UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

ACCURATE AND EFFICIENT LOCALISATION IN WIRELESS SENSOR NETWORKS USING A BEST-REFERENCE SELECTION

by

Adnan Mohammed Abu-Mahfouz

Submitted in partial fulfilment of the requirements for the degree

Philosophae Doctor (Computer Engineering)

in the

Faculty of Engineering, the Built Environment and Information Technology

Department of Electrical, Electronic and Computer Engineering

UNIVERSITY OF PRETORIA

April 2011

© University of Pretoria

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

SUMMARY

ACCURATE AND EFFICIENT LOCALISATION IN WIRELESS SENSOR NETWORKS USING A BEST-REFERENCE SELECTION

by

Adnan M. Abu-Mahfouz

Supervisor:	Dr G. P. Hancke
Co-Supervisor:	Prof. G. P. Hancke
Department:	Electrical, Electronic and Computer Engineering
University:	University of Pretoria
Degree:	Philosophae Doctor (Computer Engineering)

Keywords: ALWadHA, design objectives, distance bounding, information-fusion, localisation systems, localised algorithm, location estimation, position determination, smart references selection, wireless sensor networks.

Many wireless sensor network (WSN) applications depend on knowing the position of nodes within the network if they are to function efficiently. Location information is used, for example, in item tracking, routing protocols and controlling node density. Configuring each node with its position manually is cumbersome, and not feasible in networks with mobile nodes or dynamic topologies. WSNs, therefore, rely on localisation algorithms for the sensor nodes to determine their own physical location.

The basis of several localisation algorithms is the theory that the higher the number of reference nodes (called “references”) used, the greater the accuracy of the estimated position. However, this approach makes computation more complex and increases the likelihood that the location estimation may be inaccurate. Such inaccuracy in estimation could be due to including data from nodes with a large measurement error, or from nodes that intentionally aim to undermine the localisation process. This approach also has limited success in networks with sparse references, or where data cannot always be collected from many references (due for example to communication obstructions or bandwidth limitations). These situations require a method for achieving reliable and accurate localisation using a limited number of references.

Designing a localisation algorithm that could estimate node position with high accuracy using a low number of references is not a trivial problem. As the number of references decreases, more statistical weight is attached to each reference's location estimate. The overall localisation accuracy therefore greatly depends on the robustness of the selection method that is used to eliminate inaccurate references. Various localisation algorithms and their performance in WSNs were studied. Information-fusion theory was also investigated and a new technique, rooted in information-fusion theory, was proposed for defining the best criteria for the selection of references. The researcher chose selection criteria to identify only those references that would increase the overall localisation accuracy. Using these criteria also minimises the number of iterations needed to refine the accuracy of the estimated position. This reduces bandwidth requirements and the time required for a position estimation after any topology change (or even after initial network deployment). The resultant algorithm achieved two main goals simultaneously: accurate location discovery and information fusion. Moreover, the algorithm fulfils several secondary design objectives: self-organising nature, simplicity, robustness, localised processing and security.

The proposed method was implemented and evaluated using a commercial network simulator. This evaluation of the proposed algorithm's performance demonstrated that it is superior to other localisation algorithms evaluated; using fewer references, the algorithm performed better in terms of accuracy, robustness, security and energy efficiency.

These results confirm that the proposed selection method and associated localisation algorithm allow for reliable and accurate location information to be gathered using a minimum number of references. This decreases the computational burden of gathering and analysing location data from the high number of references previously believed to be necessary.

