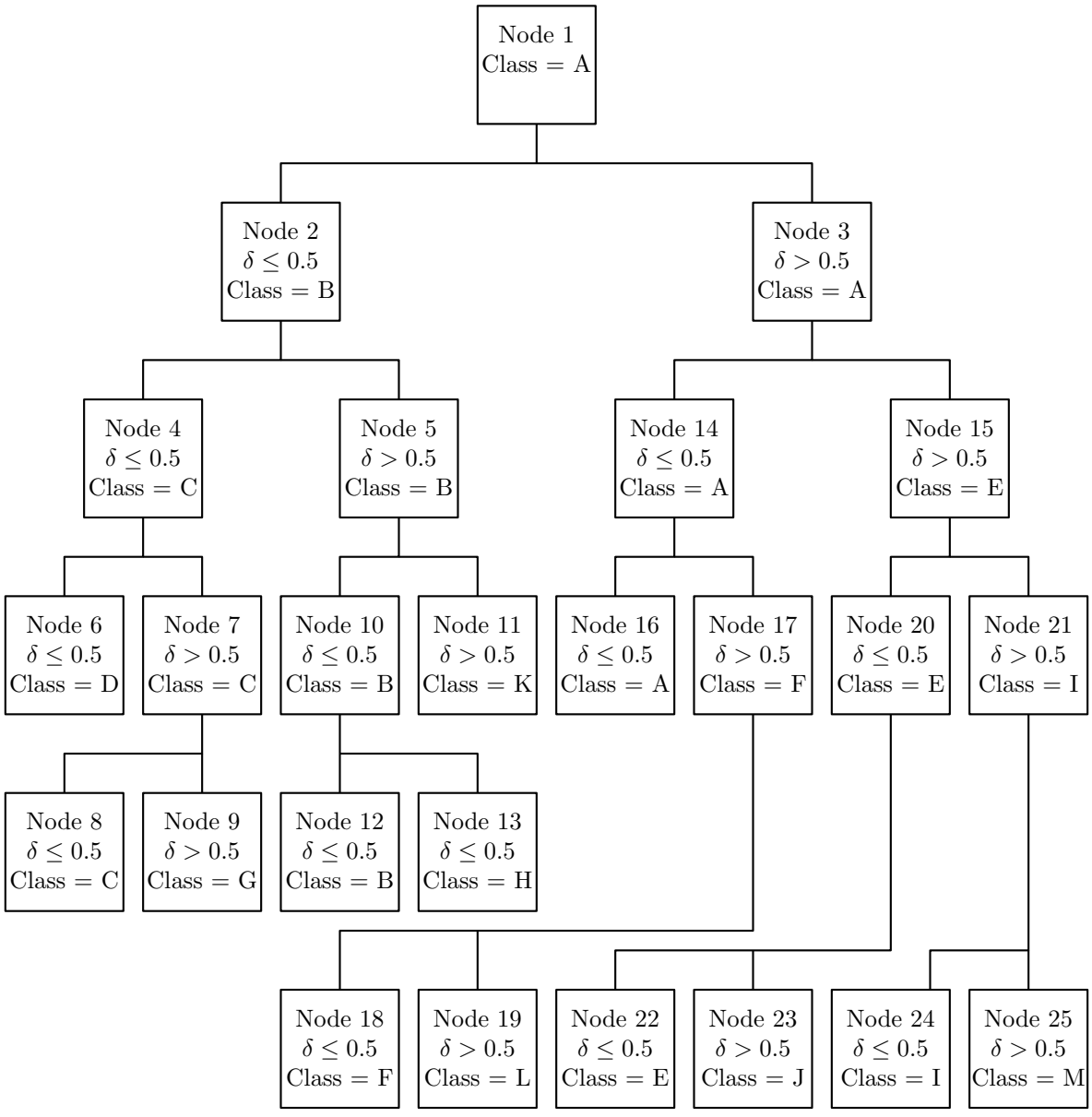


Figure 3.5: The consumption energy model. $O(log\, n)$.

The consumption energy model allow the non-beacon node identify the area (leaf)

where it is regarding the energy consumption of each beacon node. The estimated position of the non-beacon node is assumed as center of the delimited area when it is used in isolated form. But, the ECL was developed as an algorithm to be performed with the execution stamp number 1.

The only major drawback to this algorithm is that it can only locate one non-beacon node at a time and works only for four beacon nodes. However, the major advantage is that it has order of $O(log\, n)$ time complexity.

3.5 Subzone Localization (SzL)

In the previous section we explained how the non-beacon nodes are associated with one delimited area regarding the energy consumption of beacon nodes. However, delimited areas can be still dived into subzones in order to obtain better node localization accuracy. With the collected data in ECL (1300 simulations), the SzL (*Subzone Localization*) is able to determine the subzone regarding the RSSI/LQI measurements. Since this algorithm requires a previous estimated of the non-beacon node as an input data; it only can be performed with an execution stamp higher than 1.

3.5.1 Scenario

We used the TinyOS [57] to program the functionality of nodes and the network simulator AvroraZ [58] to simulate the WSN along with the obstacles. The SzL was developed and tested in the environment shown in Figure 3.5.1 where all the areas are dived into subzones such as is shown in the area K.

The general steps of the SzL are:

1. The SzL receives the delimited area from ECL.
2. The SzL receives the ID of the beacon nodes than cross the threshold.
3. The function θ, δ, λ, Ω is used according to the beacon nodes that crossed the threshold. These functions will provide the estimated position of node.

3.5.2 Function θ

When a one beacon node crosses the threshold the function θ is employed which is shown in the Algorithm 4. The variable Lrepresents the minimum Euclidean distance among beacon nodes. Where, $L = \sqrt{(B_i - B_j)^2}$, B_i and B_j are beacon nodes, r represents

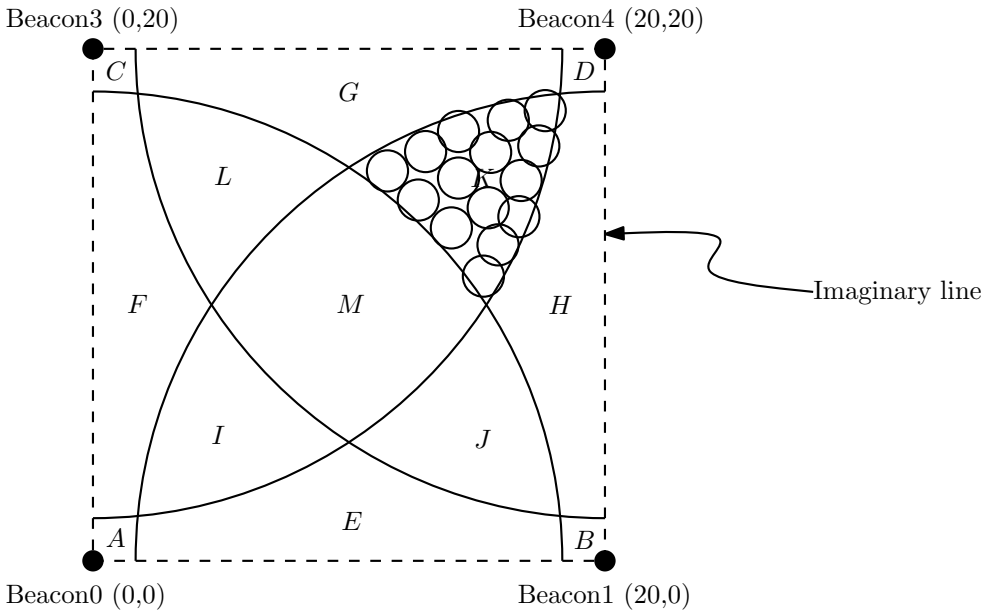


Figure 3.6: The experimental environment for the SzL.

the coverage range of the nodes, and *zone* is the detected zone, $\forall i, j$ s.t. $i \neq j$. Function θ has order of $O(n)$ time complexity such that n represents all heard beacon nodes.

Algorithm 4 Localization procedure of SzL for one heard beacon node. $O(n)$.

Require: *zone*, L, r
Ensure: (x, y)
1: **if** $zone = A$ **then**
2: **return** $(\frac{r-L}{2}, \frac{r-L}{2})$
3: **else if** $zone = B$ **then**
4: **return** $(L - \frac{r-L}{2}, \frac{r-L}{2})$
5: **else if** $zone = C$ **then**
6: **return** $(\frac{r-L}{2}, L - \frac{r-L}{2})$
7: **else if** $zone = D$ **then**
8: **return** $(L - \frac{r-L}{2}, L - \frac{r-L}{2})$
9: **end if**

3.5.3 Function δ

When two beacon nodes cross the threshold the function δ is employed which is shown in the Algorithm 5. Where, $B_i(x, y)$ is a heard beacon node or a beacon node tat crosses the threshold, ω represents the distance between B_i and a non-beacon node, (μ, ν) represents the previous estimated position in a two-dimensional Cartesian space. Function δ has order of $O(n)$ time complexity such that n represents all heard beacon nodes.

Algorithm 5 Localization procedure of SzL for two heard beacon node. $O(n)$.

Require: *zone*
Ensure: (x, y)
if $zone = E$ **then**
  $\alpha_1 \leftarrow 1, \alpha_2 \leftarrow \rho_1 \leftarrow \rho_2 \leftarrow -1$
  $\mu_1 \leftarrow B_0(x), \mu_2 \leftarrow B_1(x), \nu_1 \leftarrow B_0(y), \nu_2 \leftarrow B_1(y)$
else if $zone = F$ **then**
  $\alpha_1 \leftarrow 1, \alpha_2 \leftarrow \rho_1 \leftarrow \rho_2 \leftarrow -1$
  $\mu_1 \leftarrow B_0(y), \mu_2 \leftarrow B_3(y), \nu_1 \leftarrow B_0(x), \nu_2 \leftarrow B_3(x)$
else if $zone = G$ **then**
  $\alpha_1 \leftarrow \rho_1 \leftarrow \rho_2 \leftarrow 1, \alpha_2 \leftarrow -1$
  $\mu_1 \leftarrow B_3(x), \mu_2 \leftarrow B_4(x), \nu_1 \leftarrow B_3(y), \nu_2 \leftarrow B_4(y)$
else if $zone = H$ **then**
  $\alpha_1 \leftarrow \rho_1 \leftarrow \rho_2 \leftarrow 1, \alpha_2 \leftarrow -1$
  $\mu_1 \leftarrow B_1(y), \mu_2 \leftarrow B_4(y), \nu_1 \leftarrow B_1(x), \nu_2 \leftarrow B_4(x)$
end if
$\psi_i \leftarrow \mu - [\alpha|(\frac{L}{2} - \omega)|]$
$\pi \leftarrow \nu + \rho\{|[\frac{L}{2} - \omega][\frac{L}{(\frac{L}{2})^2}]| + [\frac{L}{(\frac{L}{2})^2}]\}$
return $\frac{1}{2}\sum_{i=1}^{2}(\psi_i, \pi_i)$

3.5.4 Function λ

The Equation 3.1 is employed when three beacon nodes cross the threshold i.e., the function λ used for the areas I, J, K and L.

$$\lambda(\omega, \mu, \nu, \rho, \alpha, \psi, \pi) \tag{3.1}$$

The non-beacon position is estimated through Table 3.6 which uses Table 3.7 in order to obtain the corresponding values of ψ and π. Where the variable ω is the estimated distance between a beacon node and a non-beacon node. The variables μ and ν are the coordinates of a plane. The variable ρ can take the positive or negative value (+1 or -1). The variables ψ and π are the result of the variables μ and ν respectively i.e., the estimated position of the non-beacon node. The value ψ of and π are established according to the Table 3.7 and used in the function λ

3.5.5 Function Ω

The Equation 3.2 is employed when the four beacon nodes cross the threshold.

$$\Omega(\omega, \mu, \nu, \rho, \alpha, \psi, \pi) \tag{3.2}$$

Where the variable ω is the estimated distance between a beacon node and a non-

Table 3.6: The estimated position by using the function λ

Area	Equation λ	Estimated position (NE)
I	$\lambda(B_0, K_x, K_y, -\rho, -\alpha, \psi_0, \pi_0)$	
	$\lambda(B_1, K_x, K_y, \rho, -\alpha, \psi_1, \pi_1)$	
	$\lambda(B_3, K_y, K_x, \rho, -\alpha, \psi_2, \pi_2)$	
J	$\lambda(B_1, J_x, J_y, \rho, -\alpha, \psi_0, \pi_0)$	
	$\lambda(B_4, J_y, J_x, \rho, -\alpha, \psi_1, \pi_1)$	
	$\lambda(B_0, J_x, J_y, \rho, \alpha, \psi_2, \pi_2)$	
K	$\lambda(B_4, K_x, K_y, \rho, \alpha, \psi_0, \pi_0)$	$NE \leftarrow (\psi_i, \pi_i) \leftarrow \frac{1}{3}\sum_{i=1}^{3}(\frac{(\psi_i, \pi_I)}{3})$
	$\lambda(B_1, K_y, K_x, \rho, \alpha, \psi_1, \pi_1)$	
	$\lambda(B_3, K_x, K_y, \rho, \alpha, \psi_2, \pi_2)$	
L	$\lambda(B_3, L_x, L_y, -\rho, \alpha, \psi_0, \pi_0)$	
	$\lambda(B_4, L_x, L_y, \rho, -\alpha, \psi_1, \pi_1)$	
	$\lambda(B_0, L_y, L_x, \rho, \alpha, \psi_2, \pi_2)$	

Table 3.7: The data transformation according to the threshold 41.5 joules.

IF	**THEN**	
ω	ψ	π
$\omega = 1,\ 2\ or\ 3$	$\psi \leftarrow \mu + (\rho \times 3)$	$\pi \leftarrow \nu + (\alpha \times 3)$
$\omega = 4$	$\psi \leftarrow \mu + (\rho \times 2)$	$\pi \leftarrow \nu + (\alpha \times 2)$
$\omega = 5\ or\ 6$	$\psi \leftarrow \mu + (\rho \times 1)$	$\pi \leftarrow \nu + \alpha$
$\omega = 7\ or\ 16$	$\psi \leftarrow \mu$	$\pi \leftarrow \nu$
$\omega = 8\ or\ 9$	$\psi \leftarrow \mu - \rho$	$\pi \leftarrow \nu - \alpha$
$\omega = 10\ or\ 11$	$\psi \leftarrow \mu - (\rho \times 2)$	$\pi \leftarrow \nu - (\alpha \times 2)$
$\omega = 12$	$\psi \leftarrow \mu - (\alpha \times 4)$	$\pi \leftarrow \nu$
$\omega = 13$	$\psi \leftarrow \mu - (\alpha \times 3)$	$\pi \leftarrow \nu$
$\omega = 14$	$\psi \leftarrow \mu - (\alpha \times 2)$	$\pi \leftarrow \nu$
$\omega = 15$	$\psi \leftarrow \mu - \alpha$	$\pi \leftarrow \nu$
$\omega = 17$	$\psi \leftarrow \mu + \alpha$	$\pi \leftarrow \nu$
$\omega = 18$	$\psi \leftarrow \mu + (\alpha \times 2)$	$\pi \leftarrow \nu$

beacon node. The variables μ and ν are the coordinates of a plane. The variable ρ can take the positive or negative value (+1 or -1). The variables ψ and π are the result of the variables μ and ν respectively i.e., the estimated position of the non-beacon node. The value of ψ and π are established according to the Table 3.8 and 3.9 and used in the function.

3.6 Triangular Centroid Localization (TCL)

We used the TinyOS [57] to program the functionality of nodes and the network simulator AvroraZ [58] to simulate the WSN along with the obstacles. The Triangular

Table 3.8: Value assignation of ψ and π based on ω

IF	**THEN**	
ω	ψ	π
$0 < \omega < 12$	$\psi \leftarrow \mu + (\rho \times 2)$	$\pi \leftarrow \nu + (\alpha \times 2)$
$\omega = 12$ *or* 13	$\psi \leftarrow \mu + \rho$	$\pi \leftarrow \nu + \alpha$
$\omega = 15$ *or* 16	$\psi \leftarrow \mu - \rho$	$\pi \leftarrow \nu - \alpha$
$0 < \omega < 17$ *or* 18	$\psi \leftarrow \mu - (\rho \times 2)$	$\pi \leftarrow \nu - (\alpha \times 2)$

Table 3.9: The estimated position by using the function ω

Area	**Equation** λ	**Estimated position (NE)**
K	$\lambda(B_0, M_x, M_y, -\rho, -\alpha, \psi_0, \pi_0)$	$NE \leftarrow (\psi_i, \pi_i) \leftarrow \frac{1}{4}\sum_{i=1}^{3}(\frac{(\psi_i, \pi_I)}{4})$
	$\lambda(B_1, M_x, M_y, \rho, -\alpha, \psi_1, \pi_1)$	
	$\lambda(B_3, M_x, M_y, -\rho, \alpha, \psi_2, \pi_2)$	
	$\lambda(B_4, M_x, M_y, \rho, \alpha, \psi_3, \pi_3)$	

Centroid Localization is fully distributed; it does not require any special hardware or synchronization time. For estimating non-beacon nodes position as first step, each beacon nodes transmits 30 packets every 2 ms in a broadcasting mode. As second step, Figure 3.7 shows how each a non-beacon node uses the received information. A non-beacon node starts in Idle State, it awaits for receiving a packet of any beacon node in order to pass to the Collection State. The non-beacon node should receive 30 packets of each heard beacon node, but some packets may not be delivered because channel obstruction. When a non-beacon node receives a packet at time 0.0 ms, it hopes for receiving the last packet from the same beacon node at time 60 ms approximately because of transmission time. When the wait time is over or all samples are larger than 29 packets, the non-beacon node changes its current state to Calculation State, where the estimated position process is realized. The non-beacon node transmits its estimated position remains in Notification State until it receives a delivery confirmation. Finally, it returns to an Idle State. For mobile non-beacon node, Notification State is connected to Collection State, which is now the initial state.

The Algorithm 6 details the Calculation State showing the pseudo-code of TCL for 2D environment. But, it can be extended to 3D environments easily. Where, V_n is a heard beacon node, $A_{x,y}$ is the triangle vertex A at the coordinate (x,y), $B_{x,y}$ is the triangle vertex B at the coordinate (x,y), $C_{x,y}$ is the triangle vertex C at the coordinate (x,y), a, b, c are the sides of the triangle (A,B,C), a_2 is the estimated triangle side (C,B) by Algorithm 2, and c_2 is the estimated triangle side (A,C) by Algorithm 2.

In Step 1, the triangle vertexes are set up. In steps 2-4, sides a, b and c are

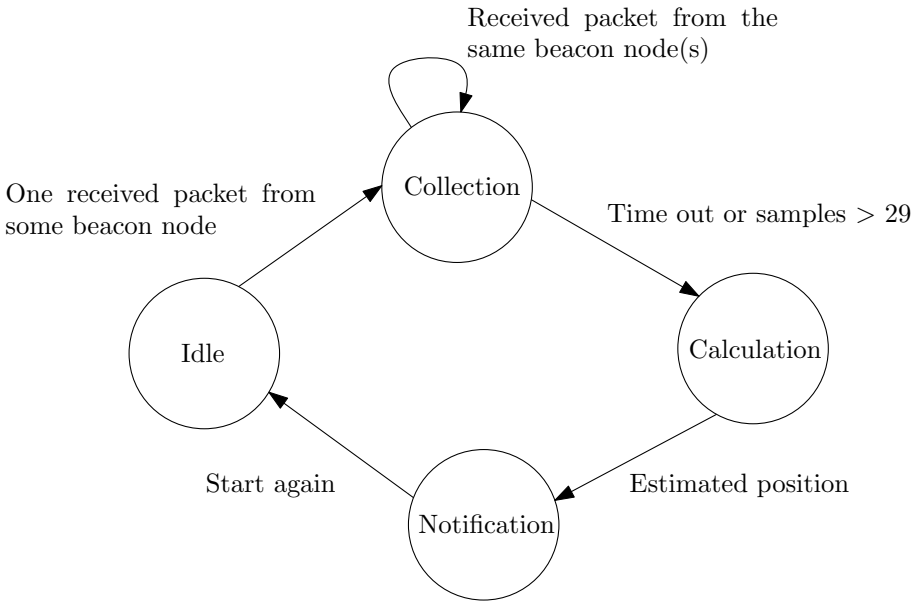


Figure 3.7: The state diagram for non-beacon nodes.

obtained. In step 5 the triangle is verified whether it can be formed correctly. We are going to suppose $|a_2 - a| > |c - c_2|$ (Step 20). In steps 22-25, the vertex B is displaced in straight line to vertex C until the side a is close enough with a_2. Steps 27-28 get the correct value of side c without modify the side a. These last steps, simulate the movement of a compas with origin in the vertex A and destiny in B until a and c match with a_2 and c_2 respectively. The partial estimated position is given for coordinates of vertex B in steps 17 or 28. The final estimated position is the average of all partial estimated position. We can consider the movement of vertex B in a sequential exploration way until a and c are close enough to a_2 and c_2, respectively. But it might require many time for large and dense wireless sensor network. Instead of greedy methods, the TCL algorithm uses simple rules bearing in mind the hot and cold game. If the vertex B is displaced in some direction and increases $|a - a_2|$, then B is moved in the opposite direction through the equation of the line or the movement of the compas. Thus, the use of angles is eliminated. For 3D localization TCL works with a prism, vertexes are A, B, C and D; for adjusting the side a and the side c is similar in Algorithm 6; for obtaining the side d the TCL simulates again the movement of a compas without changing the value of a and c. TCL algorithm has order of $O(n^2)$ time complexity.

Figure 3.8 depicts a simple example for estimating the non-beacon node localization in 2D environment. In order to show the behavior of our method we assume estimated distances by Algorithm 2 as accurate. In step 1, three involved beacon nodes are showed which are represented by white circles. The non-beacon node has a Real Position (RP)

Algorithm 6 Localization procedure of the TCL for 2D. $O(n^2)$.

Require: $V_n, A_{x,y}, B_{x,y}, C_{x,y}$
Ensure: (x, y)
1: $A \leftarrow V_1, C \leftarrow V_2, B \leftarrow \sum_{j=1}^{n}(V_j(x, y))$
2: $a \leftarrow \sqrt{(C_x - B_x)^2 + (C_y - B_y)^2}$
3: $b \leftarrow \sqrt{(A_x - C_x)^2 + (A_y - C_y)^2}$
4: $c \leftarrow \sqrt{(A_x - B_x)^2 + (A_y - B_y)^2}$
5: **if** $b_2 < a_2 + c_2$ **then**
6: **while** $b_2 < a_2$ **do**
7: $a_2 \leftarrow a_2 + 0.01$
8: $c_2 \leftarrow c_2 + 0.01$
9: **if** $|a_2 - a| < |c_2 - c|$ **then**
10: $x \leftarrow B_x$
11: **while** $|c_2 - c| > 0$ **do**
12: {increase or decrease x appropriately}
13: $y \leftarrow \frac{B_y - A_y}{B_x - A_x}(x - A_x) + A_y$
14: **end while**
15: $B_{x,y} \leftarrow (x, y)$
16: **while** $|a_2 - a| > 0$ **do**
17: {increase or decrease $B_{x,y}$ without modify c}
18: **end while**
19: **end if**
20: **if** $|a_2 - a| > |c_2 - c|$ **then**
21: $x \leftarrow B_x$
22: **while** $|a_2 - a| > 0$ **do**
23: {increase or decrease x appropriately}
24: $y \leftarrow \frac{B_y - C_y}{B_x - C_x}(x - C_x) + A_y$
25: **end while**
26: $B_{x,y} \leftarrow (x, y)$
27: **while** $|c_2 - c| > 0$ **do**
28: {increase or decrease $B_{x,y}$ without modify a}
29: **end while**
30: **end if**
31: **end while**
32: **end if**
33: {Return to 1: and establish C as the next element V_n}

in represented by the black box and the Estimated Position (EP) by Centroid [20] and it is represented by the black circle. In step 2, the vertexes A, B and C are assigned (the first triangle is formed). The sides x, b and c are calculated; a_2 and c_2 represent the estimated distance from the RP to vertex A and from RP to vertex C respectively. Since $|a_2 - a| < |c_2 - c|$, the vertex B is displaced in straight line to the vertex A until $c = c_2$. In step 4, the vertex B is adjusted for getting $a = a_2$ without changing c. The

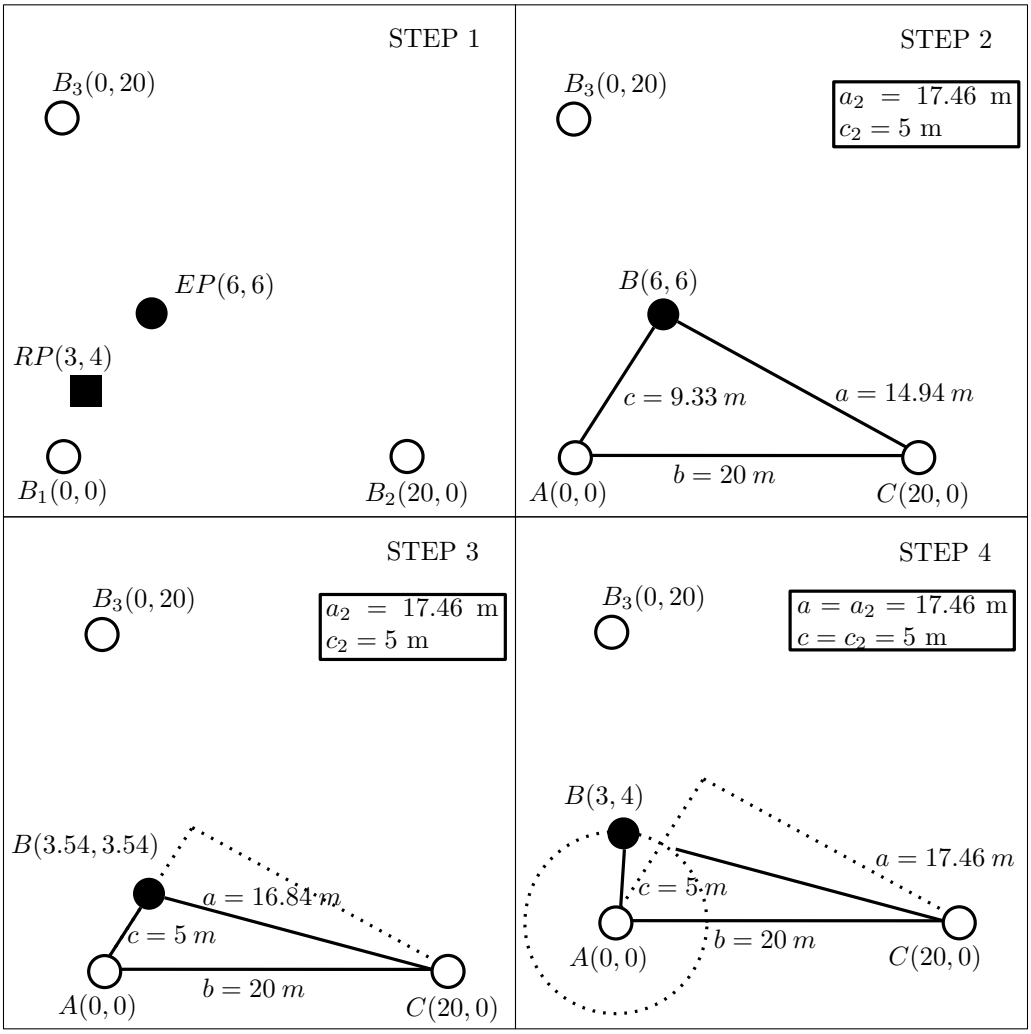


Figure 3.8: TCL steps, simulating the compas movements and the hot-cold game.

vertex B has finalized in the coordinate (3, 4). Since the distances are assumed accurate, the process is finished. However, in the process' implementation it is repeated for next couple of beacons nodes $(B1 - B3)$ and $(B2 - B3)$ as indicated in Algorithm 6.

3.7 Smart Beacon Nodes (SBN)

We used the TinyOS [57] to program the functionality of nodes and the network simulator AvroraZ [58] to simulate the WSN along with the obstacles. In this section the symbol ρ is used for representing the obstacles in order to maintain the symbol originally assigned by author of AvroraZ. The variable ρ represents the number of people per square meter in the coverage range of each node. For instance, if $\rho = 0.05$ there are 0.05 people per square meter, that is one person per every 20 square meters. Authors made a good approximation of the wave behavior of the chip CC2420 under diverse environment

conditions. The indoors model was developed considering the nodes on ceiling, walls or sometimes even on desks. The model uses Rice distribution when ρ is equal to zero and the combination of Log-normal and Rayleigh distribution for other cases.

For establishing OLI (*Obstacle Level Indicator*) values, first we have simulated 2 nodes micaZ at $base_D$ meters of separation between them, where $base_D$ takes positive integer values from 1 to 18. For each $base_D$, the variable ρ has been changed incrementally in 0.01 from 0.0 to 0.99. For each value of ρ, 10 simulations have been done; a database was created with a total of 19,000 achieved simulations. From the generated database, a simple UD (*Unknown Distribution*) has been created for each 10 increments of ρ. This process was done for each $base_D$. It can be expressed by $OLI_{coarse} = UD_{base_D,\rho_N}$. For instance, $OLI_{coarse} = UD_{base_8,\rho_5}$, that is the UD of 8 meters with $\rho = [0.5, 0.59]$. We also have created an UD of each simple ρ of each $base_D$. For example, $OLI_{fine} = UD_{base_6,\rho_{77}}$, that means, the UD of 6 meters with $\rho = 0.77$. In this form, we can identify a group of 10 incremental ρ as OLI_{coarse} and a simple ρ as OLI_{fine}.

In result section, we show that by using a simple ρ, the node localization error can be decreased up to 18% and making a cluster of simple ρ (OLI_{coarse}) up to 15%. We cannot consider only the OLI in the intersection of the coverage ranges of nodes, because the network simulator takes the major ρ between them.

The Figure 3.9 depicts the behavior of the recorded RSSI of two nodes micaZ at 5 meters and 10 meters of separation between them with $\rho = [0.0, 0.99]$. The value of the RSSI at 5 meters with $\rho = 0.92$ is almost the same value of the RSSI at 10 meters with $\rho =$ 0.0. Under this condition, some localization algorithm can estimate the distance between a pair of nodes of 5 meters as 10 meters. These kinds of situations are counterattacked by the transition of OLI.

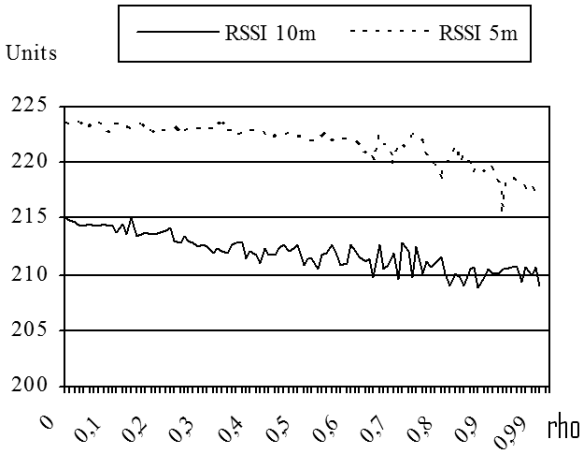


Figure 3.9: RSSI for $base_D = 5$ and $base_D = 10$ with ρ from 0.0 to 0.99.

The SBNs (*Smart Beacon Nodes*) apply the HT (*Hypothesis Testing*) for selecting the better ρ for each couple of beacon nodes. The Table 3.10 shows the acceptance region and confidence level of the statistical value z regarding the RSSI/LQI for all UD. The values were defined empirically from the database generated by recorded RSSI/LQI. The Algorithm 7 shows the technique for estimating OLI among SBNs by using HT and it has order of $O(n)$ time complexity.

Table 3.10: The SBNs. The acceptance region and the confidence level.

	RSSI	**LQI**
Acceptance region	$1.96 > z > -1.96$	$3.91 > z > -3.91$
Confidence level	95%	99.9%

Where $SBN_{k,l}$ is a couple of SBNs with connection, μ is the sample mean of $SBN_{k,l}$, μ_0 is the hypothesized population mean of UD_{base_D,ρ_N}, σ is the population standard deviation of UD_{based_D,ρ_N}, n is the sample size, and a is the acceptance region.

Algorithm 7 Localization procedure for OLI estimation. $O(n)$.

Require: $SBN_{k,l}$ μ, μ_0, σ, n, a
Ensure: UD_{base_D,ρ_N}

1: $base_D \leftarrow \sqrt{[SBN_k(x,y,z) - SBN_l(x,y,z)]^2}$
2: **for all** ρ of UD_{base_D,ρ_N} **do**
3: $\quad z \leftarrow \frac{\mu-\mu_0}{\sigma\sqrt{n}}$
4: $\quad$ **if** $-a \leq z \leq a$ **then**
5: $\quad\quad$ **return** UD_{based_D,ρ_N}
6: $\quad$ **end if**
7: **end for**
8: Return to step 2 and estimate for the next SBN_k

In step 1, the separation between SBN_k and SBN_l is calculated by using their physical position information. In steps 2 and 3, the statistical value z is obtaining respect to each UD_{base_D,ρ_N} from the fixed $base_D$ obtained in step 1. In step 4, if the value of z is greater/equal than -1.96 and lesser/equal than 1.96, in step 5 OLI is equal to the UD_{base_D,ρ_N} with the fixed $base_D$ and the estimated ρ. The process continues for all SBN_{kl} . Once estimated OLI, each SBN transmits a five-tuple (UID, X, Y, Z, OLI), where UID is the unique identifier of the SBN and its physical position at coordinates (X, Y, Z). From this information, an approximation of the RSSI behavior respect to obstacles in each part of the wireless network can be made. For instance, let be $\rho = 0.5$ between the SBN_3 and the SBN_5, such that $SBN_3(8, 4, 3)$ and $SBN_5(9, 1, 8)$. It can be expressed as $SBN_3 - SBN_5 = UD_{base_6,\rho_5}$. The same way for the rest of SBN_{kl} in the WSN.

A non-beacon node can estimate better the distance among it and the heard SBNs with the received OLI from them. Algorithm 8 shows the process. Where SBN_k is a heard SBN by N_i, μ is the sample mean from SBN_k, μ_0 is the hypothesized population mean of UD_{base_D,ρ_N}, σ is the population standard deviation of UD_{based_D,ρ_N}, n is the sample size, a is the acceptance region, and $N_i - SBN_k$ is the estimated distance between $N_i$and SBN_k

Algorithm 8 Estimating the distance among a non-beacon and head SBNs. $O(n)$.

Require: SBN_k μ, μ_0, σ, n, a
Ensure: $N_i - SBN_k$
1: $\rho_N \leftarrow \frac{1}{n}\sum_{j=1}^{n} \rho_N, \forall\, SBN_k$
2: **for all** $base_D$ of UD_{base_D,ρ_N} **do**
3: $z \leftarrow \frac{\mu-\mu_0}{\sigma\sqrt{n}}$
4: **if** $-a \leq z \leq a$ **then**
5: **return** UD_{based_D}
6: **end if**
7: **end for**
8: Return to step 1: for the next $B_{k,l}$

In step 1, the average of ρ_N from all SBN_k is computed using their OLI. In steps 2 and 3, the statistical value z is getting respect to UD_{base_D,ρ_N} with the fixed ρ_N from step 1. If the value z is greater/equal than -1.96 and lesser/equal than 1.96 in step 4, in step 5 the estimated distance between N_i and SBN_k is equal to the value of $base_D$ from U_{Dbase_D,ρ_N} with the fixed ρ and the estimated $based_D$. With OLI from SBNs non-beacon nodes can infer better the distance between each heard SBN.

3.8 Conclusion

In this section we detailed the distance estimation process based on hypotheses testing. We also detailed our five node localization algorithms; thereby we exposed different approaches for the node localization problem. For instance, the VL algorithm is based in simple adjusted vector which are created from the RSSI/LQI. On the other hand, the ECL algorithm uses the energy consumption information from the heard beacon node in order to determine a bounded area where the non-beacon might stay. The SzL algorithm can then determine a smaller area inside the estimated area by ECL. We also presented the hybrid algorithm localization for Wireless Sensor Network called Triangle Centroid localization. It is based in simple trigonometric figures for estimating the non-beacon nodes position as well as the cold-hot game. Finally, we introduce SBNs central idea is to report the status of the environment to each non-beacon node. In this first approach, we

use a generalization of the obstacles (ρ) through the network simulator AvroraZ. However, SBNs can transmit more useful information such as noise, lost packet, transmission delay and the bit error rate, just to name a few. This opens the possibility of develop NLAs with these kinds of requirements and extend the SBNs for specific or general environments.

Chapter 4

Our Architectures for Node Localization

In this Chapter four architectures are presented for node localization in a wireless sensor network. The first three architectures are only related to the context awareness and the fourth architecture addresses the node localization fusion as well as context awareness.

4.1 Logical Position of Nodes (LPN).

To the best our knowledge, the developed algorithms for estimating the node position in fully observable environments have not used the environment layout for estimating/improving the node localization, for instance [23, 20, 27, 36]. By contrast, LPN (*Logical Position of Nodes*) improves estimated positions of nodes by using the environment layout (partly/fully observable environment) and the attributes of the objects that can carry a node. The Figure 4.1 shows LPN working as a filter of a localization algorithm. The database of the environment layout contains the dimensions in coordinate terms, which is shared by LPN and context-aware application. The database Node-Object is the one to one relation between object $object_N$ and a node $node_N$, similarly for node-user. We say that LPN is executed in partly observable environment, when it has access to databases environment layout and Node-Object. LPN is carry out in fully observable environment, when it has access to databases environment layout, Node-Object and Object-Place. The last one contains the information of the relations among objects and places, for instance a desktop may belong to an office and a laboratory at a height of 0.75 meters. But this database is not always available.

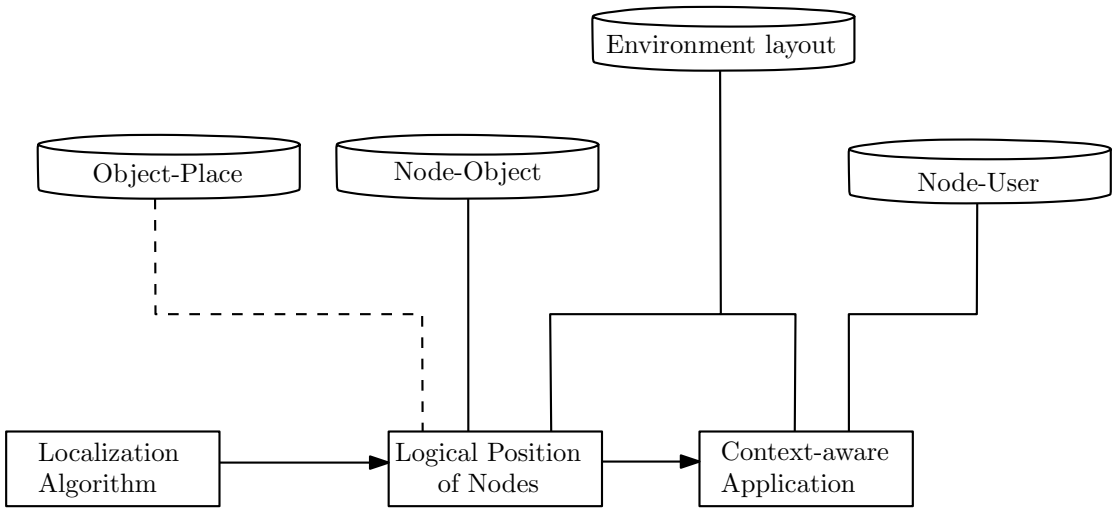


Figure 4.1: LPN working as filter of node localization algorithms.

4.1.1 Localization Algorithm

The node localization error of a NLA can be conceived as an AVU (*Area/Volume of Uncertainty*) where an unknown node might be located. In a three dimensional Cartesian space, AVU is calculated from the estimated position of the unknown node (x_e, y_e, z_e) and its average error e. The AVU can be then expressed as a cube with the lower position $AVU_{LOWER} = (x_e - e, y_e - e, z_e - e)$ and the upper position $AVU_{UPPER} = (x_e + e, y_e + e, z_e + e)$. We considered a square or cubic space in order to minimize the computational cost. After AVU calculation, both the estimated position of the unknown node and its corresponding AVU are put on the dashboard as shown in Figure 4.2.

The functionalities of nodes are determined by: (i) node Localization module which depends on the used scheme and (ii) the architecture modules: Context Awareness, Dashboard, and Likelihood. From subsection 4.3 to 4.6, the Context Awareness module is explained considering as example the relation: $node_3$-$desk_{45}$-$office_{1,2}$-0.75. In subsection 4.7, the Dashboard module is described and finally in subsection 4.8 the Likelihood module is addressed.

The sensing is only performed in the Node Localization module. The sensing mechanism such as temperature, velocity, is not part of the architecture in (ii).

LPN is made up of five steps which are explained in the follow subsections considering as example the relation: $node_3$-$desk_{45}$-$office_{1,2}$-0.75.

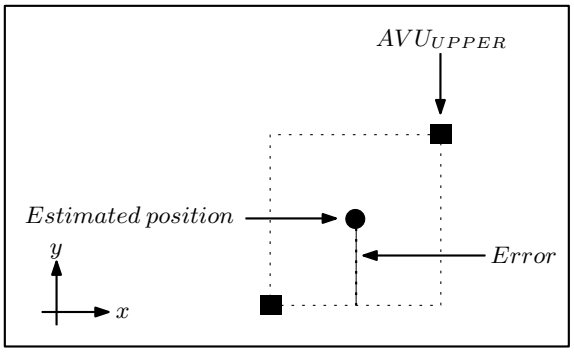


Figure 4.2: The node localization estimation and its AVU are put on the dashboard.

4.1.2 Place involved

Let a place be any space bounded by physical or non-physical limiters (segments). For instance, a laboratory, restroom, corridor, living room, bedroom, garden, or parking space, among others. In addition, they are referred under a unique identifier M. A place $Place_M$ is then represented by a set of segments s_N such that $Place_M = \{s_{1M}, s_{2M}, ..., s_{NM}\}$. The space bounded by s_{NM} is stored in the database environment layout. A place $Place_M$ is considered as a place involved, if one of its segments intersects with an AVU. The ray-box intersection method [62] is employed to resolve whether $Place_M$ is a place involved $PInvolved$; i.e., $Place_M = PInvolved \leftrightarrow s_{NM} \cap AVU$, such that $s_{NM} \in Place_M$. The result for the node $node_3$ such that $node_3$-$desk_{45}$-$office_{1,2}$-0.75 is as follows: $PInvolved$ = {$office_1$, $office_2$, $restroom_5$}. After, this process the dashboard is updated as shown in Figure 4.3.

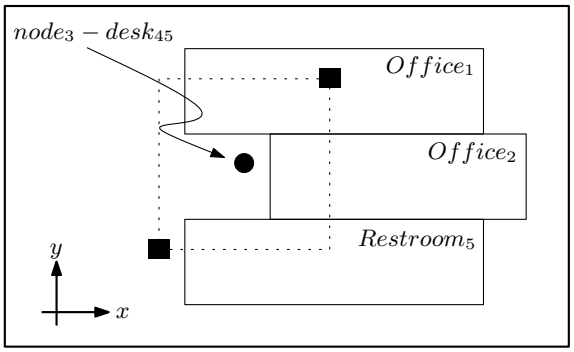


Figure 4.3: The node localization estimation and its AVU are put on the dashboard.

4.1.3 Places associated

A place involved $PInvolved$ is renamed as place associated $PAssociated$, if the unknown node is related with $PInvolved$; otherwise $PInvolved$ is eliminated. Relations of an object with places are obtained from the database Object-Place in order to determine whether an unknown node belongs to a place $Place_M$. The result for $node_3$ such that $node_3$-$desk_{45}$-$office_{1,2}$-0.75 is as follows: the place $restroom_5$ is discarded because the object $desk_{45}$ cannot belong to a restroom, therefore $PAssociated = \{Office_1, Office_2\}$. The AVU is then adjusted, $AVU = AVU - Restroom_5$. Figure 4.4 shows the dashboard after the actions of this process.

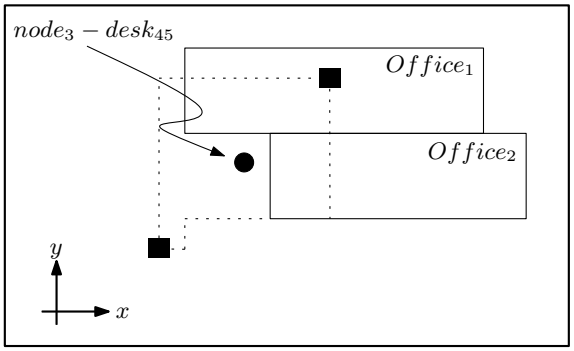


Figure 4.4: The places not related with the $node_3 - desk_{45}$are eliminated.

4.1.4 AVU compromised

The AVU is divided and adjusted to each $PAssociated$, there is then a one to one relation between a divided AVU_M and $PAssociated$ such that subscripts M have the same value i.e., AVU_M-$Place_M$ where $Place_M \in PAssociated$. Each AVU_M has also an upper position AVU_{MUPPER} and a lower position AVU_{MLOWER}. The result for the node $node_3$ is the elimination of empty spaces, as shown in Figure 4.5.

4.1.5 Spaces valid

A place $PAssociated_M$ will be renamed as fully-involved place $FIPlace_M$ if an unknown node can stay inside its corresponding AVU_M. In others words, the occupancy evaluation between an unknown node and its corresponding AVU_M is realized regarding the height (axis y) of the node (object). If the AVU_M height is greater than the node height, the AVU_M height is equal to the node height. Otherwise a place $PAssociated_M$ is discarded. The width (axis x) and/or depth (axis z) of the unknown node can also

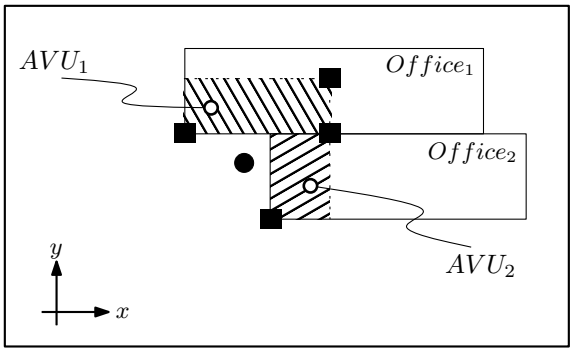


Figure 4.5: The AVU is divided and adjusted.

be taken into account in order to reduce even more the AVU_M size. The result for the node $node_3$ such that $node_3$-$desk_{45}$-$office_{1,2}$-0.75 is as follows: both offices are fully-involved places because the heights of the AVU_1 and AVU_2 are less than the unknown node height, then $FIPlace = \{Office_1, Office_2\}$. Figure 4.6 shows the dashboard after the actions of this process.

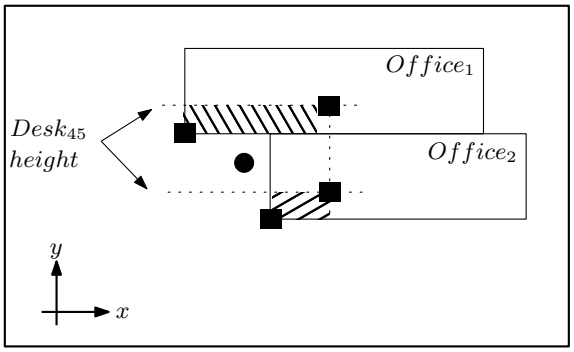


Figure 4.6: The height of AVU_1 and AVU_2 are adjustedto the $node_3$-$desk_{45}$ height.

4.1.6 Estimated position with likelihood

In this module, each AVU_M is normalized in the range $(0, 100]$ in order to indicate the object likelihood respect to each place $FIPlace_M$, as formulated in equation 4.1, where N represents the total of $FIPlace$. The estimated position of the node regarding the $FIPlace_M$ is given by equation 4.2 in which the centroid calculation between AVU_{MUPPER} and AVU_{MLOWER} is carried out.

$$L(Place_M) = \frac{AVU_M}{\sum_{M=1}^{N} AVU_M} \times 100 \tag{4.1}$$

Table 4.1: The $node_3$ likelihood regarding each place.

Node-Object	Place	Likelihood	Position
$node_3$-$desk_{45}$	$office_1$	65%	(x, y)
$node_3$-$desk_{45}$	$office_2$	35%	(x, y)

$$object_N(x, y) = \frac{AVU_{MUPPER} + AVU_{MLOWER}}{2} \tag{4.2}$$

Finally, the estimated position(s) is/are determined for the node $node_3$ as shown in Table 4.1, where the place $office_1$ has the highest likelihood of containing the unknown node $node_3$-$desk_{45}$ at the position (x, y) in a two dimensional Cartesian space.

4.2 SEA-NL: The Smart Environmental Architecture for Node Localization.

SEA-NL (*The Smart Environmental Architecture for Node Localization*) is based on the context aware computing and the architecture subsumption [28]. The Figure 4.7 depicts our architecture, which is made up of five elements: (i)The SBNs [63], (ii) the distance estimation process, (iii) NLAs, (iv) the LPN [64], and (v) the evaluator. In (i), the SBNs are able to provide two kinds of $OLIs$ among them. The OLI_{fine} has a decimal precision more than the OLI_{coarse}. For instance, $OLI_{fine} = 0.58$ and $OLI_{coarse} = 0.5$ obstacles per square meter between a couple of beacon nodes. The SBNs generates and update constantly The $OLI\ Matrix$. However, most range-based NLAs are not able to use The $OLI\ Matrix$ directly by themselves. In (ii), The $DistanceMatrix$ among beacon nodes and non-beacon nodes is estimated such as it is done in [63] by using The $OLI\ Matrix$. In (iii) the range-based NLAs employ The $DistanceMatrix$ for estimating the non-beacon node position. The range-free and/or range-based NLAs are executed in isolated way. There are then many estimated positions for each non-beacon node as used NLAs. This can be expressed by $P_{i,N}(x, y, z)$, i.e., the estimated position P of the non-beacon node i in a three-dimensional Cartesian space (x,y,z) by the NLA number N. In (iv), the LPN uses the environment layout information and the one to one relation between an object and a node for improving preliminary estimated positions of non-beacon nodes $P_{i,N}(x, y, z)$. The LPN from $P_{i,N}(x, y, z)$ provides the five-tuple $P_{i,N,M}(x, y, x, l, p)$, where l is the likelihood of each known place p to contain the non-beacon node i. The subscript M represents the different possibilities for each non-beacon node i under same NLA N.

For example, $P_{20,4,1}(1,2,3,70,1)$ and $P_{20,4,2}(2,4,6,30,6)$, i.e., through the NLA number 4 the first estimated position of the non-beacon node 20 is $(1,2,3)$ with 70% likelihood of being in the place number 1 and the second one is $(2,4,6)$ with 30% likelihood of being in the place number 6. The element (iv) is explained in greater detail in section 4.1. In (v), the *Evaluator* selects the estimated position with the highest likelihood for each non-beacon node.