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

OPSOMMING

AKKURATE EN DOELTREFFENDE LOKALISERING IN DRAADLOSE SENSORNETWERKE DEUR DIE KEUSE VAN DIE BESTE VERWYSINGS

deur

Adnan M. Abu-Mahfouz

Studieleier:	Dr G. P. Hancke
Mede-studieleier:	Prof. G. P. Hancke
Departement:	Elektriese, Elektroniese en Rekenaar-ingenieurswese
Universiteit:	Universiteit van Pretoria
Graad:	Philosophae Doctor (Rekenaar-Ingenieurswese)
Sleutelwoorde:	ALWadHA, ontwerpdoelwitte, afstandsbegrensing, informasiefusie, lokaliseringsisteme, gelokaliseerde algoritme, liggingskatting, posisiebepaling, slim verwysingkeusemaatstaf, draadlose sensornetwerke.

Baie toepassings van draadlose sensornetwerke (DSN) maak gebruik van kennis van die ligging van nodusse in die netwerk om doeltreffend te funksioneer. Lokaliseringinligting kan gebruik word as 'n basis om onder andere nasporing van items, hulp met roeteringsprotokolle en beheer oor nodusdigtheid moontlik te maak. Konfigurasie van die ligging van nodusse per hand is lomp en nie lewensvatbaar in netwerke met mobiele nodusse of dinamiese topologieë nie. DSN berus dus op lokaliseringalgoritmes wat op hoogte kan bly van al hulle nodusse se fisiese ligging.

Verskeie lokaliseringalgoritmes ondersteun die idee dat om die akkuraatheid van posisieskatting te ondersteun, 'n groot aantal verwysings gebruik moet word. Hierdie benadering het egter verskeie nadele. Die insluiting van 'n groot aantal verwysings verhoog die kompleksiteit van lokalisering sowel as die moontlikheid dat die lokaliseringskatting onakkuraat mag wees, hetsy as gevolg van die insluiting van data afkomsig van nodes met 'n groot metingsfout, of van nodusse wat bewustelik probeer om die lokaliseringproses te ondermyn. Hierdie benadering het ook beperkte sukses in netwerktopologieë met min verwysings, of waar datakommunikasie beperk is en data nie deurlopend van 'n groot aantal nodusse versamel kan word nie. In sodanige gevalle word 'n metode om betroubare en akkurate lokalisering met 'n beperkte aantal verwysings te bereik, vereis.

Dit is nie maklik om 'n lokaliseringalgoritme wat hoogs akkuraat is, maar 'n beperkte aantal verwysings gebruik, te ontwerp nie. Soos die aantal verwysings toeneem, word meer statistiese gewig toegeken aan die lokalisering van elke verwysingskatting en die oorhoofse lokaliseringakkuraatheid is dus in 'n groot mate afhanklik van die robuustheid van die onderliggende seleksiekriteria om sodoende onakkurate verwysing uit te skakel. Verskeie lokaliseringalgoritmes en hulle werkverrigting in DSN is bestudeer. Informasiefusieteorie is ook ondersoek. Vervolgens is 'n nuwe tegniek, gegrond op informasiefusieteorie, voorgestel om die beste keuse van verwysings te doen. Kriteria om 'n verwysing in te sluit, is versigtig saamgestel om slegs die verwysings te identifiseer wat die oorhoofse lokaliseringsakkuraatheid sou verbeter. Hierdie seleksiekriteria minimeer ook die aantal verfyningsiterasies en verminder sodoende die vereistes vir datakommunikasiekapasiteit, en minimeer die tyd wat dit neem voordat akkurate lokaliseringsinligting beskikbaar gestel word na 'n netwerkontplooiing of 'n verandering in topologie. Die gevolglike lokaliseringsalgoritme het gelyktydig twee hoofdoelwitte bereik: akkurate posisie-bepaling en informasiefusie. Daarbenewens voldoen die algoritme aan verskeie sekondêre ontwerpdoelwitte: self-organiserende aard, eenvoud, robuustheid, gelokaliseerde prosessering en sekuriteit.