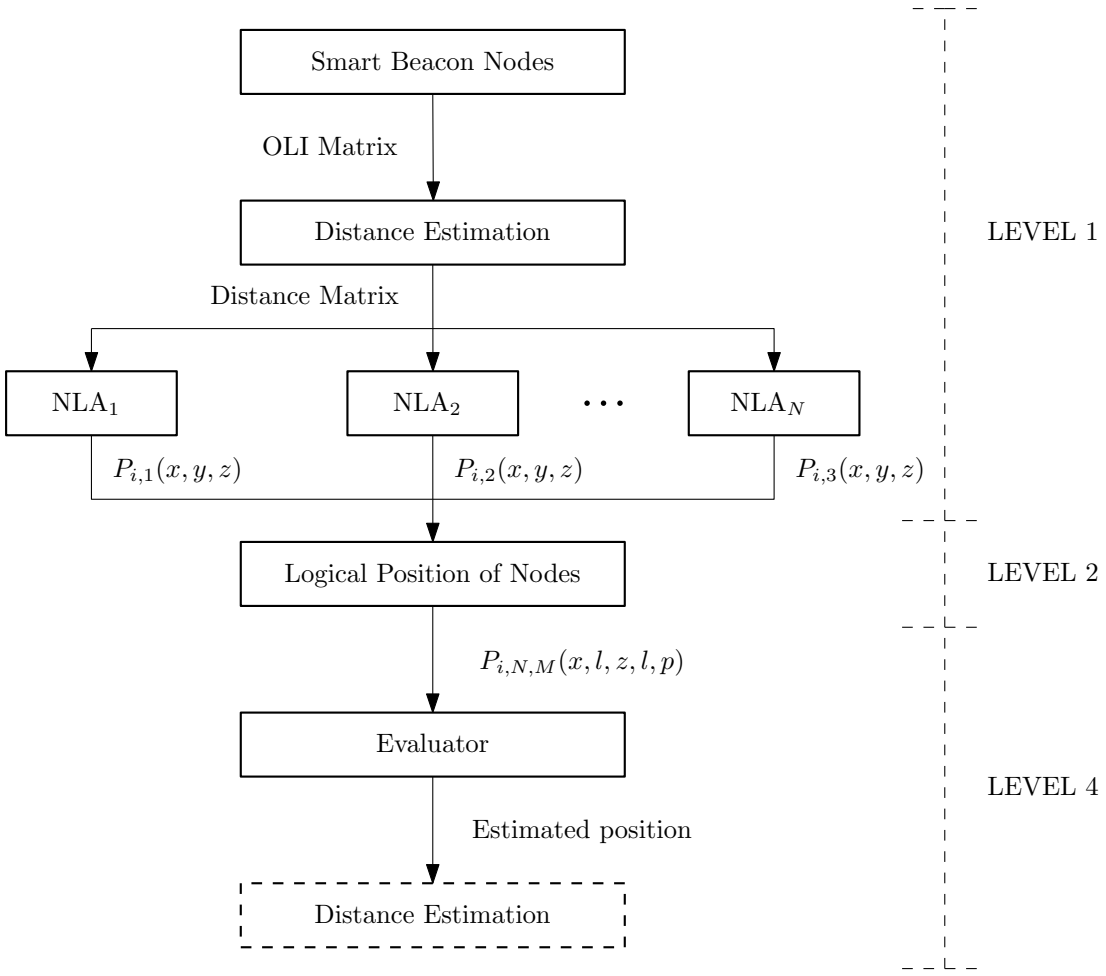


Figure 4.7: The Smart Environmental Architecture for Node Localization.

In our architecture we have identified the information fusion levels according to [46], the level 1 refines objects (nodes), which are: (i) data alignment, (ii) data/object correlation, (iii) position/kinematic and attribute estimation, and (iv) object identity estimation. The level 2 considers: (i) object aggregation, (ii) event/activity interpretation, and (iii) contextual interpretation. The level 4 includes: (i) performance evaluation, (ii) process control, (iii) source requirement determination, and (iv) mission management.

4.3 CAA-NL: A Context-Aware Architecture for Node Localization.

CAA-NL (*The Context-Aware Architecture for Node Localization*) objective is to improve the estimation of a NLA by using both the environment layout information and its corresponding objects' attributes. Our architecture is based on context awareness and architecture subsumption [28] which has been widely employed in robotics. CAA-NL is made up of five main elements: priori knowledge, node localization, context awareness, dashboard, and likelihood, as shown in Figure 4.8. The context awareness module is greater detail in the section 4.1, the other four elements are explained below.

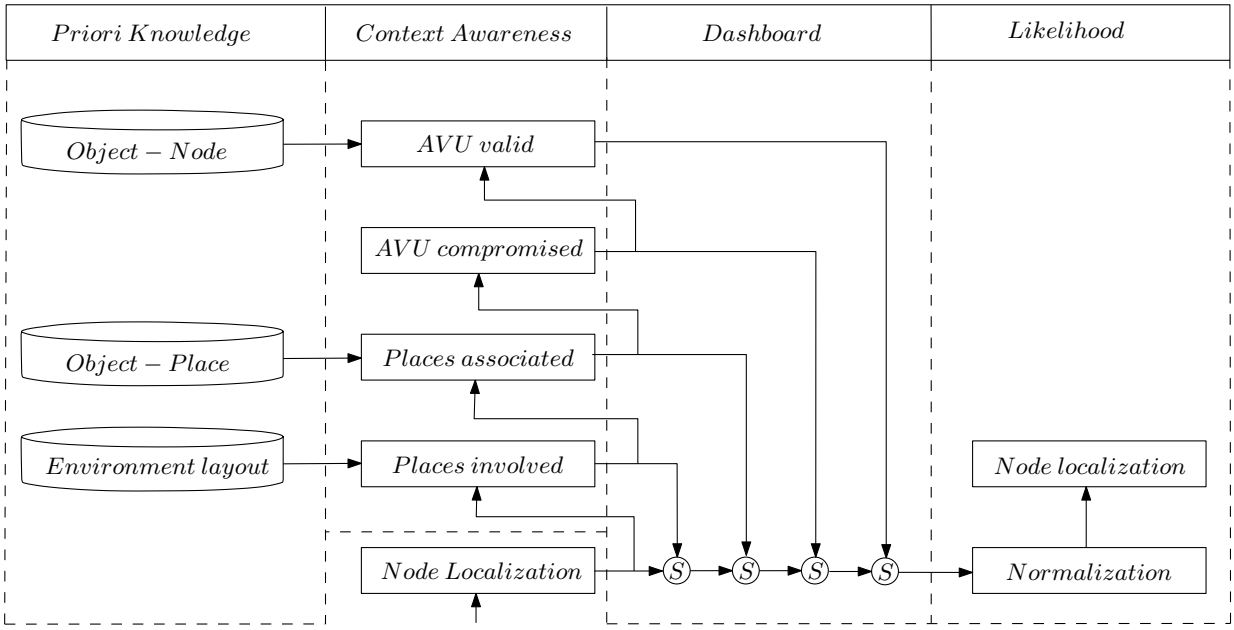


Figure 4.8: Context-Aware Architecture for Node Localization.

4.3.1 Priori Knowledge

In our architecture the priori knowledge is represented by three databases which are not generated automatically. The databases are handled as a catalog in which the information is entered manually. The database environment layout contains the dimensions and locations of known spaces, e.g. a shopping mall is divided into known spaces in a three dimensional Cartesian space (x, y, z). The environment layout database is not generated for every possible environment but rather for the application scenario where the Node Localization Algorithm is implemented. Should the environment suffers some changes; the corresponding information is updated as a catalog instead of repeating all

measurements such as in RADAR [23]. The database Object-Node contains the one to one relation between an object and a node, e.g. the node with ID number 4 is attached to the desk with ID number 6; the relation is then $node_4$-$desk_6$. Note that some common objects are considered as node themselves because of their wireless communication, such as some recent printers, laptops, cell phones, among others. The database Object-Place contains the relations among objects with places, e.g. the desk $node_4$-$desk_6$ belongs to an office $office_7$ at a height of 0.75 meters; the relation is then $node_4$-$desk_6$-$office_7$-0.75. This priori knowledge might not be easy to obtain for all places because an object can stay at different heights. Consequently, this database may not be completed for a given scenario. The CAA-NL is intended for observable indoor applications. The CAA-NL is executed in a fully-observable environment when the database Object-Place is complete. Otherwise, the CAA-NL is carried out in a partly-observable environment.

The Context-Aware Architecture for Node Localization reacts based on sensed and preloaded information in order to improve the previous estimated position of a node. The reaction can involve a non-learning mechanism such as [40, 43] in which users are alerted when information related to their location is available. Other suitable examples of this kind of systems or applications are some mobile guides for smart phones [39]. On the other hand, historical information can be used to predict future actions or intentions of entities [65] i.e., a learning mechanism. As an ongoing work, the Object-Place database will have a learning mechanism based on bootstrap weights for improving the associations among objects with places over time. However, this is beyond the scope of this paper.

4.3.2 Dashboard

The dashboard is suitable to avoid recalculating variables when not required. Besides, it can reduce the number of accesses to data storage unit(s) in the non-beacon node and allows an information priority handling. The Context Awareness module is made up of five process which are organized in ascending layers such that an upper-layer process has a higher priority than a lower-layer process. According to the subsumption architecture [28] the old information is suppress by the new one based on its priority and this is represented by the symbol "s", as shown in Figure 4.8.

4.4 HSA-NL: Hierarchical Subsumption Architecture for Node Localization

In our approach the localization error of any NLA is conceived as a square/cube which represents the uncertainty space where the non-beacon node might stay. The proposed Hierarchical Architecture for Node Localization is made up of two main phases. In the first phase estimates the node position by using a set of NLAs. In the second phase, the uncertain space is reduces employing the LPN algorithm [64]. This architecture aims to provide the estimated position of nodes as accurate as possible for any scenario making a NLAs Fusion based on execution stamps.

Our proposed architecture the NLAs' runtime is carried up in a sequential order according to execution stamps. The stamp n has higher priority than the stamp $n+1$ where n is Z^+. A NLA with the execution stamp $n+1$ depends on the NLA's outcome with the execution stamp n. For example the 2-tuple (CL_1, LPN_2), where the execution stamp number one was assigned to the CL and the execution stamp number two was assigned to the LPN. The LPN_2 requires and improves the node estimation of the CL_1. We consider a sequential-order execution of NLAs as a NLAs fusion. The criteria to form a NLAs fusion depends mainly on the environment condition, the application limitations, the NLAs' requirements, and/or human computer interactions.

The first abstraction level of our architecture is shown in the Figure 4.9 where three different kinds of nodes are employed: (i) beacon node, (ii) non-beacon node, and (iii) sink node. In (i), the notifier agent informs to non-beacon nodes the NLAs fusion to be executed regarding the delimited area where the non-beacon node stay. A delimited area can be an indoor/outdoor space with physical delimiters, non-physical delimiter, or both. In the offline process, the NLAs fusion with the best performance is taken for each delimited area. There is then a one to one relation between a delimited area and a NLAs fusion. This relation is stored in the database Fusion-Area. The relation can be expressed as $Relation(Area_M) = Fusion_N = NLA_1, NLA_2, ..., NLA_O$. In (ii), the manager agent receives the relation $Relation(Area_M)$ and it selects from the database Localization algorithms the NLAs to be executed in the localization process Localization L1. In (iii), the process Evaluation L4 expects for the outcomes of the process Context awareness L2 which improves the node estimation of the process Localization L1. The expected condition is based on a time condition or a notification. However, depending on the application the expected condition can be different. Finally, the dashed line indicates the second abstraction level of our architecture, which is detailed in Figure 4.10.

The Figure 4.10 depicts the second abstraction level of the Hierarchical Subsump-

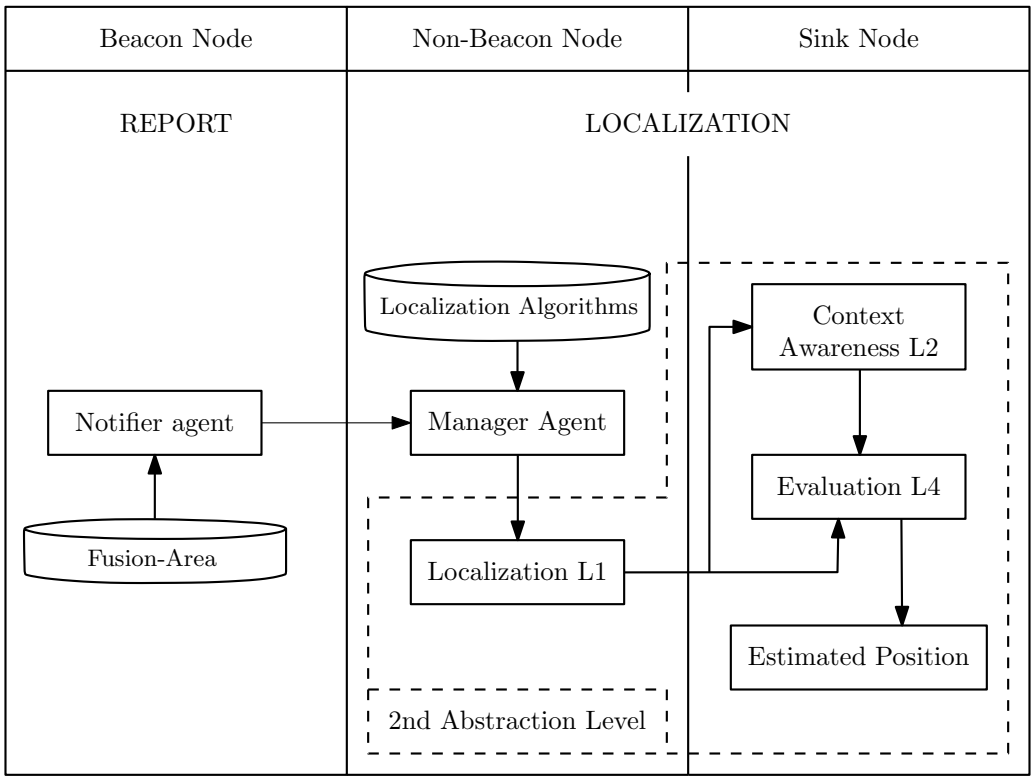


Figure 4.9: The first abstraction level of the HSA-NL.

tion Architecture for Node Localization which is based on the architecture subsumption [28]. The second abstraction level of the architecture is made up of three main elements: (i) Localization L1, (ii) Context awareness L2, and (iii) Evaluation L4. The suffixes L1, L2, and L4 correspond to the information fusion levels [46]. In (i), the database Estimated factors contains estimated factors during the execution process of employed NLAs such as the RSSI, the LQI, packages sent, the energy consumption, delays, beacon node IDs, beacon node coordinates, and so forth. Once a factor is estimated, it won't be estimated again in order to reduce the processing time.

In (ii) the LPN algorithm is performed with the last execution stamp. In this architecture part three databases are used: (a) the database Object-Node. It contains the one to one relations between an object and a node. For instance, the $desk_{45}$ has attached the $node_{33}$, then $desk_{45} - node_{33}$. Note that some common objects are considered as node themselves because of their wireless communication, e.g., some recent printers/cars, cell phones, lap tops, just to name a few. (b) The database Object-Place. It contains relations of objects with places. For instance, the $desk_{45}$ belongs to an office and laboratory. Then $desk_{45} - office_{12}$, $desk_{45} - office_{15}$, ..., $desk_{45} - laboratory_2$. (c) The database Environment. It contains the environment layout in Cartesian coordinates; however other ones can be employed. The LPN is executed in a partly observable environment, when

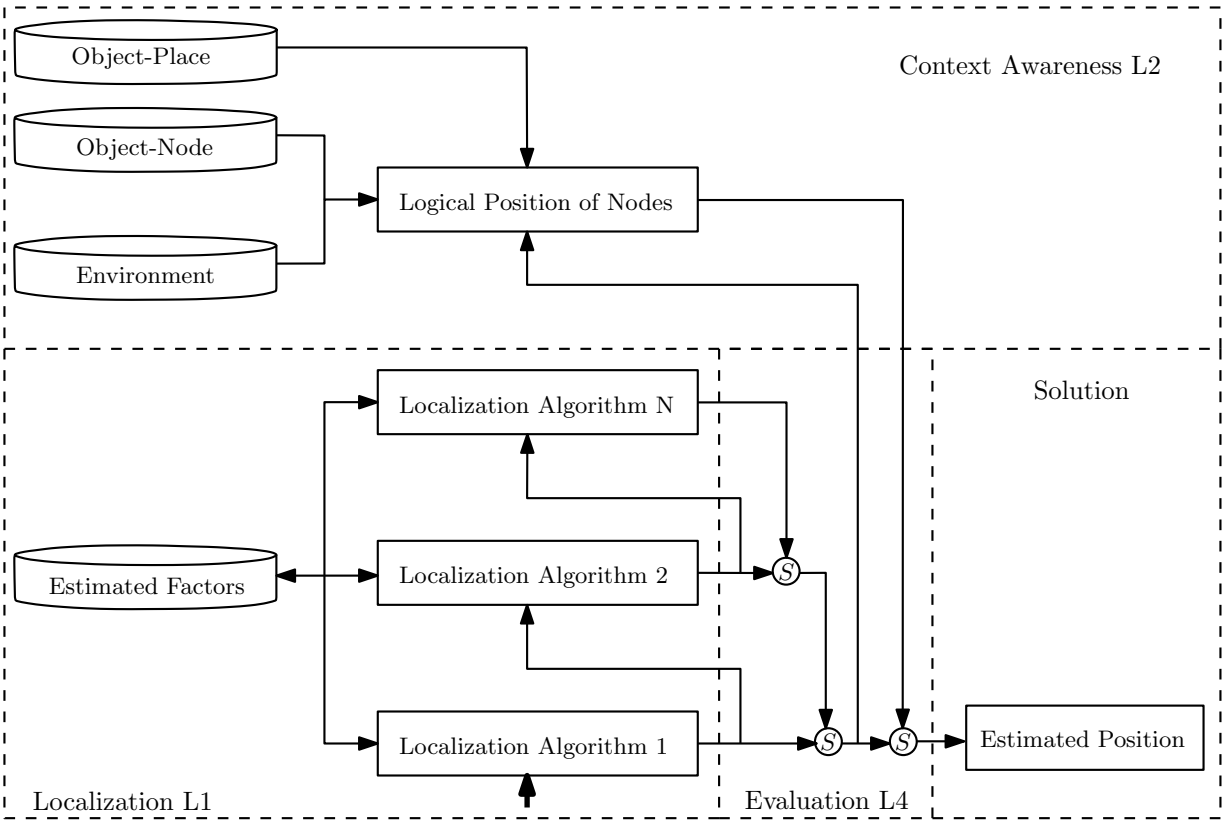


Figure 4.10: The second abstraction level of the HSA-NL.

it has access to the databases Environment and Object-Node. The LPN is carry out in a fully observable environment, when it has access to the three databases. LPN usually includes more complex process than a simple NLA; therefore it should be executed in a sink node in order to obtain the best performance. On the order hand, LPN's outcomes are more representative for a human being than a simple non-beacon node. In (iii), the symbol "s" inside of the circle means suppress [46]. It works as a control point, which allows to HA-LN provides an estimation of the node position after the first NLA has finished. The same or different expected condition can be carried out in each suppress.

4.5 Conclusion

In this section wee detailed the two different abstraction levels of the Hierarchical Subsumption Architecture for Node Localization. We also detailed the Smart Environmental Architecture for Node Localization. Both architectures have different approach but they are based on the NLAs Fusion and Context Awareness. The Smart Beacon Nodes approach has been also introduced in this section as part of the SEA-NL, however it is well-explained in the chapter 3.

Chapter 5

Experimental Evaluations and Results

5.1 Description of the scenarios

In this chapter the scenarios' specification are detailed for both the six node localization algorithms addressed in Chapter 3 and architectures presented in chapter 4. For all experimental evaluation, we used the TinyOS [57] to program the functionality of nodes and the network simulator AvroraZ [58] to simulate the WSN along with obstacles.

5.1.1 The VL, ECL, and SzL Algorithm

The three node localization algorithms VL, ECL, and SzL are an extension of Centroid Localization [17]. We tested our algorithms under a similar square topology for beacon nodes in order to make a fair benchmark as shown in Figure 5.1.

5.1.2 Triangular Centroid Localization Algorithm

Two different scenarios were defined, the first one contemplates an area of $1600m^2$ and the second one considers a volume of $32000m^3$. The beacon nodes were located in a grid way of 20m and their coverage range is about 18 meters. Both scenarios were tested under different obstruction degree at random from 1 to 10 people or obstacles are established over an occupied area.

5.1.3 Smart Beacon Nodes

Due to TCL is an extension of the Centroid Localization, we set up the position of the beacon nodes similar to the one used in [20]. The beacon nodes were located in a grid

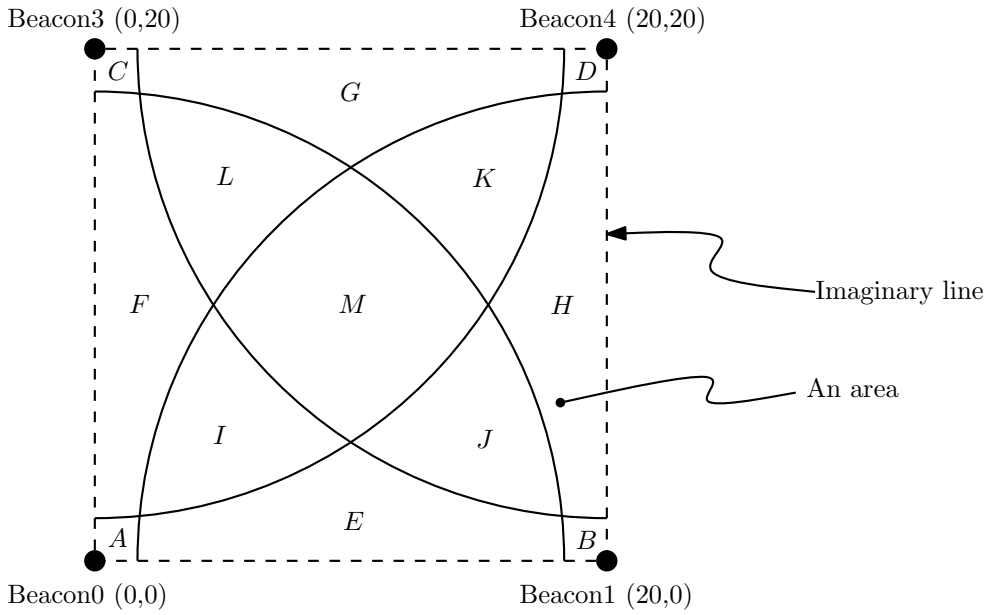


Figure 5.1: The experimental environment for ECL, VL, and SzL Algorithm.

way, with 10 meters separation among them with a coverage range about 18 meters. The first scenario contemplates an area of $1600m^2$ and the second one considers a volume of $32000m^3$. The experimental simulations were organized as follows: for each ρ of OLI_{coarse}, we created a testing group. For each testing group 30 simulations were done. A total of 300 simulations were achieved. For all simulations the non-beacon nodes were established in a random way. For instance, the testing group of $\rho[0.3, 0.39]$, such that the variable ρ was set up at random from 0.3 to 0.39. In order to measure the node localization improvement using and not using SBNs, a simple non-beacon node estimates its position ignoring any OLI, using OLI_{coarse} and using OLI_{fine}.

5.1.4 LPN, SEA-NL, CAA-NL, and HSA-NL

The Logical Position of Nodes, SEA-NL, and HSA-NL were tested in two adjoining buildings with diversity of places, each one with 8 flats. Each flat has $400m^2$ for the 2D environment and $1000m^3$ for the 3D environment. The beacon nodes were placed in a grid way with 20 meters of separation among them with a coverage range about 18 meters. A total of 6 beacon nodes were used in the 2D environment and 12 beacon nodes were employed in the 3D environment. The Table 5.1 shows the respective dimension for the 2D and 3D environments as well as the number of them.

The height of places depends of each country. For example in USA most rooms are 2,44 meters, but to the next floor is 2,6 meters allowing for the floor joists. Therefore

Table 5.1: The characteristics of the places.

Place	Number	2D dimension (meters)	3D dimension (meters)
Office	49	5x8	5x2.5x8
Corridor	12	4x20	4x2.5x20
Restroom	11	5x20	5x2.5x20
Laboratory	3	20x8	20x2.5x8
Bedroom	28	5x8	5x2.5x8
Dining room	1	20x20	20x2.5x20
Classroom	2	20x8 & 15x8	20x2.5x8 & 15x2.5x8
Warehouse	1	5x8	5x2.5x8

we consider 2,5 meter of height for each place. Widths for all places are five meters and depths are considered from eight meter to 20 meters. Relations of objects with places are considered as most commons, a desk/table ∈ Office, Dining room, Classroom, Laboratory at height of 0.75 meters; a chair ∈ Office, Dining room, Classroom, Laboratory, Warehouse at height of 0.45 meters; a printer ∈ Office, Laboratory, Warehouse at height of 0.9 meters; and a bed ∈ bedroom at height of 0.45 meters. The node height where it is attached to the object was determined based on the objects we have in our university. The simulated objects in both environments (2D and 3D) were 5 desk, 5 chairs, 5 printers and 5 beds, which were positioned at random into places valid with their respective heights according the database Object-Place. The obstacle levels were set up haphazardly from 0 to 10 persons over an occupied area. A total of 20 simulations were achieved for each scenario.

5.1.5 SEA-NL

Our simulated indoor scenarios were two: (a) two adjoining buildings with diversity of places (107), where each building had eight flats, and each flat had a surface of $400m^2$. (b) Two adjoining buildings with diversity of places (107), where each building had eight flats, and each flat had a volume of $1000m^3$. Beacon nodes for both scenarios were placed in a grid way with a minim separation of 20 meters. The places considered were: 49 offices, 12 corridors, 11 restrooms, 3 laboratories, 28 bedrooms, 1 dining room, 2 classrooms, and 1 warehouse Table 5.1. The objects considered were: 5 desks/tables, 5 chairs, 5 printers, and 5 beds. The relations of the objects with the places were as follows: a desk/table ∈ Office, Dining room, Classroom, Laboratory at height of 0.75 meters; a chair ∈ Office, Dining room, Classroom, Laboratory, Warehouse at height of 0.45 meters; a printer ∈ Office, Laboratory, Warehouse at height of 0.9 meters; and a bed ∈ bedroom at height of 0.45 meters. We considered the relations and the object's height according to the objects

we have in our center of research. Our simulated outdoor scenario is shown in the Figure 5.2, where at every intersection of brown boxes there is beacon node. A total of 512 beacon nodes were simulated in an area of 236,800 m2. The river and black areas represent places where non-beacon nodes cannot stay due to some limitation of the environment.