Die voorgestelde metode is geïmplementeer en geëvalueer deur die gebruik van 'n kommersiële simulator. Hierdie evaluasie van die werkverrigting van die voorgestelde algoritme het die doeltreffendheid daarvan teenoor dié van bestaande lokaliseringsalgoritmes bewys. Deur die gebruik van minder verwysings het die algoritme beduidende verbetering getoon wat betref akkuraatheid, robuustheid, sekuriteit en energiedoeltreffendheid. Hierdie resultate bevestig dat die voorgestelde seleksiekriteria en lokalisasie-algoritme dit moontlik maak om betroubare en akkurate netwerklokaliserings-inligting in te samel en daardeur ook die berekeningslas te verlig van insameling en analise van lokaliseringdata deur die gebruik van 'n groot aantal verwysings, wat voorheen beskou is as noodsaaklik.

ACKNOWLEDGEMENTS

Finally, I have the opportunity to express my deepest gratitude to all of those who selflessly assisted, supported and made a critical contribution to the completion of this work. Without the help, supervision, advice and encouragement of many people this thesis would have never been completed.

First and foremost, thanks to God (ALLAH), the omniscient and the bestower, for granting me abundant grace without measure, patience and ability to complete such a large endeavour.

I would like to thank my supervisors, Dr Gerhard P. Hancke and Prof. Gerhard P. Hancke, for their critical contribution, guidance, support and availability when I needed their advice. They have taken great trouble to ensure the standard of this work. I believe that they are one of the most encouraging supervisors available and that any student would like to work under their supervision. A word of thanks must also go to Dr Ntsibane Ntlatlapa, John Isaac and Johan Eksteen for their encouragement, suggestions and friendship.

It has been a pleasure to work with the Wireless Africa Research Team. The priceless criticism, suggestions and feedback from the members of this team were very helpful and fruitful. I thank all staff members of the Meraka Institute at the Council for Scientific and Industrial Research (CSIR) for help and support. I would like to acknowledge the Meraka Institute at the CSIR for making this research possible and providing the required funding to complete the work. I would like to thank Mari Ferreira, Maryna Bekker, Heleen Gous and Stefanie Steenberg at the University of Pretoria for their administrative support and friendship. A word of thanks also goes to all of my friends who keep encouraging me to work hard to achieve my goal and complete this work.

Last, but certainly not least, I am grateful to my family for their love and prayers. They really made a great contribution to this work. Special thanks and gratitude to my wife, Leena, who married me in the same month that I started this work and later became the mother of my beautiful little daughter, Leyan. I thank my wife and my daughter for their love, encouragement, patience and for the several weekends they spent alone without me because of my working at my office.

DEDICATION

…   …

…

To my family

… … … … …

… … …

…

LIST OF ABBREVIATIONS

ALWadHA	An efficient Localisation algorithm for Wireless ad hoc sensor networks with High Accuracy
AoA	Angle of Arrival
APS	Ad-hoc Positioning System
Asm	Asymmetric cryptography
AT	Avoine and Tchamkerten
BB	Bussard and Bagga
BC	Brands and Chaum
CBH	Capkun, Buttyan and Hubaux
CCD	Current Challenge Dependent
CRLB	Cramer-Rao-Lower-Bound
DB	Distance Bounding
DBPoK	Distance Bounding Proof of Knowledge
DF	Distance Fraud
dwMDS	Distributed weighted-Multidimensional Scaling
ECC	Error-Correcting Codes
ER	Error Resilience
FAR	False Acceptance Rate
GFM	Greedy Filtering by Matrix
HK	Hancke and Kuhn
k-PCD	k-Previous Challenge Dependent
KA	Kim and Avoine
KAKSP	Kim, Avoine, Koeune, Standaert and Pereira
LMS	Least Median Squares
LS	Least Squares
MAC	Media Access Control
MAC	Message Authentication Code
MAD	Mutual Authentication with Distance Bounding
MAP	Mutual Authentication Protocol
Mem	Memory
MF	Mafia Fraud
MLE	Maximum Likelihood Estimation
MMSE	Minimum Mean Square Estimate
MP	Munilla and Peinado
MSC	Meadows, Syverson and Chang
MUSE	MULtiState Enhancement
NAM	Network AniMator
NDBL	Node Distribution-based Localisation