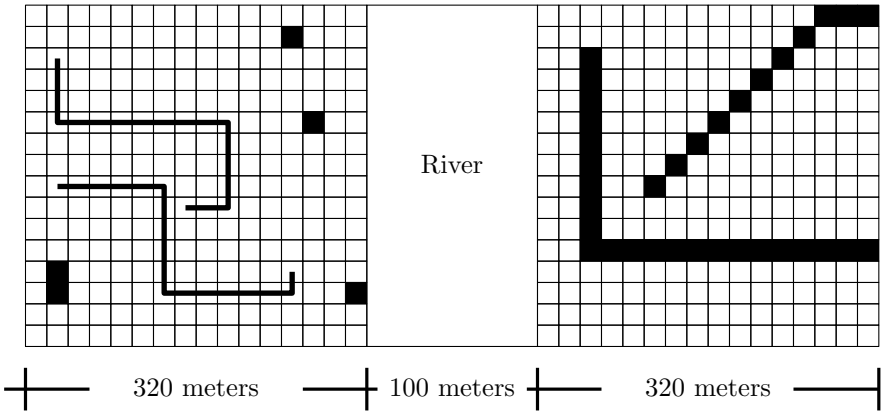


Figure 5.2: The 2D outdoor scenario for SEA-NL.

5.2 Analysis of results

Through the next subsection a comprehensive analysis of results is carried out for each of the proposed node localization algorithms as well as the architectures.

5.2.1 The Vectorial Localization Algorithm

In Table 5.2 is shown the average node localization error after 220 simulations by using the RSSI and the LQI.

Table 5.2: The accuracy performance of CL, WCL, and VL in isolation.

	CL	WCL (RSSI)	VL (RSSI)	WCL (LQI)L	VL (LQI)
Error (m)	3.39	4.29	2.06	5.94	2.83

The Table 5.2 shows that WCL (RSSI) has worse accuracy than CL. WCL increases the node localization error of CL 26.5%. The proposed algorithm VL has better precision VL than CL reducing the error 40%. In the case where it is considered LQI to estimate the distance between nodes, WCL (LQI) increase the error 75% based on the CL. On the other hand the VL error decreased the error 16.5%. On the other hand, in order

to measure the accuracy of the CL, WCL, and VL we assumed the distance separation among nodes as accurate as shown is Table 5.3.

Table 5.3: The accuracy performance of CL, WCL, and VL in isolation assuming the distance separation as accurate.

	CL	WCL	VL
Error (m)	3.39	2.09	1.6

We note that only under this unreal assumption the WCL improves 40% the CL's accuracy. However, the proposed VL improves the CL's accuracy 53%.

5.2.2 The ECL, VL, and SzL algorithms

In Table 5.4 is shown the average node localization error after 220 simulations by using the RSSI and the LQI.

Table 5.4: The accuracy performance of CL, WCL, VL, ECL, and SzL.

	CL	ECL	WCL (RSSI)	VL (RSSI)	ZsL (RSSI)	WCL (LQI)	VL (LQI)	SzL (LQI)
Error(m)	3.61	3.61	4.33	2.1	1.83	6.1	2.86	2.48

The SzL improved the CL's accuracy in 49.3% and the WCL's accuracy in 57.73% by using the RSSI. The SzL improved the CL's accuracy in 31.3%, the WCL's accuracy in 59.34% and the VL's accuracy in 13.28% by using the LQI. In order to measure the accuracy of the CL, WCL, VL and SzL we assumed the distance separation among nodes as accurate. The results are shown in the Table 5.5, where the SzL improves the CL's accuracy in 71.19%, the WCL's accuracy in 49.01% and the VL's accuracy 35%.

Table 5.5: The accuracy performance of CL, WCL, VL and SzL in isolation assuming the distance separation as accurate.

	CL	**WCL**	**VL**	**SzL**
Error (m)	3.61	2.04	1.6	1.4

5.2.3 The Triangular Centroid Localization Algorithm

In Table 5.6 shows the simulation results. With these outcomes, we demonstrate that the TCL is superior to CL and WCL in a 99.9% on average; and WCL gets better performance than CL in 37.76%. However, by the irregularity of the wave propagation is

not possible get exact distances among nodes in real scenarios. The Table 5.7 shows the localization error of algorithms when the transmission is affected by people or obstacles.

Table 5.6: The accuracy performance of CL, WCL, and TCL in isolation.

	CL	**WCL**	**TCL**
Error (m)	3.39	2.11	0.0001

Table 5.7: The accuracy performance of CL, WCL, and TCL in isolation assuming the distance separation as accurate.

	CL	**WCL (RSSI)**	**WCL (LQI)**	**TCL (RSSI)**	**TCL (LQI)**
Error (m)	3.39	4.29	5.94	1.54	2.08

The WCL is presented as extension of CL, nevertheless in our simulations WCL increases the localization error in 26.55% using RSSI and in 75.22% using LQI; it only reduces the localization error when the distances among nodes are assumed as accurate. In this sense, we believe that WCL should be further extended. TCL improves to CL in 54.5% using RSSI and in 38.68% using LQI. TCL improves to WCL in 64.1% using RSSI and in 64.98% by using LQI.

5.2.4 The Smart Beacon Nodes

The Figure 5.3 shows the average result for each group of ρ of OLI_{coarse}. Notably we can see that the localization error tends to increase when ρ increases, besides we can see the error localization for all NLAs is bigger when $\rho[0.9, 0.99]$ than everyone else. The maximum obstacle is 0.99 that is 17.81 people over an occupied area. Nevertheless, if the ρ increases, not necessarily increases the localization error. For instance, the node localization error of the WCL when it uses LQI with $\rho[0.3, 0.39]$ is lower than $\rho[0.0, 0.09]$. Similarly, these situations also can be found in the TCL. The random position of the non-beacon nodes and density/positions of beacon nodes are also important factors.

The Figure 5.4 draws the node localization error by using OLI_{coarse}. Once again, we can see the tendency of the node localization error to increase. The average error decreased significantly for some groups of ρ and for others only slightly. The localization error for all ρ of each range-based NLAs was reduced. For instance, the localization error of TCL by using LQI was reduced in a 7.53% on average and by using the RSSI was reduced in a 7.3% on average; the localization error of WCL by using the LQI was reduced in a 7.03% on average and by using the RSSI was reduced in a 5.73% on average.

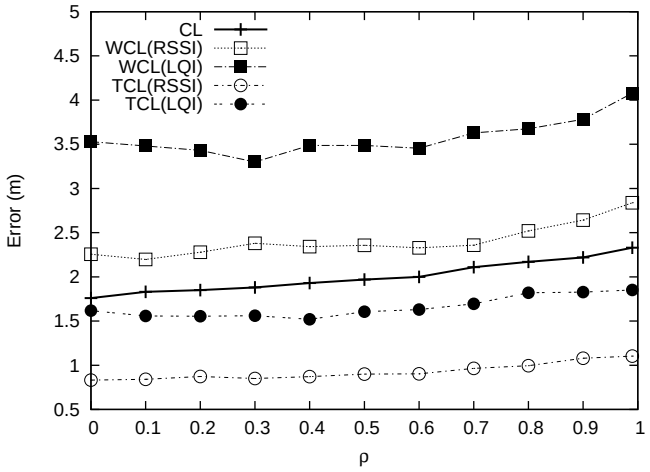


Figure 5.3: The average error of node localization algorithms without SBNs.

On the other hand, Centroid is not benefited by SBNs, due to that it does not use angle or distance for estimating the non-beacon node position, therefore it is not able to use this class of information. With this we emphasize that by using SBNs, the localization error of range-based localization algorithms can be reduced.

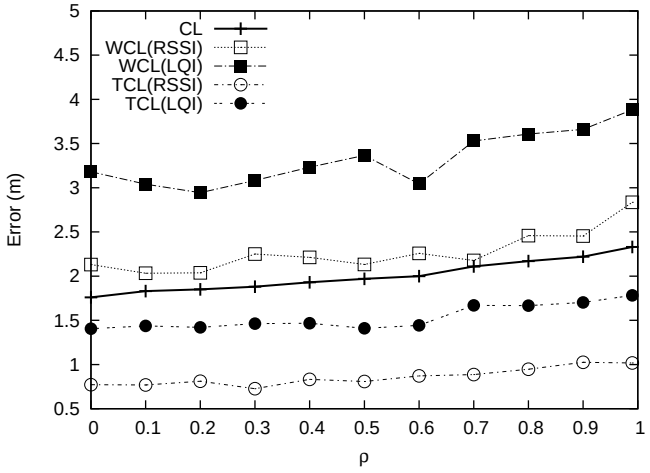


Figure 5.4: The average error using coarse ρ.

As already mentioned at the beginning of this document, for the critical applications the precision in node localization is considered as primary, where every centimeter gained can be very important. OLI_{fine} can be used for these kinds of applications in order to get more accuracy. The Figure 5.5 depicts the localization error of the used NLAs by using OLI_{fine}, where the localization error of WCL by using LQI was reduced in an 11.92% on average and by using RSSI was reduced in a 12.24% on average. The localization error of TCL by using LQI was reduced in an 11.92% on average and by using

RSSI in a 12.24% on average. The highest improvement was found in TCL, it was 18%. However, the error does not necessarily decline with a finer ρ (OLI_{fine}) for all cases. For instance, let be $\rho = 1.9$ for some couple of beacon nodes SBN_{kl}, the Algorithm 7 for estimating OLI_{fine} stops in $UD_{base-D,\rho 14}$ due to the statistical value z is within the range of the acceptance region. On the other hand, The Algorithm 7 for estimating OLI_{coarse} stops in $UD_{base-D,\rho 2}$. Thus, OLI_{coarse} gets better accuracy than OLI_{fine}. However in most cases, OLI_{fine} gets better accuracy.

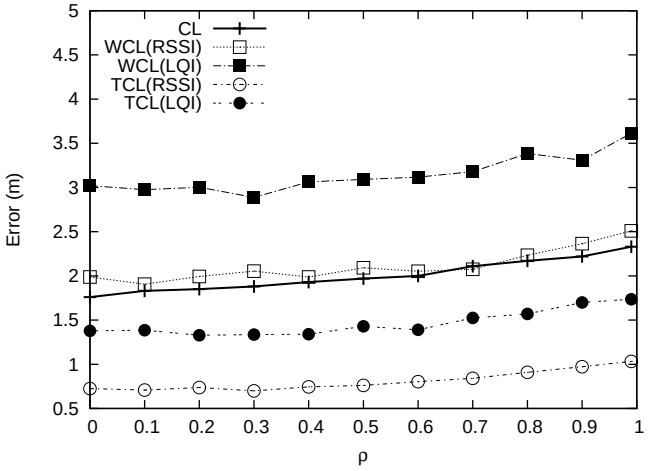


Figure 5.5: The average error using fine ρ.

Summarizing, when SNBs transmit OLI_{coarse}, the node localization error of used range-based LAs is improved in a 6.9% on average and OLI_{fine} the node localization error is ameliorated in a 12.08% on average. In Table 5.8 is shown the localization average error (meters) of tested localization algorithms when they do not use OLI (Err) and when they use OLI (Err ($OLI_{coarse/fine}$)). The Table 5.8 shows the average accuracy of the localization algorithms tested in 2D and 3D environments. Based on these results, we proofed that the inclusion of SBNs into the range-based localization algorithms TCL and WCL can decrease the node localization error up to in an 18%. SBNs can transmit OLI_{fine} for getting more accuracy; of course this indicator demands a little more off-line work than OLI_{coarse}.

The SBNs are developed in the network simulator AvroraZ, therefore the constraints of a real implementation is not possible to analyze in our simulations, however the potential constraints of use of SBNs into some localization algorithm may be are: (a) time for generating the database of RSSI/LQI and (b) establish the range acceptation region.

Table 5.8: The accuracy performance of NLAs with/without SBNs.

	Err	**Err**(OLI_{coarse})	**Err**(OLI_{fine})
CL	2	2	2
WCL (RSSI)	2.41	2.27	2.11
WCL (LQI)	3.57	3.32	3.15
TCL (RSSI)	0.93	0.86	0.81
TCL (LQI)	1.66	1.53	1.46

5.2.5 Logical Position of Nodes.

The LPN improves the node localization of Centroid, WCL, and TCL in 13.27% on average. The high improving is 18.13% on average for WCL when it uses RSSI. The minimum improving is 9.77% on average for TCL when it uses LQI. In the Table 5.9 is shown the localization average error in meters for each localization algorithm with/without LPN for 2D and 3D environments. In general, the improving of the LPN is directly related with the database environment layout and the Area/Volume of Uncertainty (localization error).

Table 5.9: The Localization Error of NLAs with/without LPN.

	2D (m)		**3D (m)**	
	Without LPN	**With LPN**	**Without LPN**	**With LPN**
CL	3.11	2.76	3.67	3.1
WCL (LQI)	5.51	4.63	6.37	5.32
WCL (LQI)	4.08	3.34	4.5	3.8
TCL(LQI)	1.91	1.72	2.25	2.03
TCL(RSSI)	1.32	1.19	1.76	1.58

5.2.6 SEA-NL.

The Figure 5.6 5.7 and 5.8 shown the localization average errors of the node localization algorithms used (the CL, the TCL, and the WCL) for the three scenarios, respective. The first column (from left to right) represents the error of the NLA in isolate way. The second column represents the localization average error of the NLA through the LPN; the third column shows the localization average error of the NLA by using SBNs, and the forth column indicates the localization average error of the NLA as part of the architecture SEA-NL. Note that SBNs are expedient only for those NLAs based on distance estimation, therefore SBNs do not benefit to the CL, but the CL whether is improved by the LPN and the SEA-NL

In Table 5.10 is shown the improvement of the node localization algorithms CL, TCL, and WCL by using the LPN, the SNBs, and our architecture SEA-NL. In the first scenario the highest improvement found of the SEA-NL was 24.24% on average in the TCL (RSSI). The smallest improvement found of the SEA-NL was 11.25% on average in the CL. In the second scenario the highest improvement found of the SEA-NL was 30.88% on average in the WCL (RSSI). We can see in the Figure 5.7 the average localization error was reduced from 4.5 to 3.11 meters. The smallest improvement found of the SEA-NL was 15.53% on average in the CL. In the third scenario the highest improvement found of the SEA-NL was 24.06% on average in the TCL (RSSI). The smallest improvement found of the SEA-NL was 10.32% on average in CL.

The SEA-NL improves to NLAs CL, WCL, and TCL is ∼19.14% in the first scenario, ∼ 21.85% in the second scenario, and ∼ 18.56% for the third scenario. Summarizing, the SEA-NL improves to the NLAs used in ∼ 19.85%. The fluctuation of the SEA-NL accuracy is provoked by the node position, density of nodes, the space layout, valid spaces, invalid spaces, and the density of obstacles over each occupied area, among others. The consideration of entire factors of a simple scenario into a simple NLA might be is not possible or feasible; instead of this, some NLAs have been developed for a specific scenario where only considered most relevant factors such as the RADAR [23].

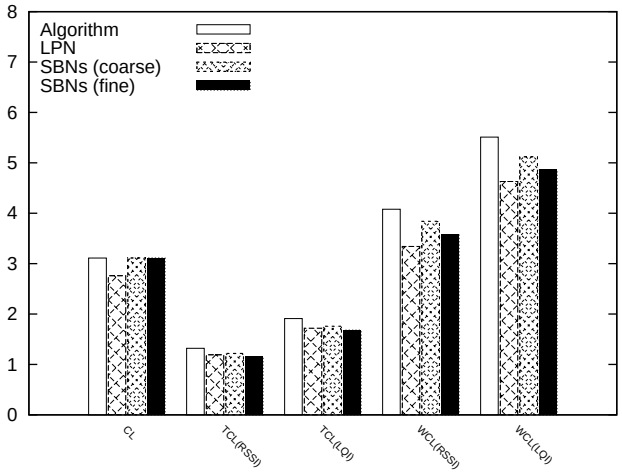


Figure 5.6: First scenario results. Two adjoining buildings, 2D.

5.2.7 CAA-NL

The CAA-NL is intended for those NLAs with poor to fair performance in order to increase their accuracy based on context awareness. In our approach the node localization error is conceived as a square/cube. The CAA-NL aims to reduce the square/cube. If

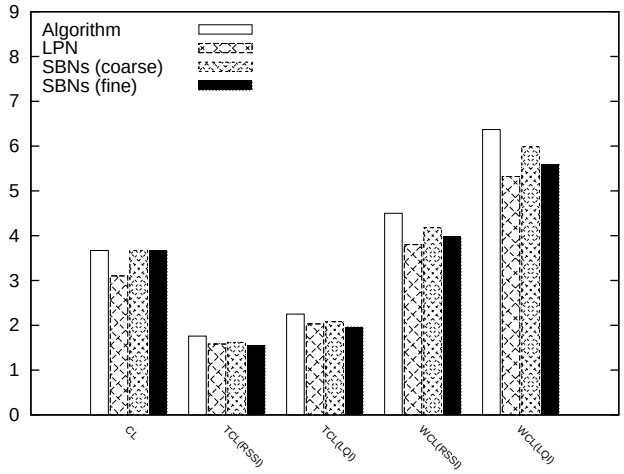


Figure 5.7: Second scenario results. Two adjoining buildings, 3D.

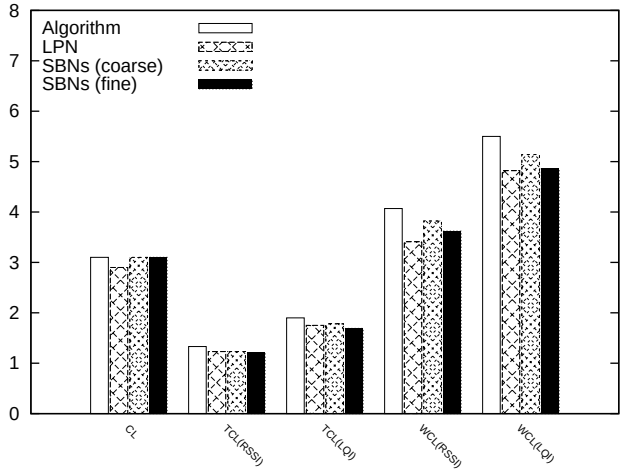


Figure 5.8: Third scenario results. Outdoor scenario from Figure 5.2.

the shape size is too small the context might not necessarily be useful enough to produce an improvement. Nevertheless, high localization accuracy for indoors can demand: (i) special hardware [18] or (ii) a large number of beacon nodes with short coverage range [40]. Our proposed architecture compensates the lack of special equipment and reduces the number of required beacon nodes.

We implemented the Centroid Localization [17], Triangular Centroid Localization [21], Weighted Centroid Localization [7], and Improved Centroid Localization [38] algorithms to validate our architecture CAA-NL. In order to be fair, the used range-based algorithms run the same distance estimation technique based on the hypothesis testing employed in [21]. In addition, they were evaluated by using both the RSSI and the LQI. Obstacles were set up at random by $\rho = [0.01, 0.99]$, where beacon nodes have a coverage range of about 36 meters of diameter with the aim of avoiding the use of a large number

Table 5.10: Improvement of the NLA's for the three scenarios.

	CL	**TCL RSSI**	**TCL LQI**	**WCL RSSI**	**WCL LQI**
Fist Scenario. Two adjoining buildings, 2D					
LPN	11.25%	11.25%	9.84% 9.94%	18.13%	15.97%
SBNs (coarse)	0%	7.57%	7.85%	5.88%	7.07%
SBNs (fine)	0%	12.87%	12.04%	12.5%	11.79%
SEA-NL	11.25%	24.24%	18.84%	23.03%	18.33%
Second scenario results. Two adjoining buildings, 3D.					
LPN	15.53%	10.22%	9.77%	15.55%	16.48%
SBNs (coarse)	0%	7.95%	7.55%	7.11%	5.97%
SBNs (fine)	0%	11.93%	12.88%	11.77%	12.4%
SEA-NL	15.53%	20.45%	20.88%	30.88%	21.5%
Third scenario results. Outdoor scenario from Figure 5.2					
LPN	6.45%	7.52%	7.89%	16.21%	12.36%
SBNs (coarse)	0%	7.51%	6.32%	6.14%	6.55%
SBNs (fine)	0%	9.02%	11.05%	11.05%	11.63%
SEA-NL	10.32%	24.06%	19.47%	21.13%	17.82%

of them as in some context-aware applications. For indoor environments, a large number of beacon nodes involves a bigger effort for establishing them and it could not be desirable from an economic perspective.

5.2.7.1 Accuracy Evaluations

The goal of this set of experiments is to determine the enhancement of the used NLAs through the CAA-NL. We considered a couple of contiguous buildings with a diversity of places. Each building has eight flats, where each flat has a surface of $400m^2$ for the 2D environment and a volume of $1000m^3$ for the 3D environment. The beacon nodes were placed in a grid way with 20 meters of separation among them. A total of six beacon nodes were used in the 2D environment and 12 beacon nodes for the 3D environment. Table 5.1 shows the dimension and quantity of places for both environments in a two/three dimensional Cartesian space.

The height of some places might be regulated by a building Act. For example, in the USA most rooms are 2.44 meters height, but to the upper floor is 2.6 meters allowing for the floor joists. Therefore, we considered 2.5 meter of height for each place. Widths for all places are 5 meters and depths were considered from 8 to 20 meters. The relations of the objects with the places were considered as follows: a desk/table $\in$ {Office, Dining room, Classroom, Laboratory} at a height of 0.75 meters; a chair $\in$ { Office,

Dining room, Classroom, Laboratory, Warehouse } at a height of 0.45 meters; a printer $\in$ {Office, Laboratory, Warehouse} at a height of 0.9 meters; and a bed $\in$ {Bedroom} at a height of 0.45 meters. A total of eighty objects were considered: 20 desktops, 20 chairs, 20 printers, and 20 beds.

5.2.7.2 Accuracy Experimental Results

A total of 100 simulations were performed for each studied NLA considering the eighty objects placed at random every time. Fig. 5.9 and Fig. 5.10 show the simulation compendium of the studied NLAs for the 2D and 3D environment. The left column for each algorithm represents its average error and the right column denotes the average error of the CAA-NL using the algorithm in question. In the 2D environment, CAA-NL improves $\sim$ 13.08% the used NLAs' accuracy. The average-largest improvement is 18.14% for the WCL algorithm when it uses the RSSI. The average-smallest improvement is 9.94% for TCL when it uses the LQI. In the 3D environment, CAA-NL improves 13.03% the used NLAs' accuracy. The average-largest improvement is 16.48% for the WCL algorithm when it uses the LQI. The average-smallest improvement is 10.71% for TCL when it uses LQI. The average-highest improvement for the WCL algorithm can be explained by its initial AVU which is the biggest compared with the other used NLAs. The node localization error reported by the authors of WCL ranges from $\sim$ 0.7 to $\sim$ 14.3 meters considering a 10 meter separation among the beacon nodes. The CAA-NL is able to reduce in average up to 2.59 meters the WCL error. In other words, based on the fact that in the USA most rooms are $\sim$ 2.5 meters height, the CAA-NL is able to reduce the location estimation as much as the equivalent to two stories from a regular building.

The environment layout and object-place relation are key factors for reducing the AVU size. Based on these two factors an initial AVU can be reduced and consequently the localization estimation can be improved. For instance, consider five places where each one has 20% of an initial AVU and only one place is a place associated, the AVU is then reduced by 80%.

The CAA-NL is able to improve the accuracy of those NLAs with reasonable accuracy, i.e. if the initial AVU is too small (centimeters), CAA-NL could not be able to reduce it. However, a too small AVU for range-based NLAs requires: (i) special hardware or (ii) many beacon nodes with a short coverage range. The ABS [27] is a suitable example of (ii).

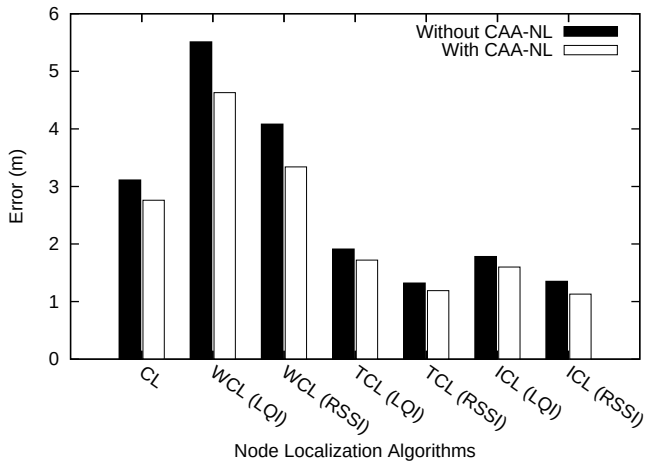


Figure 5.9: The used NLAs' accuracy in the 2D environment

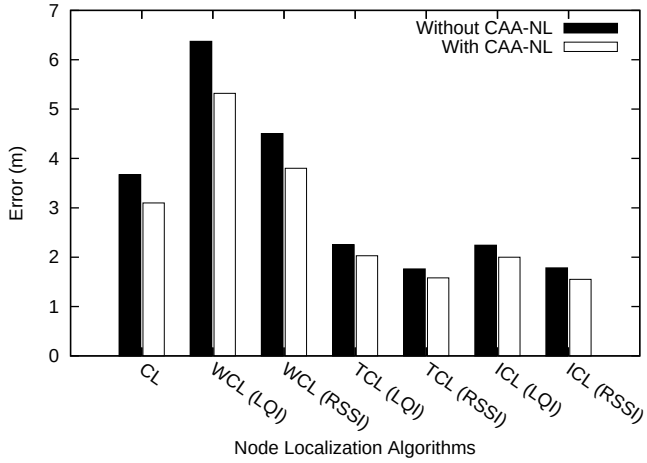


Figure 5.10: The used NLAs' accuracy in the 3D environment

5.2.8 Reducing the number of required beacon nodes.

The goal of this experiment is to determine the number of beacon nodes that can be removed by using our proposed architecture CAA-NL i.e., the number of beacon nodes that can be reduced while preserving the average accuracy of the NLAs.