ns	Network Simulator
Otcl	Object-oriented Tool command Language
Perl	Practical Extraction and Report Language
PRF	Pseudo-Random Function
PSH	Protocol Specific Header
RF	Radio Frequency
RFID	Radio Frequency Identification
RLS	Robust Least Square
RN	Random Number
RNTS	Reid, Nieto, Tang and Senadji
RSS	Received Signal Strength
RTT	Round Trip Time
SD	Standard Deviation
SP	Singelée and Preneel
SPA	Success Probability of an Adversary
TclCL	Tool command language with Classes
TCP	Transport Control Protocol
TdoA	Time Difference of Arrival
TF	Terrorist Fraud
TI	Trustability Indicator
TMA	Trujillo, Martin and Avoine
ToA	Time of Arrival
ToF	Time of Flight
TP	Tu and Piramuthu
US	Ultrasound
UWB	Ultra-Wideband
WSN	Wireless Sensor Network

TABLE OF CONTENTS

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

Chapter 1

INTRODUCTION

The proliferation of wireless communication technologies has enabled the development of wireless sensor networks (WSNs), which consist of a large number of small and cheap sensors with limited resources, such as computing, communication, storage and energy [1]. These sensor nodes are able to sense, measure and collect raw data from the environment, perform simple computations and then transmit only the required and partially processed data to the node responsible for fusion [2]. Sensor nodes can be deployed either manually at fixed locations or randomly into the field. After deployment these sensor nodes start measuring various properties of the environment, such as light, humidity, temperature, barometric pressure, velocity, acceleration, acoustics and magnetic field, using the different types of sensors that may be attached to these nodes. The measured data will be transferred by a multi-hop infrastructureless architecture to a base station, where data will be manipulated and a decision can be taken.

WSNs have been deployed extensively in areas such as military operations [3], health monitoring [4], natural disaster management [5] and hazardous environments [6]. Most of these applications require that the position of the nodes must be determined. In some scenarios node location information plays a critical role, such as data-centric storage application [7]. Several WSN techniques require highly accurate knowledge of the location, such as the geographical routing technique [8, 9], network security [10] and energy efficient management [11]. The main advantages of node location information are enhancing the efficiency of the WSNs, identifying the location of an event of interest, facilitating numerous application services and assisting in various system functionalities [12].

A few examples can be mentioned to show the importance of determining the nodes' location. Sensor nodes that are equipped with a thermal sensor could be used for fire detection. As soon as they detect a fire they send an alarm to the base station, which instructs the rescue team to take action. However, if these sensor nodes did not also send their location, then the base station would be unable to indicate the place of the fire to the rescue team. The location of sensors will be even more important if they are used in a battlefield to detect the location of enemy tanks or troops.

One way to localise sensor nodes is through manual configuration, when the fixed location of sensor nodes is predetermined. This solution is too cumbersome, and it would not be feasible in large WSNs. In addition, it could be very difficult to apply in inaccessible terrain, on the battlefield or in disaster relief operations. Moreover, existing location systems are not always suitable for WSN deployment; for example, a global positioning system (GPS) cannot be deployed inside buildings; localisation methods based on mobile cell/base station triangulation would only be practical in areas within deployed infrastructure; and most WiFi location techniques work only indoors.

Therefore, establishing the location of nodes in WSNs is a very challenging task. Recently, several "location-discovery" algorithms for WSNs have been proposed for this purpose. One approach that has been followed by these algorithms is to use special nodes called "beacons", which know their location (e.g. through a GPS receiver or manual configuration). The other nodes that do not know their location, sometimes referred to as "unknowns", use different techniques to compute their own position based on the location information of the beacons and the measured distance to these beacons. The term "reference nodes" or simply "references" will be used in this study to refer to the sensor nodes which are willing to help other nodes to estimate their position. Therefore, the reference set includes beacons and knowns (i.e. unknowns which have obtained their position) which are willing to act as a reference for other unknowns.

1.1 PROBLEM STATEMENT

Several localisation algorithms rely on using all or most of the available references to enhance their performance. They are based on the assumption that using more references could enhance the accuracy of estimation. However, to implement an efficient localisation algorithm for WSNs, this assumption should be reconsidered for the following reasons:

- The complexity of computation of a localisation algorithm increases in proportion to the number of references used [13], so more references require more computation, more memory space and more energy consumption. A resource-constrained network such as WSN, however, needs to reduce the number of actively participating references as far as possible.

- The validity of the assumption could be compromised in a "hostile" environment. One or more malicious nodes could deliberately provide incorrect location information to mislead other nodes. Preventing such types of malicious node from taking part in the localisation process will lead to more accurate estimation of position than when all the available references are used. Furthermore, from a security and privacy perspective, only a subset of nodes should take part in a task.

- The availability of a high number of references is a critical issue that cannot be guaranteed in WSNs for two main reasons: The first is the dynamic changes in WSNs due to nodes dying or nodes moving. The second is that in WSNs it is not realistic to expect that all nodes will always be able to participate in every task (owing for example to lack of energy or the existence of obstacles).

- Estimate of location are based on one type of information fusion that combines complementary data to draw inferences. In other words, a node can fuse the location of and the measured distance to the neighbouring references to obtain its position. However, when the amount of additional incorrect data outweighs the amount of additional correct data, this can reduce the overall performance of the fusion process [14].

- Distance-measurement techniques are all subject to errors. In a noisy environment, the position estimation will be more accurate if the node excludes those references that could bias the estimate toward an inaccurate location.

For these reasons, in order to enable unknown nodes to estimate their own location, it is desirable to select those references (i.e. subset of references) that could contribute more to accuracy, rather than using all the available references. However, selecting the proper subset of references is a very challenging problem. The goal is to use a low number of references to achieve high accuracy. Using a simple but inefficient technique, such as selecting the nearest three references, would make it possible to achieve the first part of this goal, but it will be impossible to achieve high accuracy. On the other hand, using a complicated technique to eliminate only undesired references could enable the achievement of good accuracy, but that could compromise other issues such as simplicity and energy efficiency.

Several localisation algorithms are based on using a subset of references instead of using all of them. Most of these algorithms focus on improving the criteria for selecting references. However, specifying the proper number of references that should be used to guarantee a certain level of accuracy has rarely been discussed in the literature.

1.2 THESIS STATEMENT

In the light the above, the hypothesis of this study is that a *localisation algorithm can rely on using a low number of references to achieve an accurate estimation without compromising the simplicity, security, robustness or the energy efficiency of the algorithm.*

Using all of the available references could enhance the accuracy of position estimation. However, following this approach in WSNs with limited resources could result in several constraints and problems, as mentioned earlier. An efficient localisation algorithm for WSNs will be designed, which relies on proper selection criteria for references in order to enable sensor nodes to estimate their position with good accuracy but using a low number of references.

Designing a proper method to select the best subset of references to contribute to high accuracy is a challenge. However, using this subset of references would not only overcome the problems associated with using all of the available references, but would also help to achieve several design objectives. A subset of references makes the localisation algorithm tolerant of failures of nodes and so enhances its *robustness*. Reducing the number of references used will dramatically reduce the computation and communication overheads, which will improve the *energy efficiency* of the algorithm. *Security* can be achieved by excluding malicious nodes from the selected subset of references. The selection criteria may be defined in a manner that will fulfil the three required conditions of the "*localised position discovery algorithm*", which will be mentioned in the next section.