5.2.8.1 Reducing the number of required beacon nodes experimental results

Extensive simulations were carried out in order to determine the localization average error of the CL, WCL, TCL, and ICL algorithms. The placement of beacon nodes was randomly chosen. The number of beacon nodes was incremented progressively from 6 to 50. A total of 30 simulations were performed. This process was executed for each of

the considered NLA giving a total of 5280 simulations. The node localization accuracy of the four considered NLAs improves as the number of beacon nodes is increased, see Fig. 5.11. Based on our simulation results, our proposed architecture CAA-NL reduces the number of required beacon nodes by ∼25.93%, as shown in Table 5.11, among other statistical valuesââ. A clear upward or downward tendency over the number of beacon nodes is not observable because of the random distribution of nodes, environment layout, and relation object-place. However, other important factors can be considered such as the environment dimensions, coverage range of nodes, and used NLA. Despite all factors involved our proposed architecture is able to reduce the number of required beacon nodes while keeping the same accuracy of the NLA. For instance, The average error of the WCL algorithm using 15 beacon nodes is ∼ 3.73 meters, on the other hand through CAA-NL using 7 beacon node is ∼ 3.7 meters, i.e. the reduction of beacon nodes is 55.33% which represented the average-largest reduction, as shown in Fig. 5.12. This reduction has a positive impact from both an economic perspective and infrastructure effort.

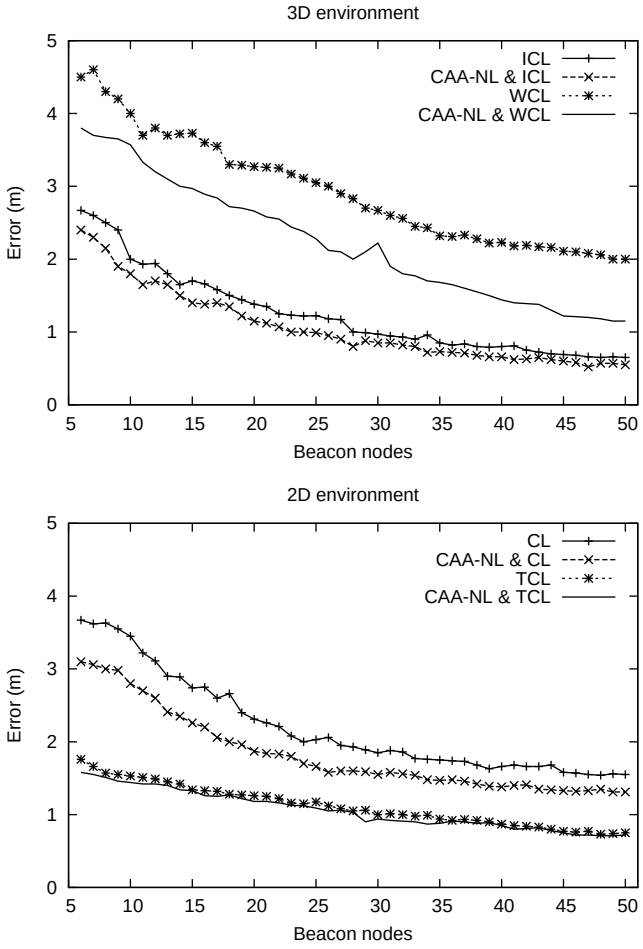


Figure 5.11: Node localization error regarding the number of beacon nodes.

Table 5.11: Reduction of Bacon nodes (%) by CAA-NL

	ICL	WCL	CL	TCL	
Minimum	7.14	13.28	25	2.22	**11.91**
Average	18.5	38.45	33.82	12.94	**25.93**
Maximum	33.3	55.33	50	33.33	**43**
Std Dev	4.8	7.54	5.34	7.86	**6.4**

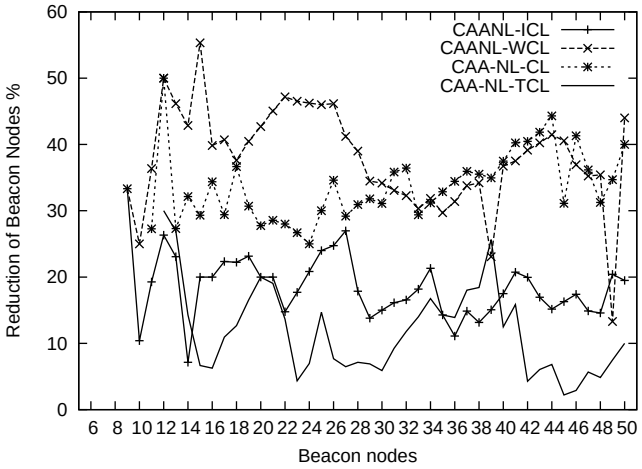


Figure 5.12: Node localization error regarding the number of beacon nodes.

5.2.9 Limitations and constraints

The Context-Aware Architecture for Node Localization objective is to improve the estimation of a NLA by using both the environment layout information and its corresponding objects' attributes. For example, a printer cannot stay on the ceiling or in a restroom. In other words, several space and belonging restrictions can be established from a known object considered in a given context. Accomplishing the same accuracy for all possible applications is not the goal of our proposed CAA-NL. In fact, we believe that same accuracy is not achievable because of the diversity and number of factors which might be required to address.

The complexity is determined by the maximum complexity given by either (i) the Context Awareness module or (ii) the Node Localization module. In (i) the module has order of n time complexity. In (ii) the complexity is subjected to the implemented Node Localization Algorithm, for instance the Weighted Centroid Localization has order of n time complexity or even worst the Multidimensional Scaling has order of n^3 time complexity. In (i) the communication cost is negligible since it involves a unique ACK packet per unknown node which informs the estimated position. For example in (ii) the

communication cost can be high if the used NLA is a hop-count based approach.

The CAA-NL scalability is subjected to the node localization module scalability. For instance, if the implemented node localization algorithm has order of n^2 time complexity or higher, then neither the algorithm [66] nor the CAA-NL are scalable. In other words, the CAA-NL scalability depends on the employed algorithm. The CAA-NL scalability can also be bounded by the network properties. For example, a zigbee network can consist of a maximum of 65535 nodes distributed in subnetworks of 255 nodes.

5.3 HSA-NL

Some NLAs are able to performance better than others in the same NLAs fusion, but it depends on their requirements and flexibility. The results show that a NLAs fusion can significantly reduce the localization error estimation. However, the size of the fusion is not directly related to the amount of the error reduction.

For instance, in Figure 5.13 several NLAs fusions are shown where the NLAs Fusion of size five CL_1, VL_2, VL_3, VL_4, VL_5 reduce the node localization error from 5.07 to 3.1 meter. On the other hand, the NLAs fusion of size three CL_1, TCL_2, SzL_3 has the same accuracy. However, the NLAs Fusion of size five is able to performance better than a NLAs fusion of size three under other environment conditions. With this evidence we proof that there is not a universal NLAs fusion, but there is a NLAs fusion able to performance better than a simple NLAs for each area.

The NLA's accuracy under this approach is hard to estimate because there are many factors that make the NLAs' accuracy fluctuate widely such as:

- The number of the execution stamp assigned.
- The algorithm that is performance before.
- The NLAs fusion sizes.
- The diversity of places.
- The Obstacle Level indicator.
- Quality of the databases of the LPN.
- The beacon nodes' density.

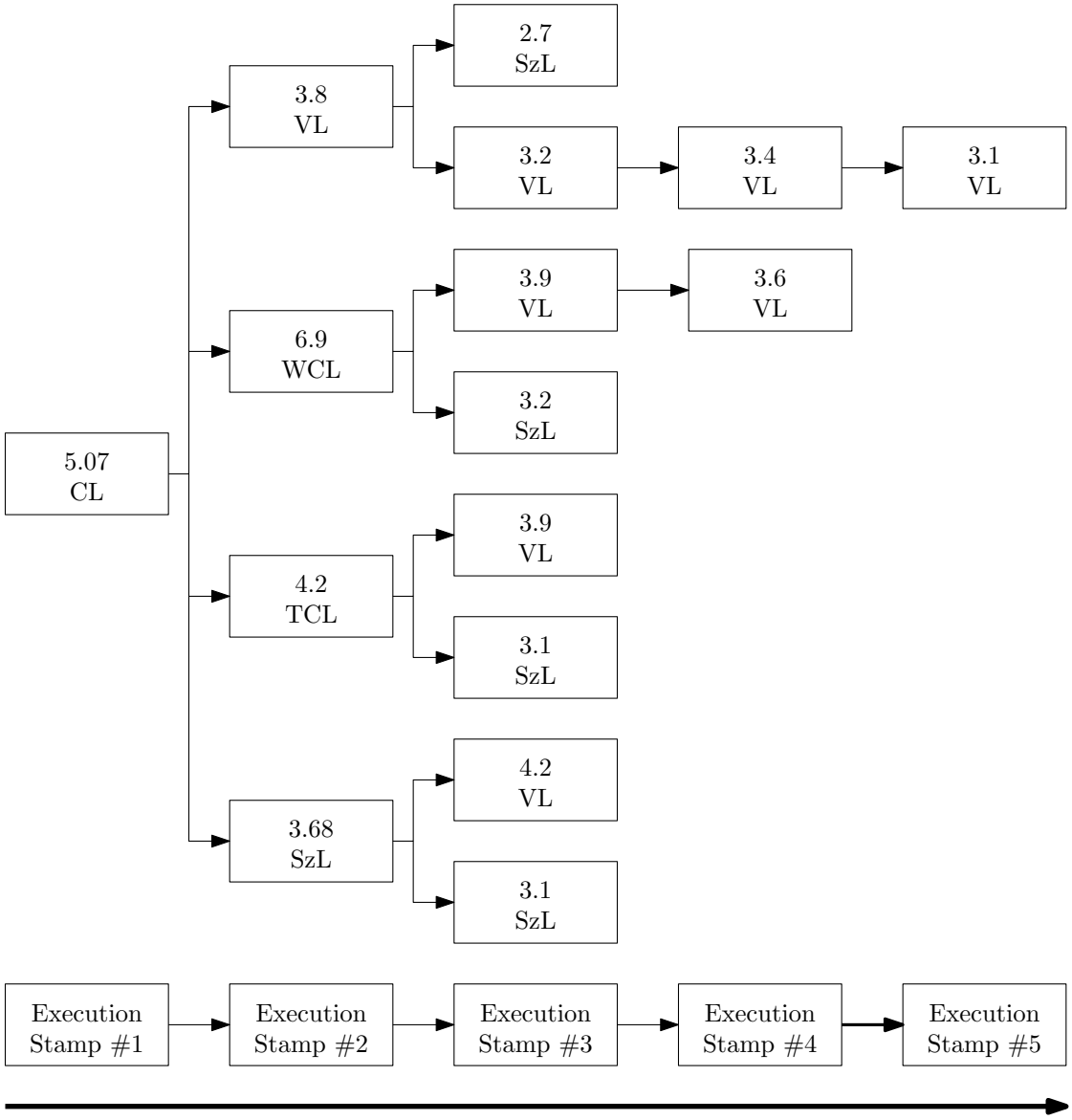


Figure 5.13: Node localization error regarding the number of beacon nodes.

Chapter 6

Ongoing Work and Conclusion

The Triangle Centroid localization algorithm is one of the most powerful algorithms which is based in simple trigonometric figures. After extensive simulations, we prove that our proposed algorithm has better performance than CL and WCL in 2D and 3D environments considering different levels of obstacles. Using the RSSI, TCL improves the accuracy of CL in 54% and WCL in 64%; using the LQI, TCL improves to CL in 38% and WCL in 64.98%. When we assume exact distances only to measure the accuracy under a perfect (unrealistic) environment TCL is superior to CL and WCL.

Smart Beacon Nodes are able to transmit the Obstruction Level Indicator between each couple of them. After extensive simulations using obstacles in random way in 2D and 3D environments, we proofed that the inclusion of SBNs into the range-based localization algorithms Triangular Centroid Localization and Weighted Centroid Localization improve their accuracy. Specifically, when SBNs transmit OLI_{coarse} LAs improve the node localization in a 6.9% on average and with OLI_{fine} they ameliorate the node localization in a 12.08% on average. In additional, the highest improvement was found in TCL, it was 18% The central idea is to report the status of the environment to each non-beacon node. In this first approach, we use a generalization of the obstacles (ρ) through the network simulator AvroraZ. However, SBNs can transmit more useful information such as noise, lost packet, transmission delay and the bit error rate, just to name a few. This opens the possibility of develop LAs with these kinds of requirements and extend the SBN for specific or general environments.

We presented Logical Position of Nodes which improves estimated position of nodes by validating the environment layout and the properties of the objects that can hold a node. LPN provides the possible positions of each node along with the likelihood of belong to each place. Our proposal was tested in a couple of adjoining buildings considering obstacles from 0 to 10 persons over an area occupied through the network simulator

AvroraZ in 2D and 3D environments. LPN improved the node localization of Centroid Localization, Weighted Centroid Localization and Triangular Centroid Localization in a 13.27% on average. Based on our experimental results, LPN can work as a filter of localization algorithms. However, it can be used for others ones. In this first approach, the association of the objects with places is binary, that is an object belongs or does not belong to some place. However, the association can be handled with a priori probability. For instance, $desktop_{45}$ belongs to a laboratory with 60% likelihood.

The Smart Environmental Architecture for Node Localization which is based on Information Fusion and Context Awareness. Our architecture was tested under different scenarios (indoors and outdoors) considering obstacles over an occupied area. Each considered NLA was evaluated for those scenarios and then complemented by using the SBN's, the LPN and the SEA-NL. The SBN's do not benefit to the CL because it is not based on distances or angles, however the CL whether is improved by the LPN and the SEA-NL. Summarizing, our architecture improved to the NLA's used 19.14% on average.

The Context-Aware Architecture for Node localization improves the estimated position of nodes by using both the environment layout and its corresponding objects' attributes. Our proposed architecture CAA-NL provides the estimated positions of each node together with the likelihood of belonging to a known place. The CAA-NL was tested in a couple of contiguous buildings considering from 0.01 to 0.99 obstacles per square meter at random in a 2D/3D environment. CAA-LN improves ∼13.05% the accuracy of the Centroid Localization, Weighted Centroid Localization, Improved Centroid Localization, and Triangular Centroid Localization algorithms. The environment layout and object-place relation are key factors for improving the NLA accuracy. In a second scenario, the CAA-NL was extensively tested in order to determine the reduction of beacon nodes (%) that can be removed while keeping the same accuracy of the NLA. The average-largest reduction found was ∼55.33% for the WCL localization. The reduction in the number of required beacon nodes depends mainly on the random distribution of nodes, environment layout, and relation object-place. However, others factors can be considered such as the environment dimensions, coverage range of nodes, and used NLAs.

The main contributions of Part I (LOCALIZATION) are:

- Three architecture for node localization in a Wireless Sensor network:

  The Logical Position of Nodes.

  Smart Node Architecture for Node Localization.

  The Hierarchical Subsumption Architecture for Node Localization.

- Six node localization algorithms:

  The Vectorial Localization Algorithm.

  The Energized Centroid Localization.

  The Subzone Localization algorithm.

  The Triangular Centroid Localization.

  The Smart Beacon Nodes.

  The Logical Position of Nodes.

- A novel solution for node localization considering the variations of the energy consumption of nodes.

- A research subarea focused on developing NLAs with an execution stamp n.

- The use of context awareness for improving the estimated position of a node.

- We proved that for the studied NLAs the number of required beacon nodes can be highly reduced by the use of context awareness without affecting their average accuracy (it is believed that this is also true for similar NLAs to the studied ones).

- We successfully introduced the subsumption principle to the area of context aware computing for a localization process.

PART II

TRACKING

Chapter 7

Localization of a Mobile Node in Shaded Areas

Our proposed mobile node architecture addresses the tracking of a mobile node considering: (i) a main node tracking system/algorithm that is feasible enough for non-shaded areas and (ii) a subsystem that supports the node tracking in shaded areas. A vehicle with a GPS on-board is a suitable example for (i); nevertheless our approach does not consider any particular main system/algorithm. Instead, our proposed PRMM (*Probabilistic Random Mobility Model*)) generates sequential location points, trajectory, based on the CTR (*Center Turning Radius*) that in turn is both an inherent vehicular feature [29] and a vehicular displacement restriction. On this basis alone diverse mobile nodes can be represented, for instance a utility car with $CTR = 6.4$ meters or smaller values for robots. Our proposed model neither is limited to simulate vehicle trajectories nor tries to replicate the driving style of a person.

As secondary system/algorithm, we adopt the well-known particle filtering approach because it is able to handle the location uncertainty during a shaded area and improves the node location estimation over the time. We propose not only a solution for locating a mobile node in shaded areas but also a Probabilistic Random Mobility Model for generating of paths, both based on CTR.

The proposed architecture for tracking a mobile node is made up of three main modules: (i) Probabilistic Random Mobility Model, (ii) Location Subsystem, and (iii) Priority Suppress as shown in Fig. 7.1. In (i), several parameters can be established in order to obtain sequential location points (l_i) and observations (Y_i) at time i. The four submodules Timer, Initial Parameters, Probability, and Noise are given in greater detail in section 7.1. In (ii), Y_i is used as a unique input for generating an alternative estimated position $E[x_i]$ based on particle filtering. The Location Subsystem is given in greater detail in section 7.2. In (iii), a selective process based on priority and availability is performed in order to select l_i when available and $E[x_i]$ during shaded areas such that

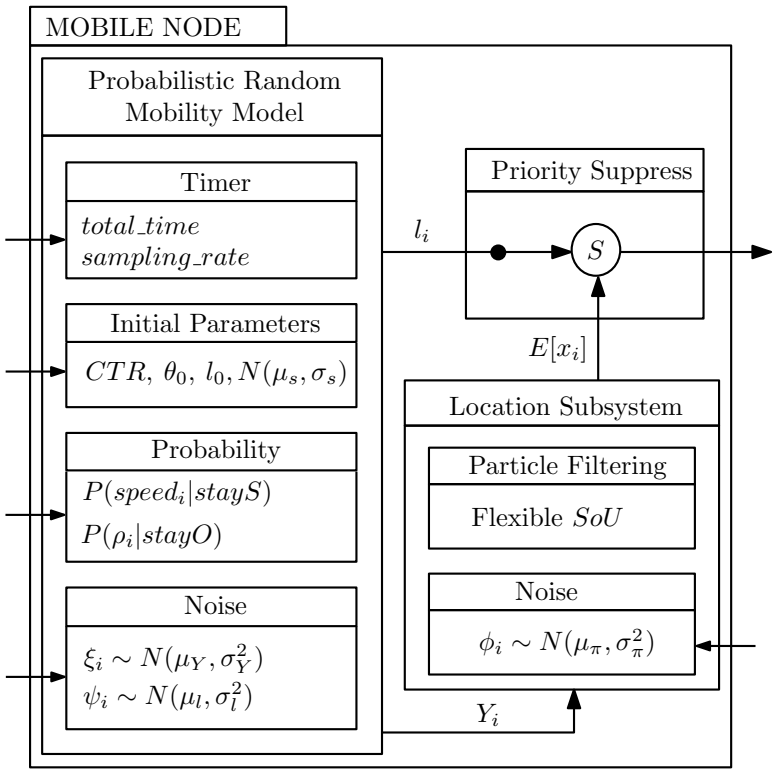


Figure 7.1: General Architecture for mobile node tracking in shaded areas.

the black dot over the line l_i represents the highest priority.

7.1 Probabilistic Random Mobility Model

Our Probabilistic Random Mobility Model generates sequential points, trajectories, of l_i and Y_i in a two dimensional Cartesian space (x, y) such that units are expressed in meters. The sequential points are generated from: (i) the bounded displacement based on the mobile node CTR, (ii) probability of turn every amount of seconds, (iii) probability of increase/decrease the speed every amount of seconds, and (iv) noise based on Normal Distribution.

Algorithm 9 shows the procedure for generating of l_i and Y_i. Let *sampling_rate* be the number of samples per second, i be a time slot s.t. $i = 1\ second/sampling\_rate$, and *total_time* be the total time of simulation. All time variables are handled in the PRMM *Timer* module as shown in Figure 7.1. The default initialization ($i = 0$) of the mobile node variables from step 1 to 5 is contextualized as follows: A utility car ($CTR = 6.4$) starts its movement at the position $(0, 0)$ with an orientation θ_0 which is randomly chosen from 0 to 2π radians. The $speed_0$ and threshold thr are established from the assumption that $speed_i \sim N(\mu_s, \sigma_s^2)$ with a Normal distribution based on the central limit theorem.

The use and definition of *thr* are addressed in the step 16 of the Algorithm 9.

Algorithm 9 Generating of trajectories from the vehicle CTR.

Require: $i \geq 0$, $total\_time > 0$, $sampling\_rate > 0$
Ensure: $l_i, speed_i, \rho_i, \theta_i, Y_i$
1: $CTR \leftarrow 6.4$ *or given*
2: $l_0(x, y) \leftarrow (0, 0)$ *or given*
3: $\theta_0 \leftarrow Random[0, 2\pi)$ *or given*
4: $speed_0 \leftarrow \mu_s$
5: $thr \leftarrow \mu_s + \sigma_s^2/2$
6: **for** $i = 1$ **to** $total\_time \times sampling\_rate$ **do**
7: **if** $i \; mod \; sampling\_rate \; mod \; secondS = 0$ **then**
8: $speed_i \leftarrow A(speed_{i-1}, P(speed_i|\cdot), \mu_s, \sigma_s^2)$
9: **else**
10: $speed_i \leftarrow speed_{i-1}$
11: **end if**
12: $displacement_i \leftarrow speed_i/sampling\_rate$
13: $b_i(x) \leftarrow l_{i-1}(x) + displacement_i \times cos(\theta_{i-1})$
14: $b_i(y) \leftarrow l_{i-1}(y) + displacement_i \times sin(\theta_{i-1})$
15: **if** $i \; mod \; sampling\_rate \; mod \; secondO = 0$ **then**
16: $\rho_i \leftarrow B(CTR, P(\rho_i|\cdot), thr, displacement_i)$
17: **end if**
18: **if** $\rho_i \neq 0$ **then**
19: $l_i \leftarrow C(|\rho_i|, l_{i-1}, b_i)$
20: **else**
21: $l_i \leftarrow b_i$
22: **end if**
23: $Y_i \leftarrow D(l_i, \mu_Y, \sigma_Y^2)$
24: $l_i \leftarrow E(l_i, \mu_l, \sigma_l^2)$
25: **return** l_i, Y_i
26: **end for**

The PRMM update is based on the smallest time slot i.e., the time variable i. Hence, the quantity of l_i and Y_i is calculated by $total\_time \times sampling\_rate$, (Step 6). The mobile node speed can be changed every $secondS$ second through the probabilistic function $A(\cdot)$ such that $secondS = 5$ is the default value, (Steps 7-11). An increase and decrease in the mobile node speed have the same probability of occurrence and it is expressed in equation 7.1 such that $stayS = 0.8$ is the default value.

$$P(speed_i|stayS) = \begin{cases} stayS, & speed_i = \mu_s \\ \frac{1-stayS}{2}, & otherwise \end{cases} \tag{7.1}$$

The speed assignment by equation 7.2 is based on the three sigma rule such that j and i are two uniform random numbers in the interval $[0, 1]$ and $g = 1 - stayS$.

$$A(\cdot) = \begin{cases} \mu_s, & i = [0, stayS) \\ \mu_s + j\sigma_s^2, & i = \Big[\ stayS, stayS + 0.341g\ \Big) \\ \mu_s - j\sigma_s^2, & i = \Big[\ stayS + 0.341g, stayS + 0.682g\ \Big) \\ \mu_s + \sigma_s^2 + j\sigma_s^2, & i = \Big[\ stayS + 0.682g, stayS + 0.818g\ \Big) \\ \mu_s - \sigma_s^2 - j\sigma_s^2, & i = \Big[\ stayS + 0.818g, stayS + 0.954g\ \Big) \\ \mu_s + 2\sigma_s^2 + j\sigma_s^2, & i = \Big[\ stayS + 0.954g, stayS + 0.976g\ \Big) \\ \mu_s - 2\sigma_s^2 - j\sigma_s^2, & i = \Big[\ stayS + 0.976g, 1\ \Big] \end{cases} \tag{7.2}$$

Based on the fact that if the mobile node speed is greater than zero a sudden displacement in the opposite direction might not be possible. On this basis alone, l_i can be established in a well-bounded space from CTR. Let the Fig. 7.2 takes place in order to illustrate the bounded displacement of a mobile node based on its CTR as well as the rest of the Algorithm 9.

The dotted arc represents all possibilities $l_i(x, y)$ for a mobile node based on its CTR. If the mobile node goes in a straight line, it would stay at $b_i(x, y)$ which is determined from $l_{i-1}, displacement_i$, and θ_{i-1}, (Steps 12-14). If turning, the position $b_i(x, y)$ is then displaced over the dotted arc ρ radians.