1.3 RESEARCH OBJECTIVES

The main objective of this study is to develop an efficient localisation algorithm that enables sensor nodes to estimate their location with high accuracy. Existing localisation algorithms have targeted several design objectives, including the following:

- **Accuracy:** In localisation systems, the accuracy of location estimation may be regarded as the most important design objective. An efficient localisation algorithm should not introduce a high estimation error (owing for example to using complex computations or improper techniques). On the other hand, this algorithm should be able to deal with the error caused by iterative estimation.

- **Self-organising properties:** A localisation algorithm should be independent of global infrastructure and beacon placement, which implies that there is no fine control over the placement of the sensor nodes when the network is installed [15], especially if random deployment is the only possible way to distribute sensor nodes (e.g. in inaccessible terrain or on the battlefield). Thus the localisation algorithm should not require the beacons to be placed in certain locations or in a specific pattern (e.g. a triangle).

- **Simplicity:** Resource-constrained networks such as WSNs need a simple localisation algorithm in terms of computations, resources required, number of references used and the number of required iterations before getting an accurate position estimate. A simple localisation algorithm is not only a resource-efficient algorithm, but it also reduces the error that could be introduced because of complex computations.

- **Robustness:** Sensor nodes are prone to failure due to lack of power or physical damage. Location discovery is based on physical measurements, which may be markedly inaccurate owing to several types of error that could result from measurement; finite precision, objective function-specific, intractable optimisation tasks; or distributed algorithms [16]. Therefore, a localisation algorithm should be tolerant of node failures and the various localisation errors.

- **Energy efficiency:** Sensor nodes can only be equipped with a limited power source, which might be impossible to replenish, and so the sensor node lifetime is mainly dependent on battery lifetime [2]. Therefore, an energy-aware localisation algorithm should employ several techniques to reduce the computation and communication overheads, thus reducing energy consumption.

- **Conditions for localised algorithms:** Localised algorithms are a special type of distributed algorithm in which only a subset of nodes in the WSN participates in sensing, communication and computation [17]. Therefore, the algorithms used for "location discovery", or discovering the location of nodes, should meet the following three conditions: Firstly, requesting and processing of information takes place only locally, without any central coordination overheads. Secondly, only a subset of nodes takes part in the process of estimating the position. Finally, the selected subset is the one most likely to contribute to a highly accurate position estimate.

- **Information fusion**: Information fusion can play two roles in localisation algorithms: a supporting role and a leading role. In the *supporting role*, information fusion acts as a tool to assist the localisation algorithms, by using one of the information-fusion techniques to aid the location-discovery process. In the *leading role*, the localisation algorithms are designed to support an information-fusion application. The information-fusion techniques used guide the location-discovery process and the fusion process simultaneously. This means that the localisation algorithm should be designed with two objectives: location discovery and achieving information fusion.

- **Security:** The key role they play and the fragility of the localisation systems make them possible targets of an attack that could compromise the entire functioning of a WSN and lead to incorrect plans and decision making [18]. WSNs require a secure localisation system that is able to work in a hostile environment and to prevent compromised nodes from participating in the localisation process.

A quantitative comparison referred to previously [15] showed that there is no localisation algorithm that performs best, considering different design objectives. The authors [19] further confirm that there is no single localisation algorithm that fulfils all of these objectives because of the fundamental limitation of ad-hoc localisation systems that use only range measurements. Thus, designing a localisation algorithm that fulfils all of these design objectives can be considered a challenge that provides more motivation to this investigation.

1.4 CHAPTER OVERVIEWS

Chapter 2 analyses the different categories of localisation algorithms, reviews the general concepts of localisation systems, and compares several approaches that can be used to select a subset of references.