The mobile node turning can be changed every *secondO* seconds through the probabilistic function $B(\cdot)$ such that $secondO = 7$ is the default value, (Steps 15-17). A left and right turn have the same probability of occurrence and *stayO* represents the probability of non-turning. Hence, the turning probability is $1 - stayO$ and this is expressed by equation 7.3, where $stayO = 0.8$ as the default value.

$$P(\rho_i | stayO) = \begin{cases} stayO, & \rho_i = 0 \\ \frac{1-stayO}{2}, & otherwise \end{cases} \tag{7.3}$$

The maximum turning of a mobile node is bounded by the angle λ_i which is derived from CTR, $l_{i-1}(x, y)$, and $b_i(x, y)$. The angular velocity w_i can be simplified as expressed in equation 7.4 since the formed triangle is right and $displacement_i = speed_i / sampling\_rate$. The angle α_i can be then easily estimated by equation 7.5.

$$w_i = \frac{displacement_i}{CTR} \tag{7.4}$$

$$\alpha_i = \frac{\pi - w_i}{2} \tag{7.5}$$

The angle λ_i is reduced as the mobile node increases its speed in order to avoid skidding or overturning. The safe range of turning can be calculated considering mobile node mass, friction, sensor measurements, and forces involved, among others. However,

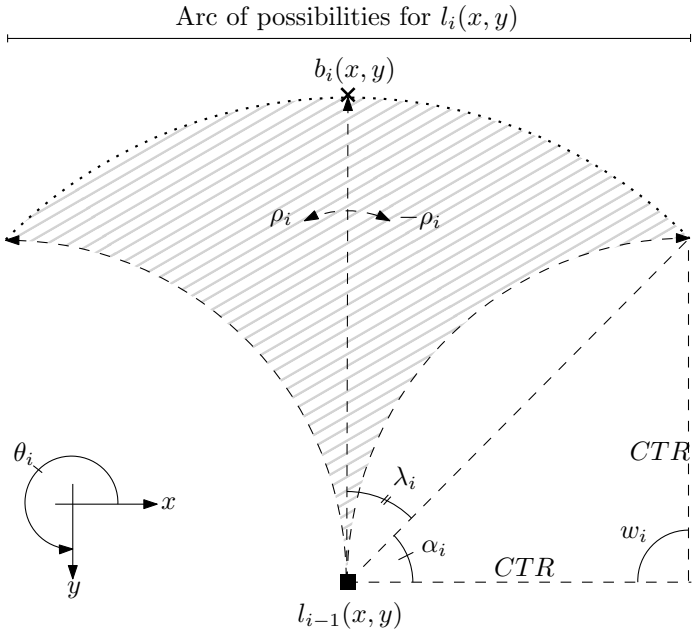


Figure 7.2: Arc of possibilities for $l_i(x,y)$ for a mobile node ■ based on the vector rotation (l_{i-1}, b_i) such that its magnitude is $\sqrt{2CTR^2}$.

this calculation is beyond the scope of this article. Instead, the angle λ_i is gradually reduced while the mobile node speed is greater than the threshold thr as shown in equations 7.6 and 7.7 such that $thr = \mu_s + \sigma_s^2/2$ as the default value, (Step 5).

$$\lambda_i = \pi/2 - \alpha_i' \tag{7.6}$$

$$\alpha_i' = \begin{cases} \frac{(speed_i - thr)(\frac{\pi}{2} - \alpha_i)thr}{\mu_s + {\sigma_s}^2} + \alpha_i, & speed_i > thr \\ \alpha_i, & otherwise \end{cases} \tag{7.7}$$

The value of ρ_i is assigned by function $B(\cdot)$ as shown in equation 7.8 such that i and j are two uniform random numbers in the interval $[0, 1]$, (Step 16).

$$B(\cdot) = \begin{cases} 0, & i = [0, stayO) \\ j\lambda_i, & i = \left[stayO, \frac{stayO+1}{2} \right) \\ -j\lambda_i, & i = \left[\frac{stayO+1}{2}, stayO \right] \end{cases} \tag{7.8}$$

If there is a turning i.e., $\rho_i \neq 0$ the mobile node position l_i is then displaced ρ_i radians over the dotted arc as shown in Figure 7.2. The turning is determined by the function $C(\cdot)$ in equation 7.9 such as $dx = l_i(x) - l_{1-i}(x)$ and $dy = l_i(y) - l_{1-i}(y)$. The angle θ_i is then easily determined by using l_i and l_{i-1}. If there is no turning l_i is then equal to b_i, (Step 20-22).

$$C(\cdot) = \begin{pmatrix} cos\ \rho_i & -sin\ \rho_i \\ sin\ \rho_i & cos\ \rho_i \end{pmatrix} \begin{pmatrix} dx \\ dy \end{pmatrix} + l_{i-1} \begin{pmatrix} x \\ y \end{pmatrix} \tag{7.9}$$

In the Step 23, the observation Y_i is determined nearby of l_i by the function $D(\cdot)$ as shown in equation 7.10 such that $\xi_i \sim N(\mu_Y, \sigma_Y^2)$ with a Normal distribution based on the central limit theorem. The observation error ξ_i is calculated by equation 7.11, where $Rangle = Random[0, 2\pi)$ and $N(0.3, 0.1)$ as the default value.

$$D(\cdot) = l_i \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} \xi_i \times cos\ Rangle \\ \xi_i \times sin\ Rangle \end{pmatrix} \tag{7.10}$$

$$\xi_i = \frac{speed_i \times \mu_Y}{sampling\_rate} + Random[-\sigma_Y^2, \sigma_Y^2] \tag{7.11}$$

Finally, in the step 24 the noise ψ_i is added to the sequential point l_i in a similar process to the process performed in the step 23 such that $\psi_i \sim N(0.1, 0.05)$ as the default value.

The Figure 7.3 shows ten paths created by our PRMM. They were created with the same initial values except the dash one. The latter has the constant value of ρ_i equal to -0.02, (Step 16). In other words, the mobile node is turning to the right 0.02 radians every *secondO* seconds. Similar action can be realized for the mobile speed in step 8. The distance and orientation between points l_i are determined from the probability of randomly change both the mobile node speed and orientation. However, predefined values into the Probabilistic Random Mobility Model allow the creation of deterministic and semi-deterministic paths.

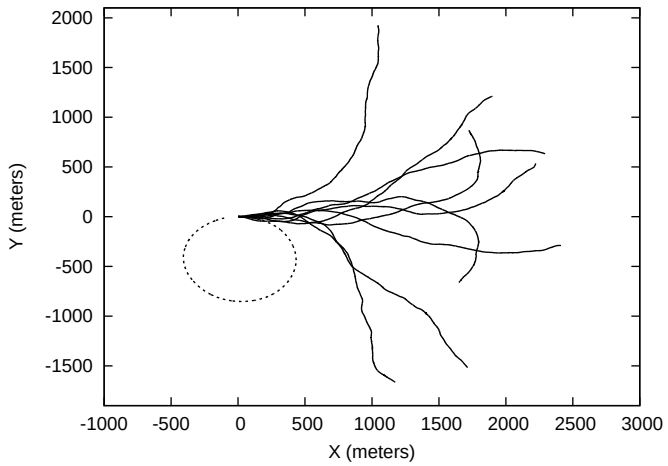


Figure 7.3: Several paths created by PRMM starting at the coordinate (0,0).

As bottom line, the large diversity of generated paths by PRMM allows our proposed Location Subsystem module be tested with many and diverse scenarios.

7.2 Location Subsystem

We define SoU (*Space of Uncertainty*) as the space where a mobile node might stay at a given time instance. We define also Flexible *SoU* as the *SoU* resizing over the time. In Figure 7.2, suppose that $|b_i - l_{i-1}|$ is the maximum mobile node displacement. The rising tiling pattern represents then *SoU* at the instance i. In other words, the arc of possibilities for $l_i(x, y)$ is forward projected regarding the maximum mobile node displacement. Hence, the pseudo-cone shape is maintained despite variations in speed. The Flexible Space of Uncertainty can be expressed as a 4-tuple $(E[x_{i-1}], \theta_i, CTR, maxD)$ such that $E[x_{i-1}]$ is the expected value in the previous instance, θ_i is the standard angle, and $maxD$ is the maximum mobile node displacement $(\mu_s + \sigma_s \times 3)/sampling\_rate$.

As far as we know, in the-state-of-the-art works the space of uncertainty of a mobile node is represented by a circle or square. However as stated earlier, considering the mobile node status and its inherent features the space can be significantly reduced. For example in a noise-free environment, let a circle with radius equal to 6.4 m be the uncertainty circle. The area where a mobile node might stay is 128.68 m^2 and the area of *SoU* is 13.99 m^2. In other words, there is a reduction of 89.13% in the area, which can be even greater as the mobile speed increases after the threshold thr.

In the Location Subsystem module, we have successfully introduced the Flexible *SoU* into Particle Filtering. Our proposed algorithm in this module adopts the general framework Sequential Importance Resampling in order to approximate the posterior probability distribution $p(x_i|Y_{0:i})$ by a weighting set of N particles. The calculation of the distribution is performed recursively using a Bayes filter and assuming that the Markov property is holds.

Algorithm 10 shows the procedure to compute the expected mobile node position $E[x_i]$. Initially, the Location Subsystem has no knowledge about its position. Hence, N particles are drawn randomly around the expected value $E[x_0]$ i.e., at any possible location. In step 2, the probability for each particle w_0^p is uniformly established because there is no prior information. In step 3, the expected value $E[x_0]$ is determined by a centroid estimation because all particles have the same weight. In step 6 x_i^p is computed using x_{i-1}^p and Y_i because x_i^p reflects all previous observations. Hence, the Probabilistic Random Mobility Model plus noise ϕ_i is used as the importance function π such that $\phi_i \sim N(0.3, 0.1)$ as default value. In step 7, the assigned weights w_i^p are normalized in

Algorithm 10 Particle Filtering + Flexible SoU.

Require: Y_i

Ensure: $E[x_i], x_i^p$

{INITIALIZATION, $i = 0$}

Draw particles N around expected value $E[x_0]$

Uniform weighting. $w_0^p = 1/N$.

$E[x_0] \leftarrow \sum_{p=1}^{N} x_0^p \times w_0^p$

loop

  {PREDICTION, $i > 0, \forall p$}

  $\pi(x_i^p | E[x_{i-1}^p], Y_i, \phi_i)$

  $w_i^p \leftarrow P(Y_i | x_i^p) = 1 - \frac{|Y_i - x_i^p|}{\sum |Y_i - x_i^p|}$

  $E[x_i] \leftarrow \sum_{p=1}^{N} x_i^p \times w_i^p$

  {FILTERING}

  while $x_i^p \notin SoU_i$ **do**

    Resampling($x_i^p | E[x_i], SoU_i$)

  end while

  $w_i^p \leftarrow 1 - \frac{|E[x_i] - x_i^p|}{\sum |E[x_i] - x_i^p|}$

  $\tilde{w}_i^p \leftarrow w_i^p \times w_{i-1}^p$

  $w_i^p \leftarrow \frac{\tilde{w}_i^p}{\sum_{p=1}^{N} \tilde{w}_i^p}$

  { ESTIMATION }

  $E[x_i] \leftarrow \sum_{p=1}^{N} x_i^p \times w_i^p$

  return $E[x_i]$

end loop

the range $[0, 1]$ based on the distance between Y_i and x_i^p. In step 8, the expected mobile node position $E[x_i]$ is determined by all particles and their corresponding weights such that $\{(x_i^p, w_i^p) | \, p \in [1, N], \sum_{p=1}^{N} = 1\}$.

Through steps 9-11, particles x_i^p with negligible weights are replaced by new particles with higher weights around $E[x_i]$ and within SoU_i. In the steps 12-14, the weights are updated to $\tilde{w}_i^p$ based on previous valid possible locations and $\tilde{w}_i^p$ is normalized to w_i^p. In the step 15, the estimated mobile node position, $E[x_i]$, is calculated based on the posterior distribution which is represented by the weighted set (x_i^p, w_i^p). Finally, $E[x_i]$ is provided to the Priority Suppress module.

7.3 Priority Suppress

We successfully introduced the suppress principle of the subsumption architecture [28] into our proposed architecture. The principle has been widely employed in robotics and software agents because complex behaviors can be divided into simple modules that in turn are organized into layers. Information of a upper layer can subsume information of lower layers and this is graphically represented by the symbol "s", as shown in Figure 7.1. In our proposed architecture the upper layer is identified by the black dot over the line l_i. In other words, during non-shaded areas the estimated position l_i by the main node tracking system subsumes the expected value $E[x_i]$ calculated by the Location Subsystem.

Chapter 8

Experimental Evaluation and Results

In this chapter, we describe the experiments we conducted to measure the effectiveness of our proposed architecture for a mobile node localization in shaded areas. In particular, we are interested in the Location Subsystem evaluation considering the space of uncertainty as a circle and pseudo-cone shape. The former alludes to the most used form in the node tracking based on Particle Filtering and the latter refers to our proposed Flexible SoU. We use the term *General PF* to refer to the use of a circle as space of uncertainty and *PF + Flexible SoU* to refer to our proposal.

Hereafter, all simulation results are based on the default values shown in Table 8.1 unless indicated otherwise.

In Figure 8.1 is shown the particle behavior of General PF and PF + Flexible SoU such that $N = 50$, $\theta_0 = 0$, and $i = 4$. The mobile node is represented by the large circle where arrows indicate its trajectory. The General PF particles are grouped around the mobile node at the current position (3.33, 0). However, 20% of the particles are behind the mobile node at the position $i - 1$. Based on the fact that if the mobile node speed is greater than zero a sudden displacement in the opposite direction might not possible. Hence, only 80% of the particles must be considered as valid. Moreover, considering the vehicle's CTR even more particles might be removed. On the other hand, the particles of our proposed PF + Flexible SoU are drawn regarding the bounded-forward displacement of the mobile node and they are grouped so that a pseudo-cone shape is formed. Besides, there is no a particle behind the mobile node at the position $i - 1$.

The pseudo-cone form is observed even for a higher speed as long as it is less than the threshold thr. For example, Figure 8.2 depicts the particle behavior when the mobile node speed is greater than thr which changes the opening of the pseudo-cone form making it seem as a pseudo line.

The sampling rate determines the update of $E[x_i]$, which can increase the local-

Table 8.1: Considering default values as the base scenario.

	VALUE
$total\_time$	18000 seconds
$sampling\_rate$	4 per second
CTR	6.4 meters
θ_0	$Random[0\pi, 2\pi)$
l_0	$(0, 0)$
$speed_i$	$N(8.33, 2.77)$
$P(speed_i \mid stayS)$	$stayS = 0.8$
$secondS$	5 seconds
$P(\rho_i \mid stayO)$	$stayO = 0.8$
$secondO$	5 seconds
ξ_i	$N(0.3, 0.1)$
ψ_i	$N(0.1, 0.05)$
ϕ_i	$N(0.3, 0.1)$
N	25 particles
Markov chain size	50 states

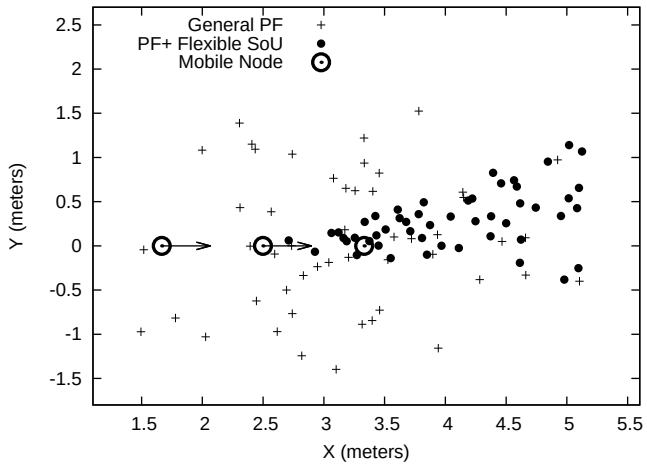


Figure 8.1: The behavior of the set of particles as a pseudo-cone form.

ization error of Location Subsystem module. We established $sampling\_rate = 4$ as the default value because for both General PF and PF + Flexible SoU the average estimation error is kept below of one meter as shown in Figure 8.3. A higher value for $sampling\_rate$ involves a higher execution frequency of Location Subsystem module and therefore a higher energy consumption. However, it is not a guarantee that the error will be less than one meter. For example, a higher mobile node speed increases the distance separation among measurement points so that the error for $sampling\_rate = 4$ and $speed = 41.66$ is similar to $sampling\_rate = 1$ and $speed = 11.11$ as shown in Figure 8.4 where (n) represents the value for $sampling\_rate$.

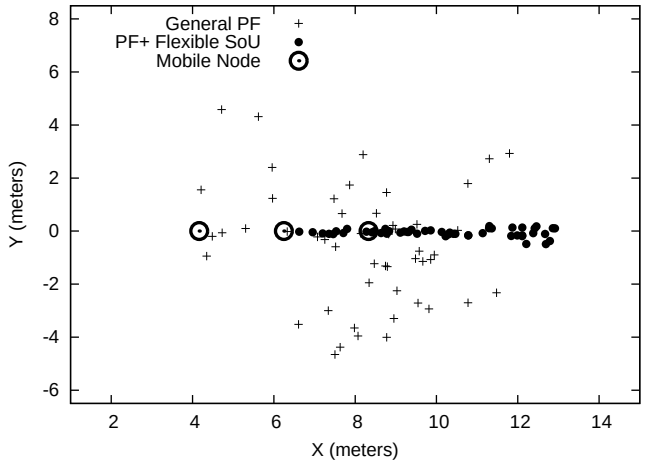


Figure 8.2: The behavior of the set of particles as a pseudo-line form.

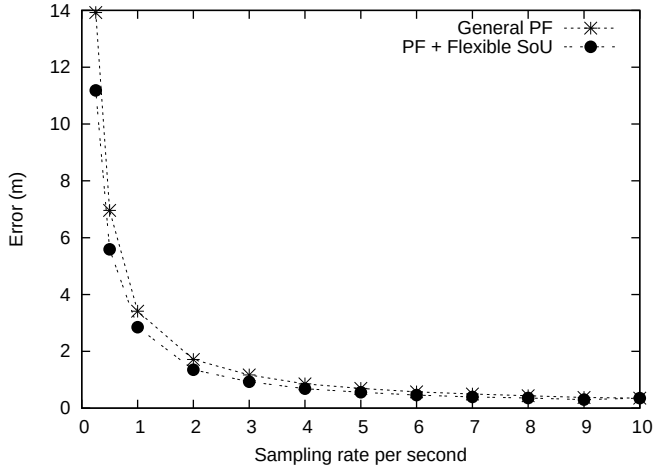


Figure 8.3: The Location Subsystem performance over different sampling rate.

During the experiments carried out in Figure 8.4 the value of σ_s^2 was kept to 2.77 m/s and μ_s was gradually increases in units of 2.77 m/s staring at 2.78 m/s. Obviously, the probability of changing the speed also affects the accuracy, but it is a PRMM property used in combination with other parameters to generate diversity in trajectories. Hence, the probability is not considered as an element of the Location Subsystem module but rather the recorded states of the mobile node using recursively a Bayes filter and assuming that the Markov property is holds.

On the other hand, the number of particles handled is another key factor. Maintaining a large number of particles can improve the accuracy, but requires additional memory and increases the execution time because the particle matrix size is determined by the Markov chain size (states) and number of particles. Figure 8.5 shows the Location Subsystem module accuracy under different number of particles, where (n) represents the

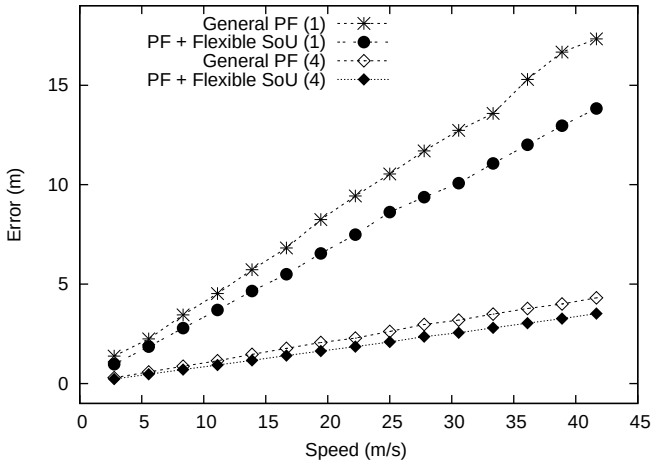


Figure 8.4: Scenario 3. The behavior of the set of particles under the speed changing

value for *sampling_rate*. The error is quickly reduced as the number of particles increases, but it is fairly stable after 25 particles. The accuracy improvement for $25 < N < 501$ is negligible.

The particle movement from the previous to the current state is carried out by the importance function π in which the noise ϕ allows particles effervesce. But, a high noise can make particles go far away and causes a lot of resampling. We found that $\phi \sim N(0.3, 0.1)$ has the best performance and PF + Flexible *SoU* dampens better high values in μ_π than General PF as show in Figure 8.6, where (n) represents the value for *sampling_rate*.

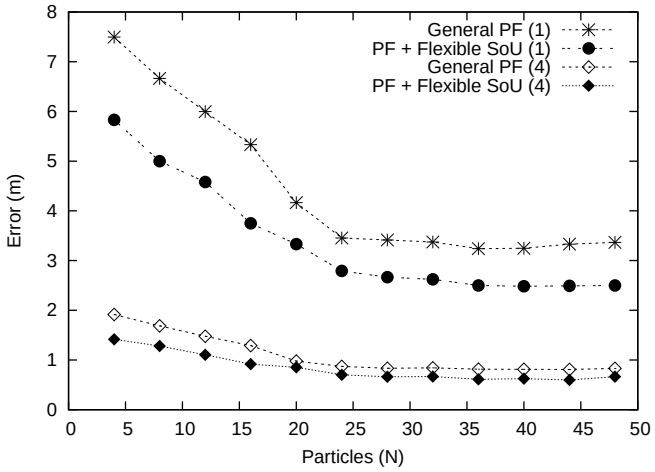


Figure 8.5: Impact of Particle Set size.

Based on our experiments, sampling rate, speed, number of particles, and noise (ξ, ψ, and ϕ) are the most relevant accuracy elements. Both General PF and PF + Flexible

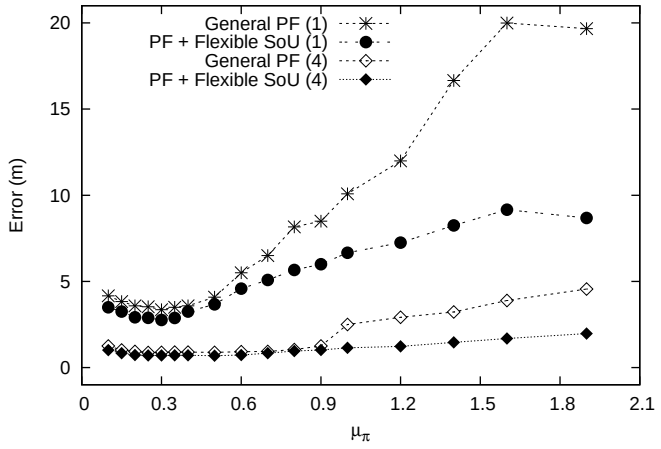


Figure 8.6: Noise over the set particle such that $\sigma_\pi^2 = \mu_\pi/3$.

SoU are fairly stable for different values in CTR, $P(\rho_i|stayO)$, and $P(speed_i|stayS)$. The change frequency in the mobile orientation, *secondO*, does not have an important impact in the accuracy because there is a bounded-forward displacement based on CTR. The change frequency in the mobile speed, *secondS*, can have an impact from one time instance to another, but the average error tends to be almost the same for both a low and high value in *secondS*.

Chapter 9

Ongoing Work and Conclusion

We presented not only a solution for locating a mobile node in shaded areas but also a Probabilistic Random Mobility Model for generating of paths, both based on CTR. Our Mobility Model is able to generate mobile node paths including the probability of change both the speed and orientation in a bounded-forward displacement. It is believed that a better paths can be achieved by incorporating more kinetically elements in the Model, but it is beyond the scope of this article. Our main goal was to evaluate the Location Subsystem module performance under diversity of scenarios considering the space of uncertainty as a circle and pseudo-cone shape (Flexible SoU). The latter is used as part of our solution for mobile localization in shaded areas, which reduces the space at least 89.13%. Nevertheless, the space reduction is not kept in the same proportion for accuracy reduction between General Particle Filtering and our proposed PF + Flexible SoU because there are many independence factors in the Mobility Model and Location Subsystem modules.

The main contributions of Part II (TRACKING) are:

- An architecture for locating of a mobile node during shaded areas.
- We successfully introduced the suppress principle of the subsumption architecture [28] into our proposed architecture as a priority-selective control between l_i and $E[x_i]$.
- A Probabilistic Random Mobility Model for generation of paths of any vehicle [29].
- We successfully introduced the Flexible SoU upon the area of Particle Filtering and generation of paths.
- The use of the CTR and status of a mobile node, in combination, to reduce and adapt the space of uncertainty.

Appendix A

Real vehicle Tracking

A.1 Introduction

The rapidly changing landscape, lack of navigation planning, and increasing traffic are some of the factors that are fueling a rising need for highly efficient navigation and allied services for the masses. An efficient navigation system can also act as the backbone for other services to be delivered along an active route the user is traveling though - what are known as vertical LBS (*Localization Based Services*). Live traffic information, local search, permissive local advertisements, mobile contact trackers and SOS are some examples of vertical LBS.

One of the developments in the LBS industry has been the emergence of technologies that have demonstrated a viable solution without needing a GPS device. Operators themselves have been trying for more than half a decade to use dynamic location awareness to provide customized mobile services to consumers, but there has nevertheless been a noticeable delay in bringing efficient location-based services over mobile devices.

Today, such services are often via Web browser and hence considered as Web services. The additional challenges to be considered are the richness, personalization and ubiquity of services to mobile user, and the linking of services to a relevant context. Therefore, another challenge has been the lack of understanding among application developers about what makes a location-based service appealing to common person. Systems that can deliver intelligent information in relation to the context of the user's location awareness simply do not exist. The entire LBS branch has been revolving around harnessing the value out of the users' location awareness and not the location's context awareness. This is a fact that is confirmed by the research team under the future computing environments at Georgia Tech, which is dedicated to the invention of novel application s using context-aware computing technology to assist everyday activities. This team admitted

that the majority of context aware computing is restricted to location-aware computing for mobile applications and not the location-context-aware computing.

Nowadays there are many vehicle owners that have adopted the GPS as tool; for instance, for getting the best path for some destination. But, the GPS still is not able to work under all possible situations, due to the need of line-of-sight with the satellites. To solve this issue, we present in this Predoctoral work the vehicle localization in latitude and longitude terms when the GPS on-board does not work by using external acceleration reading which are not influenced directly by tire skid or velocity number of the car.

A.2 General problem

The GPS needs line-of-sight with the satellites in order to estimate the node position. The external sensor used to assistance the GPS are quite affected by the environment noise and the vehicle movements.

A.3 Objective

- To keep the vehicle position although the GPS on-board is in shown areas by using an external sensor.
- To express the vehicle position at latitude and Longitude terms.

A.4 Vehicle localization

The hardware used to develop this system is GPS eTrex Vista H of Garmin [see Appendix C] and sensor OS5000 of Oceanserver [see Appendix C]. In the Figure A.1 is shown the information flow. In the box Readings the sensor OS5000 provides the acceleration in three axes and the GPS gives the localization car in terms of latitude and losale menorngitude. But, the output line of the GPS represents the information as non-available all the time due to shaded areas.

The information provided by the box Readings to box Conversion cannot be used directly. The process calculus of Conversion converts GPS outputs in acceleration terms and vice versa for Sensor outcomes. The GPS readings from time t and time $t+1$ are represented as displacement in the plane x (The Ecuador line) and in the plane y (Meridian of Greenwich) as is shown in Equation (A.1) and Equation (A.2) respectively.

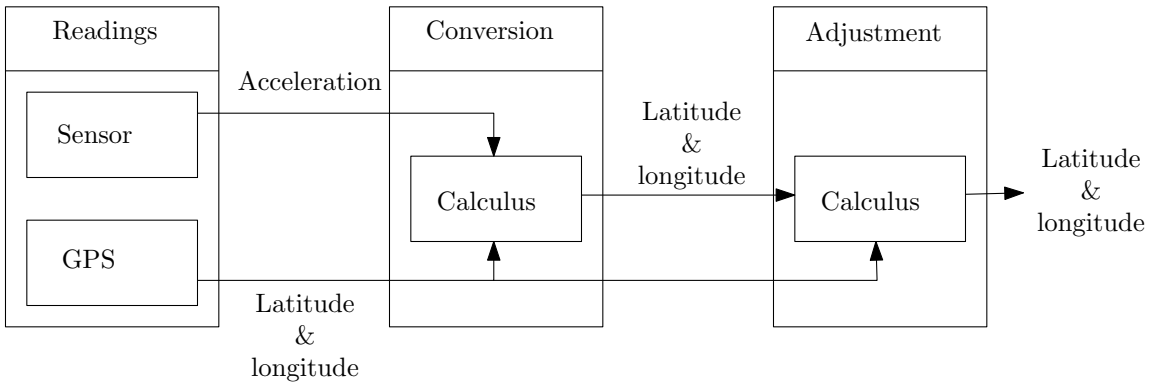


Figure A.1: Diagram flow.

$$gps_x = \frac{(gps_{long2} - gps_{long1})cos(gps_{lat1})\pi}{180} \tag{A.1}$$

$$gps_y = \frac{(gps_{lat2} - gps_{lat1})\pi}{180} \tag{A.2}$$

For expressing the variables gps_x and gps_y in distance terms, in Equation (A.3) and Equation (A.4) are adjusted by multiplication of β (6378000) which is the circumference of the world in meters. In Equation (A.5) is estimated the distance from t to $t+1$ by using Equation (A.3) and Equation (A.4).

$$gps_{dx} = gps_x\beta \tag{A.3}$$

$$gps_{dy} = gps_y\beta \tag{A.4}$$

$$gps_{dxy} = \sqrt{{gps_{dx}}^2 + {gps_{dy}}^2} \tag{A.5}$$

In Equation (A.8) is gets the velocity from t to t+1 through the use of Equation (A.6) and Equation (A.7), which estimate the velocity in plane x and y. However, this conversion is just an approach. The maximum estimation error in our simulation is 13.9907 meters and a minimum estimation error of 0.9642 meters when the GPS always has line-of-sight with satellites.

$$gps_{Vfx} = \frac{gps_{dx}}{t} \tag{A.6}$$

$$gps_{Vfy} = \frac{gps_{dy}}{t} \tag{A.7}$$

$$gps_{Vfxy} = \sqrt{{gps_{Vfx}}^2 + {gps_{Vfy}}^2} \tag{A.8}$$

At this point is possible to express the sensors reading in latitude and longitude by supporting GPS data conversion. When GPS is available are used Equation (A.9) and Equation (A.10) for getting velocity in the plane x and y respectively. Otherwise, the estimated velocity by sensor readings in time $t + 1$ are used for getting acceleration in time $t + 1$ as indicated Equation (A.11) and Equation (A.12).

$$sensor_{Vfx} = A_{fx}t + gps_{Vfx} \tag{A.9}$$

$$sensor_{Vfy} = A_{fy}t + gps_{Vfy} \tag{A.10}$$

$$sensor_{Vfx} = A_{fx}t + sensor_{Vfx} \tag{A.11}$$

$$sensor_{Vfy} = A_{fy}t + sensor_{Vfy} \tag{A.12}$$

Either Equation (A.9) and Equation (A.10) or Equation (A.11) and Equation (A.12); the Equation (A.13) is used for getting the velocity in time $t + 1$.

$$sensor_{Vfxy} = \sqrt{{sensor_{Vfx}}^2 + sensor_Vfy^2} \tag{A.13}$$

The Equation (A.14) and Equation (A.15) provide the displacement in plane x and the plane y respectively.

$$sensor_{dx} = sensor_{Vfx}t \tag{A.14}$$

$$sensor_{dy} = sensor_{Vfy}t \tag{A.15}$$

When GPS readings are available are used Equation (A.16) for obtaining the longitude and Equation (A.17) for latitude, otherwise Equation (A.18) and Equation (A.19).

$$sensor_{long2} = \frac{180 sensor_{dx}}{cos(gps_{lat1})\pi} + gps_{long1} \tag{A.16}$$

$$sensor_{lat2} = \frac{180 sensor_{dy}}{\pi} + gps_{lat1} \tag{A.17}$$

$$sensor_{lat2} = \frac{180 sensor_{dx}}{\beta \pi cos(sensor_{lat1})} + sensor_{long1} \tag{A.18}$$

$$sensor_{lat2} = \frac{180 sensor_{dy}}{180 \beta \pi} + sensor_{lat1} \tag{A.19}$$

Last equations are used in sequential form without considering previous results in state calculus. For adjusting the latitude and longitude in a flooding way with all equations, in the box adjusted the state calculus2 from Figure A.1 adds one variable for all couple of result of GPS and sensor. For instance, $gps_{lat2} \approx sensor_{lat2} \times A$; then, $a = \frac{gps_{lat2}}{sensor_{lat2}}$. Thus, the weights are distributed. $A = (A + A^*)/2$, where A^* is the value in the time t and A is the value in time $t + 1$. The system through calculus2 tries to optimize the Sensor estimation with the GPS results.

A.5 Preliminary result (GPS-Accelerometer-Vehicle)

We have established two scenarios, the first one has been done in streets of the Curitiba city and the second one was a simulation considering a bigger distance. For the first scenarios, the GPS and sensor were incorporated into the car in the top of the in front panel [see Appendix C]. The initial point (A) was at the latitude -25.45327 and the longitude -49.25005 and second point (B) was at the latitude -25.471702 and the longitude -49.261766 as shown in Figure A.2, a total of 2.6 kilometers.

The scenario considers abrupt stops. In Figure A.3 is shown the estimated acceleration readings of sensor and adjusted acceleration sensor (calculus2) when the GPS is outside of shaded areas.

At first glance, the acceleration sensor appears like a constant; however it has many variationsas is illustrates in Figure A.5.

In Figure A.5 are shown the longitude estimation of sensor (calculus) and adjusted sensor (calculus2), where the estimations are almost exactly. However, it is not the case for latitude as illustrated in Figure A.6.

The bigger localization errors appear in the second 123 and in the second 443, the maximum error recorded of direct conversion (calculus) is 1587.12 meters; the minimum error is 0.3647 meters; average error of 113.6617 meters; a median of 47.1350 meters and a standard deviation of 240.3355 meters. For adjusted localization (calculus2) the maximum error is 829.8699 meters; the minimum error is 0 meters; the average error of 78.3252; a median of 40.4277 and a standard deviation of 133.74 meter as is shown in Figure A.7. Should the GPS is inside shaded areas, we evaluate with stratification $k = 5$ from a set

Figure A.2: Scenario of the first test. Image from http://maps.google.com.br/

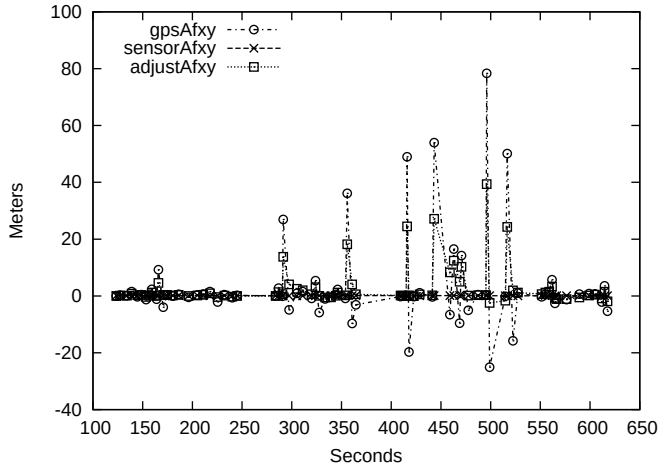


Figure A.3: Estimated Acceleration from the external sensor

of 78 instances, all them from the time 29 second to 617 seconds. As first test, instance 4,5,6,7 and 8 was set up at zero to represent GPS enable. A total of 15 combinations

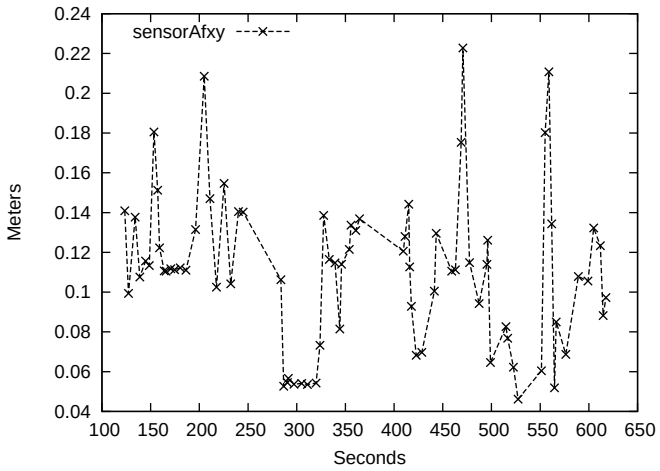


Figure A.4: Estimated Acceleration from the external sensor

Figure A.5: Comparison of Longitude.

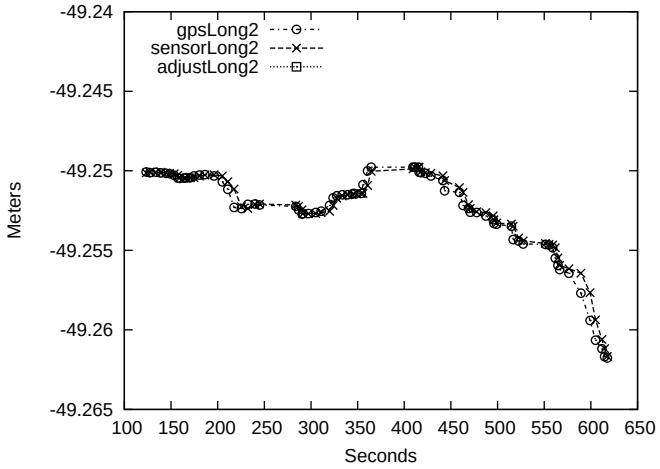
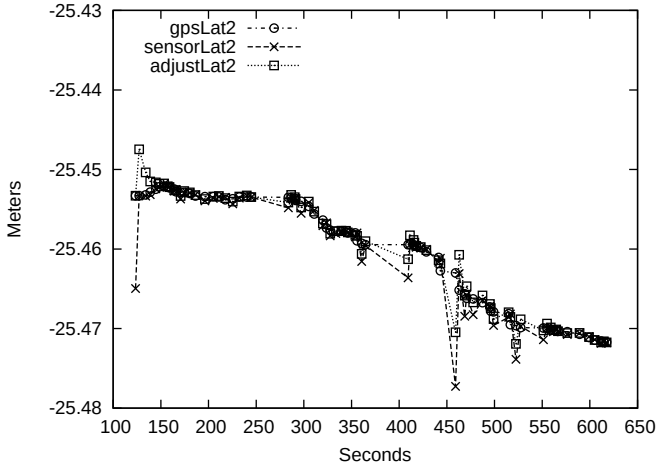
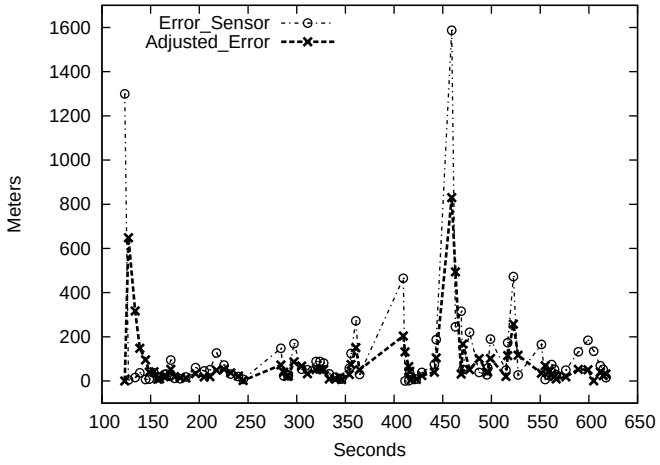


Figure A.6: Comparison of Latitude.



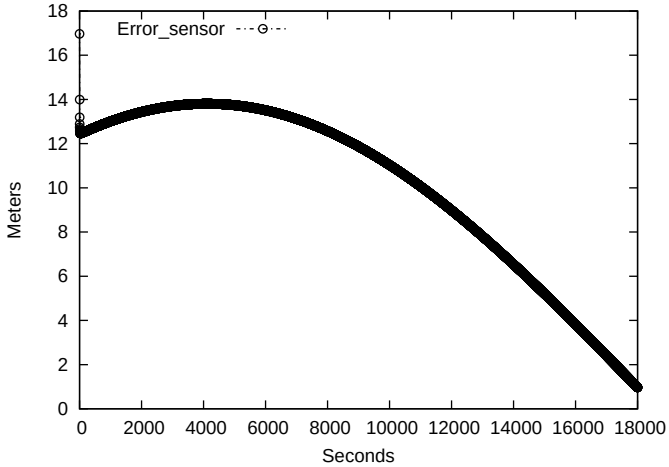
were achieved. The state calculus results had the maximum error of 1741.4703 meters and minimum error of 0.42293 meters. The error localization for adjusted calculus was 1018.48 meters and minimum error of 1.087 meters.

Figure A.7: Localization Error.



For the experimental simulation, the initial point (A) was established at the latitude -25.57471 and longitude -54.56348 and second point (B) was set up at the latitude -23.6344 and longitude -52.41682, a total of 305.470 kilometers. In Figure A.8 is depicted the estimation error by state calculus.

Figure A.8: Localization Error.



With these results we proof that the direct calculus have an inherent error which have a maximum error of 13.99 meters; a minimum error of 0.9642 meters and average error of 11.8689 meters. For adjusted calculus the error maximum is just 1 meter; minimum of is 0 meters and average error of 0.0006 meters. The GPS has been simulated in shaded

area for 509.117 meters that is from time 4999 seconds to 5029 seconds when the bigger errors appear (13.7398 meters). The maximum error of the state calculus was 425.84669 meter and the maximum error of state calculus2 was 418.9789 meters.

A.6 In front panel

Figure A.9: Testing in the vehicle. The GPS and the sensor OS5000.

A.7 Conclusion

We presented the vehicle localization when the GPS on-board fails during shaded areas by using an external sensor. The sensor readings are expressed in acceleration terms; they are converted at latitude and longitude terms by direct transformation in two-dimensional Cartesian space, where axis x represents the Ecuador and the axis y the Meridian de Greenwich.

After transformations the outcome is presented in latitude and longitude terms like GPS results. In a real test made in Curitiba city considering abrupt stops we evaluated our proposal by stratification analysis, where the maximum error was of 829.8699 meters and minimum error of 0 meters. In our simulation the maximum error was 13.99 meters. However, by introducing the adjusted function the error is decreased at 1 meter.

Appendix B

Software and Definitions

B.1 AvroraZ

AvroraZ is an extension of the Avrora emulator - The AVR Simulation and Analysis Framework - which allows the emulation of the Atmel AVR microcontroller based sensor node platforms with IEEE 802.15.4 compliant radio chips thus allowing emulation of sensor nodes such as Crossbow's MicaZ Micaz.

Avrora is a set of simulation and analysis tools for programs written for the AVR microcontroller produced by Atmel and the Mica2 sensor nodes. Avrora contains a flexible framework for simulating and analyzing assembly programs, providing a clean Java API and infrastructure for experimentation, profiling, and analysis.

AvroraZ is based on design, implementation and verification of several extensions to Avrora: the address recognition algorithm, an indoor radio model, the clear channel assessment (CCA) and link quality indicator (LQI) of the IEEE 802.15.4 standard. The motivation of this implementation is to enable precise emulation of IEEE 802.15.4 based protocols without any modifications in the code developed for the real hardware.

The tool is being tested and evaluated using the implementation of beacon-enabled mode of the IEEE 802.15.4 protocol stack developed in nesC, under the TinyOS operating system for the CrossBow MICAz motes called Open-zb as well as new add-ons to this implementation that allow mesh topology.

Source: http://citavroraz.sourceforge.net/

B.2 TinyOS

TinyOS is an open source, BSD-licensed operating system designed for low-power wireless devices, such as those used in sensor networks, ubiquitous computing, per-

sonal area networks, smart buildings, and smart meters. A worldwide community from academia and industry use, develop, and support the operating system as well as its associated tools, averaging 35,000 downloads a year.

Source: http://www.tinyos.net/

B.3 C++

C++ is a programming language that is general purpose, statically typed, free-form, multi-paradigm and compiled. It is regarded as an intermediate-level language, as it comprises both high-level and low-level language features. Developed by Bjarne Stroustrup starting in 1979 at Bell Labs, C++ was originally named C with Classes, adding object oriented features, such as classes, and other enhancements to the C programming language. The language was renamed C++ in 1983, as a pun involving the increment operator.

C++ is one of the most popular programming languages and is implemented on a wide variety of hardware and operating system platforms. As an efficient compiler to native code, its application domains include systems software, application software, device drivers, embedded software, high-performance server and client applications, and entertainment software such as video games. Several groups provide both free and proprietary C++ compiler software, including the GNU Project, LLVM, Microsoft and Intel. C++ has greatly influenced many other popular programming languages, most notably C# and Java. C++ is also used for hardware design, where the design is initially described in C++, then analyzed, architecturally constrained, and scheduled to create a register-transfer level hardware description language via high-level synthesis.

B.4 Gnuplot

Gnuplot is a portable command-line driven graphing utility for Linux, OS/2, MS Windows, OSX, VMS, and many other platforms. The source code is copyrighted but freely distributed. It was originally created to allow scientists and students to visualize mathematical functions and data interactively, but has grown to support many non-interactive uses such as web scripting. It is also used as a plotting engine by third-party applications like Octave. Gnuplot has been supported and under active development since 1986.

B.5 Java

Java is a general-purpose, concurrent, class-based, object-oriented computer programming language that is specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere" (WORA), meaning that code that runs on one platform does not need to be recompiled to run on another. Java applications are typically compiled to bytecode (class file) that can run on any Java virtual machine (JVM) regardless of computer architecture. Java is, as of 2012, one of the most popular programming languages in use, particularly for client-server web applications, with a reported 10 million users. Java was originally developed by James Gosling at Sun Microsystems and released in 1995 as a core component of Sun Microsystems' Java platform. The language derives much of its syntax from C and C++, but it has fewer low-level facilities than either of them.

B.6 RSSI

The RSSI is the abbreviation of Receive Signal Strength Indication which means Indicator Received Signal Strength. The RSSI values ranging from 0 to 255 expressed as a single byte. The maximum value of the RSSI varies by vendor e.g. Cisco cards' values ranging from 0 to 100, however Atheros cards work in values ranging from 0 to 127. On the other hand, a simulated MicaZ node in the simulator AvroraZ takes values between 105 and 255. Therefore, we cannot generalize the RSSI values. This is the reason why the IEEE 802.11 standard defines no relationship between RSSI and power level in mW or dBmM; hardware vendors must provide such data (accuracy, granularity, and ranges power).

B.7 LQI

The LQI is short for Link Quality Indicator. The LQI reports the received packet's quality considering the energy detected and the single-noise estimated rate. The values range is from 0 to 255 and a similar form to the RSSI values might vary depending on the vendor. The values captured in the simulator AvroraZ fluctuate between 45 and 103.

B.8 Weka

Weka is a collection of machine learning algorithms for data mining tasks. The algorithms can either be applied directly to a dataset or called from your own Java code. Weka contains tools for data pre-processing, classification, regression, clustering, association rules, and visualization. It is also well-suited for developing new machine learning schemes. Found only on the islands of New Zealand, the Weka is a flightless bird with an inquisitive nature. The name is pronounced like this, and the bird sounds like this. Weka is open source software issued under the GNU General Public License.

Available at: http://www.cs.waikato.ac.nz/ml/weka/

Appendix C

Hardware

C.1 MicaZ

The MICAz is a 2.4 GHz Mote module used for enabling low-power, wireless sensor networks, see Figure C.1.

Figure C.1: MicaZ mote.

C.1.1 Wireless measurement system

- 2.4 GHz IEEE 802.15.4, Tiny Wireless Measurement System.
- Designed Specifically for Deeply Embedded Sensor Networks.
- 250 kbps, High Data Rate Radio.
- Wireless Communications with Every Node as Router Capability

- Expansion Connector for Light, Temperature, RH, Barometric Pressure, Acceleration/Seismic, Acoustic, Magnetic and other Crossbow Sensor Boards.

C.1.2 Applications

- Indoor Building Monitoring and Security.
- Acoustic, Video, Vibration and Other High Speed Sensor Data.
- Large Scale Sensor Networks (1000+ Points).

C.1.3 Product features include

- IEEE 802.15.4 compliant RF transceiver.
- 2.4 to 2.48 GHz, a globally compatible ISM band.
- Direct sequence spread spectrum radio which is resistant to RF interference and provides inherent data security.
- 250 kbps data rate.
- Supported by MoteWorksTM wireless sensor network platform for reliable, ad-hoc mesh networking.
- Plug and play with Crossbow's sensor boards, data acquisition boards, gateways, and software.

enables the development of custom sensor applications and is specifically optimized for low-power, battery-operated networks. MoteWorks is based on the open-source TinyOS operating system and provides reliable, ad-hoc mesh networking, over-the-air-programming capabilities, cross development tools, server middleware for enterprise network integration and client user interface for analysis and a configuration.

C.1.4 Processor and Radio Platform (MPR2400CA)

The MPR2400 is based on the Atmel ATmega128L. The ATmega128L is a low-power microcontroller which runs MoteWorks from its internal flash memory. A single processor board (MPR2400) can be configured to run your sensor application/processing and the network/radio communications stack simultaneously. The 51-pin expansion connector supports Analog Inputs, Digital I/O, I2C, SPI and UART interfaces. These

interfaces make it easy to connect to a wide variety of external peripherals. The MICAz (MPR2400) IEEE 802.15.4 radio offers both high speed (250 kbps) and hardware security (AES-128).

C.1.5 Sensor Board

Crossbow offers a variety of sensor and data acquisition boards for the MICAz Mote. All of these boards connect to the MICAz via the standard 51-pin expansion connector. Custom sensor and data acquisition boards are also available. Please contact Crossbow for additional information.

C.1.6 Base Stations

A base station allows the aggregation of sensor network data onto a PC or other computer platform. Any MICAz Mote can function as a base station when it is connected to a standard PC interface or gateway board. The MIB510 or MIB520 provides a serial/USB interface for both programming and data communications. Crossbow also offers a stand-alone gateway solution, the MIB600 for TCP/IP-based Ethernet networks.

C.2 GPS eTrex vista H of garmin

The GPS device is shown in the Figure C.2 and its specifications are shown in the Table C.1.

Figure C.2: MicaZ mote.

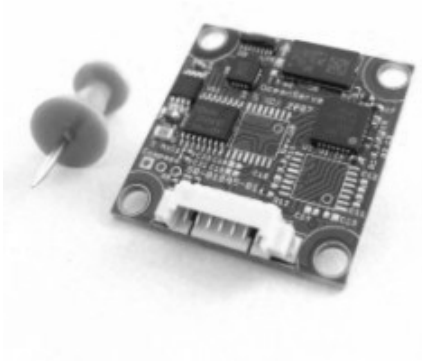
Table C.1: The GPS Specs.

Physical and Performance	
Unit dimensions, WxHxD:	2.0" x 4.4" x 1.2" (5.1 x 11.2 x 3.0 cm)
Display size, WxH:	1.1" x 2.1" (2.8 x 5.4 cm)
Display resolution, WxH:	160 x 288 pixels
Display type:	4 level gray LCD
Weight:	5.3 oz (150 g) with batteries
Battery:	2 AA batteries (not included)
Battery life:	18 hours
Waterproof:	yes (IPX7)
Floats:	No
High-sensitivity receiver:	Yes
Interface:	USB

C.3 Sensor OS5000 of oceanserver

The OS5000 families of compasses are a new class of sensor components providing best in class performance for under $200.00 (USD) in low volume. The sensor is shown in the Figure C.3.

Figure C.3: Sensor OS5000.



C.3.1 Features

- Compass accuracy, 0.5 degrees RMS heading while level, 1°RMS $< \pm 30$°Tilt, 1.5°RMS.
- $\pm$ 60° Tilt, undisturbed field. 1 Degree resolution.
- Roll & Pitch full rotation, typical 1°accuracy $< \pm 30$°tilt.
- Pitch Angles +/-90 degrees, Roll Angles +/- 180 degrees.

- Tilt-compensated (electronically gimbaled).
- Tiny size, 1"x1"x0.3", less than 2 grams weight.
- Low Power Consumption, < 30ma @ 3.3V.
- Hard and soft-iron compensation routines.
- Optional support for a high resolution Depth or Altitude sensor (24 bit AD).
- Serial Interface:
  - RS232, USB or TTL.
  - Baud rate programmable 4,800 to 115,000 baud.
- Rugged design:
  - 10,000 G shock survival.
  - -40°C to 80°C operating temperature (Accuracy specified for 0°C to 50°C).
- ASCII sentence output, in several formats, NMEA checksum.
- High Data Update Rate to 40HZ.
- Support for True or Magnetic North Output.
- Precision components:
  - 3 Axis magnetic sensors from Honeywell.
  - 3 Axis Accelerometers from ST Microelectronics.
  - 24 bit differential Analog to Digital converters from Analog Devices.
- 50 MIPS processor supporting IEEE floating point math.

Appendix D

Acknowledgement

The fourth best paper award was bestowed in the 2011 IEEE Radio and Wireless Week Sensor and Sensor Network in Phoenix, Arizona, USA on October 5th, 2011. The awarded work is titled Node localization in WSN using Trigonometric figures as shown in Figure D.1.

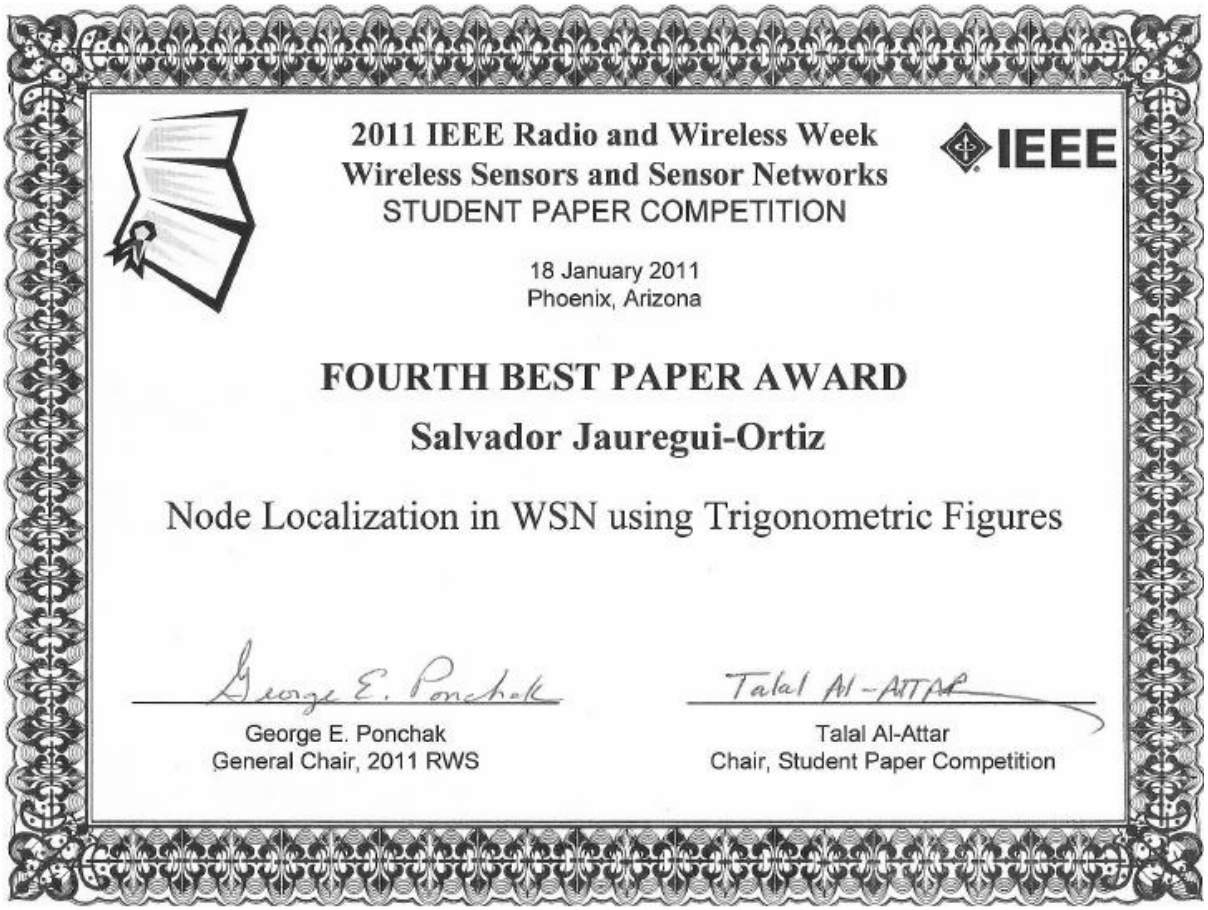


Figure D.1: Acknowledgement from IEEE.

Appendix E

Publications

2014: **Salvador Jauregui-Ortiz**, Michael Barbeau, Evangelos Kranalis, Edson Scalabrin, and Mario Siller. "Localization of a Mobile Node in Shaded Areas". Submitted in March 2014 to Sensors (ISSN 1424-8220; CODEN: SENSC9), JCR 1.953.

2014: **Salvador Jauregui-Ortiz**, Mario Siller, Félix Ramos, and Edson Scalabrin. "CAA-NL: Context-Aware Architecture for Node Localization". Finalized. Ready to submit.

2014: **Salvador Jauregui-Ortiz**, Mario Siller, and Edson Scalabrin. "Hierarchical Subsumption Architecture for Node Localization in a WSN". Finishing writing.

2012: **Salvador Jauregui-Ortiz**, Mario Siller, Félix Ramos and Edson Scalabrin. "Smart Environmental Architecture for Node Localization in a WSN". The 8th international Conference on Intelligent Environments, IE'12. pp. 222-227, Guanajuato, Mexico. ISBN: 978-1-4673-2093-1.

2011: **Salvador Jauregui-Ortiz**, Mario Siller, Félix Ramos and Edson Scalabrin. "Improving node localization by using Logical Position of Nodes". In proceedings of UCAmI 2011.

2011: **Salvador Jauregui-Ortiz**, Mario Siller, Félix Ramos and Edson Scalabrin. "Improving node localization in WSN by using obstruction level indicator". IEEE, SMC'11, pp. 1981-1985, ISSN: 1062.922X, Anchorage, Alaska, USA. October, 9-12, 2011.

2011: **Salvador Jauregui-Ortiz**, Mario Siller and Félix Ramos. "Node localization in WSN using trigonometric figures". IEEE, RRW, WisNet, pp. 65-68, ISBN: 978-1-42448414-0, Phoenix, Arizona, USA, January 16-18, 2011. *Fourth Best paper award (see Appendix D).*

2009: **Salvador Jauregui** and Mario Siller. "A big picture on localization algorithms considering sensor logic location". System, Man and Cybernetics, SMC. IEEE International Conference on. October 2009. On Pages: 734 - 739, ISBN: 978-1-4244-2793-2.

2008: Miguel A. Sánchez-Acevedo, Gustavo A. Torres-Blanco, **Salvador Jauregui-Ortiz**, Ernesto López-Mellado, Félix Ramos-Corchado. "Formation of robots based on local interactions" SENIE October 2008, On pages. 462-468 ISBN: 978-970-31-0944-9.

Bibliography

[1] Chen Xueli et al. "Role of wireless sensor networks in forest fire prevention." In: *Computer Engineering and Technology (ICCET), 2010 2nd International Conference on*. Vol. 4. 2010, pp. V4–12–V4–14. DOI: 10.1109/ICCET.2010.5486359.

[2] Junguo Zhang et al. "Forest fire detection system based on wireless sensor network." In: *Industrial Electronics and Applications, 2009. ICIEA 2009. 4th IEEE Conference on*. 2009, pp. 520–523. DOI: 10.1109/ICIEA.2009.5138260.

[3] P. Hoogeboom. "Classification of Agricultural Crops in Radar Images." In: *Geoscience and Remote Sensing, IEEE Transactions on* GE-21.3 (1983), pp. 329–336. ISSN: 0196-2892. DOI: 10.1109/TGRS.1983.350562.

[4] Wang Jian and Du Shiping. "Study on the Image Segmentation of Field Crops Based on the Fusion of Infrared and Visible-Light Images." In: *Photonics and Optoelectronic (SOPO), 2010 Symposium on*. 2010, pp. 1–4. DOI: 10.1109/SOPO.2010.5504350.

[5] Shining Li, Jin Cui, and Zhigang Li. "Wireless Sensor Network for Precise Agriculture Monitoring." In: *Intelligent Computation Technology and Automation (ICICTA), 2011 International Conference on*. Vol. 1. 2011, pp. 307–310. DOI: 10.1109/ICICTA.2011.87.

[6] V.L. Narasimhan, A.A. Arvind, and K. Bever. "Greenhouse Asset Management Using Wireless Sensor-Actor Networks." In: *Mobile Ubiquitous Computing, Systems, Services and Technologies, 2007. UBICOMM 2007. International Conference on*. 2007, pp. 9–14. DOI: 10.1109/UBICOMM.2007.43.

[7] R.I. da Silva et al. "Wireless sensor network for disaster management." In: *Network Operations and Management Symposium (NOMS), 2010 IEEE*. 2010, pp. 870–873. DOI: 10.1109/NOMS.2010.5488351.

[8] R. Ohbayashi et al. "Monitoring system for landslide disaster by wireless sensing node network." In: *SICE Annual Conference, 2008*. 2008, pp. 1704–1710. DOI: 10.1109/SICE.2008.4654938.

[9] S. Jauregui and M. Siller. "A big picture on localization algorithms considering sensor logic location." In: *Systems, Man and Cybernetics, 2009. SMC 2009. IEEE International Conference on*. 2009, pp. 734–739. DOI: 10.1109/ICSMC.2009.5346836.

[10] Nissanka B. Priyantha et al. "The cricket compass for context-aware mobile applications." In: *Proceedings of the 7th annual international conference on Mobile computing and networking*. MobiCom 2001. Rome, Italy: ACM, 2001, pp. 1–14. ISBN: 1-58113-422-3. DOI: 10.1145/381677.381679. URL: http://doi.acm.org/10.1145/381677.381679.

[11] Bachrach Jonathan and Taylor Christopher. "Localization in Sensor Networks." In: *Handbook of Sensor Networks*. John Wiley and Sons, Inc., 2005, pp. 277–310. ISBN: 9780471744146. DOI: 10.1002/047174414X.ch9. URL: http://dx.doi.org/10.1002/047174414X.ch9.

[12] Mustapha Boushaba, Abdelhakim Hafid, and Abderrahim Benslimane. "High accuracy localization method using AoA in sensor networks." In: *Computer Networks* 53.18 (2009), pp. 3076 –3088. ISSN: 1389-1286. DOI: http://dx.doi.org/10.1016/j.comnet.2009.07.015. URL: http://www.sciencedirect.com/science/article/pii/S1389128609002412.

[13] M. Dakkak et al. "Indoor localization method based on RTT and AOA using coordinates clustering." In: *Comput. Netw.* 55.8 (2011), pp. 1794–1803. ISSN: 1389-1286. DOI: 10.1016/j.comnet.2011.01.010. URL: http://dx.doi.org/10.1016/j.comnet.2011.01.010.

[14] M. Maroti et al. "Shooter localization in urban terrain." In: *Computer* 37.8 (2004), pp. 60–61. ISSN: 0018-9162. DOI: 10.1109/MC.2004.104.

[15] Jeffrey Hightower, Roy Want, and Gaetano Borriello. *SpotON: An Indoor 3D Location Sensing Technology Based on RF Signal Strength*. UW CSE 00-02-02. Seattle, WA: University of Washington, Department of Computer Science and Engineering, 2000.

[16] Tian He et al. "Range-free localization schemes for large scale sensor networks." In: *Proceedings of the 9th annual international conference on Mobile computing and networking*. MobiCom 2003. San Diego, CA, USA: ACM, 2003, pp. 81–95. ISBN:

1-58113-753-2. DOI: 10.1145/938985.938995. URL: http://doi.acm.org/10.1145/938985.938995.

[17] N. Bulusu, J. Heidemann, and D. Estrin. "GPS-less low-cost outdoor localization for very small devices." In: *Personal Communications, IEEE* 7.5 (2000), pp. 28–34. ISSN: 1070-9916. DOI: 10.1109/98.878533.

[18] Andreas Savvides, Chih-Chieh Han, and Mani B. Strivastava. "Dynamic fine-grained localization in Ad-Hoc networks of sensors." In: *Proceedings of the 7th annual international conference on Mobile computing and networking*. MobiCom 2001. Rome, Italy: ACM, 2001, pp. 166–179. ISBN: 1-58113-422-3. DOI: 10.1145/381677.381693. URL: http://doi.acm.org/10.1145/381677.381693.

[19] Drago Niculescu and Badri Nath. "DV Based Positioning in Ad Hoc Networks." English. In: *Telecommunication Systems* 22.1-4 (2003), pp. 267–280. ISSN: 1018-4864. DOI: 10.1023/A:1023403323460.

[20] J. Blumenthal et al. "Weighted Centroid Localization in Zigbee-based Sensor Networks." In: *Intelligent Signal Processing, 2007. WISP 2007. IEEE International Symposium on*. 2007, pp. 1–6. DOI: 10.1109/WISP.2007.4447528.

[21] S. Jauregui-Ortiz, M. Siller, and F. Ramos. "Node localization in WSN using trigonometric figures." In: *Wireless Sensors and Sensor Networks (WiSNet), 2011 IEEE Topical Conference on*. 2011, pp. 65–68. DOI: 10.1109/WISNET.2011.5725030.

[22] Rui Huang and Gergely V. Záruba. "Monte Carlo localization of wireless sensor networks with a single mobile beacon." In: *Wirel. Netw.* 15.8 (2009), pp. 978–990. ISSN: 1022-0038. DOI: 10.1007/s11276-008-0096-3. URL: http://dx.doi.org/10.1007/s11276-008-0096-3.

[23] P. Bahl and V.N. Padmanabhan. "RADAR: an in-building RF-based user location and tracking system." In: *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*. Vol. 2. 2000, 775–784 vol.2. DOI: 10.1109/INFCOM.2000.832252.

[24] S.Y. Seidel and T.S. Rappaport. "914 MHz path loss prediction models for indoor wireless communications in multifloored buildings." In: *Antennas and Propagation, IEEE Transactions on* 40.2 (1992), pp. 207–217. ISSN: 0018-926X. DOI: 10.1109/8.127405.

[25] Radu Stoleru, John A. Stankovic, and Sang H. Son. "Robust node localization for wireless sensor networks." In: *Proceedings of the 4th workshop on Embedded networked sensors*. EmNets '07. Cork, Ireland: ACM, 2007, pp. 48–52. ISBN: 978-1-59593-694-3. DOI: 10.1145/1278972.1278984. URL: http://doi.acm.org/10.1145/1278972.1278984.

[26] R. Stoleru, J.A. Stankovic, and S.H. Son. "On composability of localization protocols for wireless sensor networks." In: *Network, IEEE* 22.4 (2008), pp. 21–25. ISSN: 0890-8044. DOI: 10.1109/MNET.2008.4579767.

[27] Roy Want et al. "The active badge location system." In: *ACM Trans. Inf. Syst.* 10.1 (1992), pp. 91–102. ISSN: 1046-8188. DOI: 10.1145/128756.128759. URL: http://doi.acm.org/10.1145/128756.128759.

[28] R.A. Brooks. "A robust layered control system for a mobile robot." In: *Robotics and Automation, IEEE Journal of* 2.1 (1986), pp. 14–23. ISSN: 0882-4967. DOI: 10.1109/JRA.1986.1087032.

[29] *American Association of State Highway and Transportation Officials*. URL: athttp://www.transportation.org/Pages/default.aspx.

[30] Radhika Nagpal, Howard Shrobe, and Jonathan Bachrach. "Organizing a global coordinate system from local information on an ad hoc sensor network." In: *Proceedings of the 2nd international conference on Information processing in sensor networks*. IPSN'03. Palo Alto, CA, USA: Springer-Verlag, 2003, pp. 333–348. ISBN: 3-540-02111-6. URL: http://dl.acm.org/citation.cfm?id=1765991.1766014.

[31] G. V. Zàruba et al. "Indoor location tracking using RSSI readings from a single Wi-Fi access point." In: *Wirel. Netw.* 13.2 (Apr. 2007), pp. 221–235. ISSN: 1022-0038. DOI: 10.1007/s11276-006-5064-1. URL: http://dx.doi.org/10.1007/s11276-006-5064-1.

[32] A. Lewandowski and C. Wietfeld. "A comprehensive approach for optimizing ToA-localization in harsh industrial environments." In: *Position Location and Navigation Symposium (PLANS), 2010 IEEE/ION*. 2010, pp. 516–525. DOI: 10.1109/PLANS.2010.5507255.

[33] L. Girod and D. Estrin. "Robust range estimation using acoustic and multimodal sensing." In: *Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on*. Vol. 3. 2001, 1312–1320 vol.3. DOI: 10.1109/IROS.2001.977164.

[34] P.Q.C. Ly, S.D. Elton, and D.A. Gray. "AOA estimation of two narrowband signals using interferometry." In: *Phased Array Systems and Technology (ARRAY), 2010 IEEE International Symposium on*. 2010, pp. 1004–1009. DOI: 10.1109/ARRAY.2010.5613238.

[35] M.S. Brandstein, J.E. Adcock, and H.F. Silverman. "A closed-form location estimator for use with room environment microphone arrays." In: *Speech and Audio Processing, IEEE Transactions on* 5.1 (1997), pp. 45–50. ISSN: 1063-6676. DOI: 10.1109/89.554268.

[36] A. Faheem, R. Virrankoski, and M. Elmusrati. "Improving RSSI based distance estimation for 802.15.4 wireless sensor networks." In: *Wireless Information Technology and Systems (ICWITS), 2010 IEEE International Conference on*. 2010, pp. 1–4. DOI: 10.1109/ICWITS.2010.5611813.

[37] P. Corral, V. Almenar, and A.C de C.Lima. "Distance Estimation System Based on ZigBee." In: *Spread Spectrum Techniques and Applications, 2008. ISSSTA '08. IEEE 10th International Symposium on*. 2008, pp. 817–820. DOI: 10.1109/ISSSTA.2008.159.

[38] Liu. Yu, Yi. Xiao, and He You. "A Novel Centroid Localization for Wireless Sensor Networks." In: *International Journal of Distributed Sensor Networks* vol. 2012, Article ID 829253 (2012). DOI: doi:10.1155/2012/829253.

[39] Rodrigo Vera, Sergio F. Ochoa, and Roberto G. Aldunate. "EDIPS: an Easy to Deploy Indoor Positioning System to support loosely coupled mobile work." In: *Personal Ubiquitous Comput.* 15.4 (2011), pp. 365–376. ISSN: 1617-4909. DOI: 10.1007/s00779-010-0357-x. URL: http://dx.doi.org/10.1007/s00779-010-0357-x.

[40] A. Ziotopoulos, M.F. Jacome, and G. de Veciana. "An RFID-Based Platform Supporting Context-Aware Computing in Complex Spaces." In: *Mobile Data Management, 2007 International Conference on*. 2007, pp. 294–298. DOI: 10.1109/MDM.2007.62.

[41] Jose Bravo et al. "Towards Natural Interaction by Enabling Technologies: A Near Field Communication Approach." In: *Constructing Ambient Intelligence*. Ed. by Max Mühlhäuser, Alois Ferscha, and Erwin Aitenbichler. Vol. 11. Communications in Computer and Information Science. Springer Berlin Heidelberg, 2008, pp. 338–351. ISBN: 978-3-540-85378-7. DOI: 10.1007/978-3-540-85379-4_39. URL: http://dx.doi.org/10.1007/978-3-540-85379-4_39.

[42] J. Bravo et al. “Modeling contexts by RFID-sensor fusion.” In: *Pervasive Computing and Communications Workshops, 2006. PerCom Workshops 2006. Fourth Annual IEEE International Conference on*. 2006, 5 pp.–34. DOI: 10.1109/PERCOMW.2006.95.

[43] Bravo. J et al. “RFID-Sensor Fusion: An experience at clinical sessions.” In: In workshop PTA 2006, 2006, pp. 347–35347–354.

[44] J. Bravo et al. “Supporting Informal Meetings in Hospitals.” In: *Pervasive Health Conference and Workshops, 2006*. 2006, pp. 1–4. DOI: 10.1109/PCTHEALTH.2006.361678.

[45] Y. Bar-Shalom. *Tracking and Data Association*. San Diego, CA, USA: Academic Press Professional, Inc., 1987. ISBN: 0-120-79760-7.

[46] D.L. Hall and J. Llinas. “An introduction to multisensor data fusion.” In: *Proceedings of the IEEE* 85.1 (1997), pp. 6–23. ISSN: 0018-9219. DOI: 10.1109/5.554205.

[47] Jared B. Bancroft and Gérard Lachapelle. “Data Fusion Algorithms for Multiple Inertial Measurement Units.” In: *Sensors* 11.7 (2011), pp. 6771–6798. ISSN: 1424-8220. DOI: 10.3390/s110706771. URL: http://www.mdpi.com/1424-8220/11/7/6771.

[48] T. Kleine-Ostmann and A.E. Bell. “A data fusion architecture for enhanced position estimation in wireless networks.” In: *Communications Letters, IEEE* 5.8 (2001), pp. 343–345. ISSN: 1089-7798. DOI: 10.1109/4234.940986.

[49] Yu-Chee Tseng et al. “Location Tracking in a Wireless Sensor Network by Mobile Agents and Its Data Fusion Strategies.” In: *Proceedings of the 2Nd International Conference on Information Processing in Sensor Networks*. IPSN’03. Palo Alto, CA, USA: Springer-Verlag, 2003, pp. 625–641. ISBN: 3-540-02111-6. URL: http://dl.acm.org/citation.cfm?id=1765991.1766034.

[50] Donghoon Lee et al. “Sensor fusion localization system for outdoor mobile robot.” In: *ICCAS-SICE, 2009*. 2009, pp. 1384–1387.

[51] Byoung-Suk Choi and Ju-Jang Lee. “Sensor network based localization algorithm using fusion sensor-agent for indoor service robot.” In: *Consumer Electronics, IEEE Transactions on* 56.3 (2010), pp. 1457–1465. ISSN: 0098-3063. DOI: 10.1109/TCE.2010.5606283.

[52] Rui Huang and Gergely V. Záruba. "Monte Carlo Localization of Wireless Sensor Networks with a Single Mobile Beacon." In: *Wirel. Netw.* 15.8 (2009), pp. 978–990. ISSN: 1022-0038. DOI: 10.1007/s11276-008-0096-3. URL: http://dx.doi.org/10.1007/s11276-008-0096-3.

[53] Yu-Jhong Fu et al. "A Single Mobile Anchor Localization Scheme for Wireless Sensor Networks." In: *High Performance Computing and Communications (HPCC), 2011 IEEE 13th International Conference on*. 2011, pp. 946–950. DOI: 10.1109/HPCC.2011.137.

[54] Lingxuan Hu and David Evans. "Localization for Mobile Sensor Networks." In: *Proceedings of the 10th Annual International Conference on Mobile Computing and Networking*. MobiCom '04. Philadelphia, PA, USA: ACM, 2004, pp. 45–57. ISBN: 1-58113-868-7. DOI: 10.1145/1023720.1023726. URL: http://doi.acm.org/10.1145/1023720.1023726.

[55] Aline Baggio and Koen Langendoen. "Monte Carlo Localization for Mobile Wireless Sensor Networks." In: *Ad Hoc Netw.* 6.5 (July 2008), pp. 718–733. ISSN: 1570-8705. DOI: 10.1016/j.adhoc.2007.06.004. URL: http://dx.doi.org/10.1016/j.adhoc.2007.06.004.

[56] Hamid Mirebrahim and Mehdi Dehghan. "Monte Carlo Localization of Mobile Sensor Networks Using the Position Information of Neighbor Nodes." In: *Proceedings of the 8th International Conference on Ad-Hoc, Mobile and Wireless Networks*. ADHOC-NOW '09. Murcia, Spain: Springer-Verlag, 2009, pp. 270–283. ISBN: 978-3-642-04382-6. DOI: 10.1007/978-3-642-04383-3_20. URL: http://dx.doi.org/10.1007/978-3-642-04383-3_20.

[57] P. Levis et al. "TinyOS: An Operating System for Sensor Networks." In: *Ambient Intelligence*. Ed. by Werner Weber, JanM. Rabaey, and Emile Aarts. Springer Berlin Heidelberg, 2005, pp. 115–148. ISBN: 978-3-540-23867-6. DOI: 10.1007/3-540-27139-2_7. URL: http://dx.doi.org/10.1007/3-540-27139-2_7.

[58] Rodolfo de Paz Alberola and Dirk Pesch. "AvroraZ: extending Avrora with an IEEE 802.15.4 compliant radio chip model." In: *Proceedings of the 3nd ACM workshop on Performance monitoring and measurement of heterogeneous wireless and wired networks*. PM2HW2N '08. Vancouver, British Columbia, Canada: ACM, 2008, pp. 43–50. ISBN: 978-1-60558-239-9. DOI: 10.1145/1454630.1454637. URL: http://doi.acm.org/10.1145/1454630.1454637.

[59] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification*. 2nd ed. New York: Wiley, 2001.

[60] The University of Waikato. *Weka*. 2003. URL: http://www.cs.waikato.ac.nz/ml/weka/.

[61] J. R. Quinlan. "Improved use of continuous attributes in C4.5." In: *J. Artif. Int. Res.* 4.1 (Mar. 1996), pp. 77–90. ISSN: 1076-9757. URL: http://dl.acm.org/citation.cfm?id=1622737.1622742.

[62] Amy Williams et al. "An efficient and robust ray-box intersection algorithm." In: *ACM SIGGRAPH 2005 Courses*. SIGGRAPH '05. Los Angeles, California: ACM, 2005. DOI: 10.1145/1198555.1198748. URL: http://doi.acm.org/10.1145/1198555.1198748.

[63] S. Jauregui-Ortiz et al. "Improving node localization in WSN by using obstruction level indicator." In: *Systems, Man, and Cybernetics (SMC), 2011 IEEE International Conference on*. 2011, pp. 1981–1985. DOI: 10.1109/ICSMC.2011.6083962.

[64] S. Jauregui-Ortiz, M. Siller, and F. Ramos. "Improving node localization by using Logical Position of Nodes." In: *In Proc. UCAmI*. 2011.

[65] Anind K. Dey. "Understanding and Using Context." In: *Personal Ubiquitous Comput.* 5.1 (2001), pp. 4–7. ISSN: 1617-4909. DOI: 10.1007/s007790170019. URL: http://dx.doi.org/10.1007/s007790170019.

[66] Robert Sedgewick and Kevin Wayne. *Algorithms, 4th ed.* 2011. ISBN-10: 032157351X, ISBN-13: 9780321573513. Addison-Wesley Professional, 992 pp.