Chapter 3 outlines a new localisation algorithm based on a subset of references called ALWadHA (**a**n efficient **l**ocalisation algorithm for **w**ireless **ad** hoc sensor networks with **h**igh **a**ccuracy) and highlights the advantages of several techniques used by this algorithm. It explains how the current version of the network simulator ns-2 (ns-2.34) was extended by adding new modules to simulate localisation systems in WSNs. It explains the class hierarchy of new classes, reviews the structure of extended ns-2, indicates the guidelines for using the new localisation system and gives an overview of the tools used to manipulate the resulting trace files. The extended ns-2 will be used to evaluate the ALWadHA and to compare its performance with other localisation algorithms. This chapter focuses mainly on two metrics: estimation error and number of references used. Several experiments will be performed, considering different aspects of evaluating these two metrics. With regard to the design objectives, this chapter investigates accuracy, self-organising abilities, simplicity and robustness.

Chapter 4 explains the concept of information fusion and reviews several information-fusion techniques used by localisation systems. It explains the three conditions required by algorithms to be considered as localised algorithms. It shows how the three filters used by ALWadHA assist in achieving these three conditions and selecting the best subset of references. The information fusion used by localisation systems has been classified in three levels. Use of these three levels by ALWadHA makes information fusion play a leading role and achieves the most important objectives of information fusion for WSNs, namely improving accuracy and saving energy. Several experiments will be performed to evaluate ALWadHA algorithms in terms of the mean error at each iteration, the number of "location request" and "location response" packets and the remaining energy. This chapter deals with the following design objectives: information fusion, localised algorithms and energy efficiency.

Chapter 5 discusses the security aspect of localisation systems. It starts by reviewing the security attacks that could compromise each component of a localisation system, then it discusses the main techniques used by the secure localisation algorithms to prevent these attacks. It discusses the techniques that can be used to implement a secure distance estimation and suggests a distance-bounding approach as a promising solution for ALWadHA. It defines a comparison framework that will be used to compare selected distance-bounding protocols, discusses the selected protocols and aspects affecting their practical implementation, after which it provides a comparative performance summary. Finally, the chapter will investigate the attack resistance of the ALWadHA algorithm

Chapter 6 concludes the research work, summarises its main contributions and finally suggests possible areas and challenges for future work.

Chapter 2

BACKGROUND

This chapter reviews the general concepts of localisation systems. In addition to giving an overview of the topic, the purpose of this chapter is to give the scope of the literature relating to different categories of localisation algorithms, and the various approaches that can be used to select a subset of references. However, a further literature review will be provided in chapters 4 and 5.

2.1 CATEGORISATION OF LOCALISATION ALGORITHMS

Localisation algorithms differ from one another in various features [20], such as the way of collecting input data, the state of sensor nodes (which could be static or mobile), the place of deployment (indoors or outdoors), applicability in a 2-D or a 3-D plane, the requirement of additional hardware, the way of requesting location information (either on demand or periodic) and the node responsible for location estimation (which could be the sensor node itself or another sensor node).

Localisation algorithms can be classified using different types of categories. Franceschini *et al.* [21] classify them according to the following four categories: pre-configured coordinates; nodes' location propagation; granularity of information; and computational distribution. One could also classify them further, on the basis of the number of estimations and the set of references used. According to these classifications, the proposed localisation algorithm can be classified as a beacon-based, incremental, fine-grained, distributed, successive-refinement and subset-references algorithm. The rest of this section will describe the reasons for adopting these approaches when developing the proposed algorithm.

2.1.1 Pre-configured coordinates

Localisation algorithms can be classified as either “beacon-based” or“beacon-free”, based on whether there are any nodes with pre-configured coordinates or not. In the *beacon-based* type, location discovery requires special sensor nodes, called beacons, that know their location through a GPS receiver or manual configuration [22]. The second, the *beacon-free* approach, does not assume the availability of beacon nodes; it rather estimates the relative locations of nodes from a set of geometric constraints extracted from proximity measurement [23]. The beacon-free approaches could involve high cost of collaboration among the sensor nodes and increase the communication overheads, which is undesirable for energy-starved WSNs.

2.1.2 Location propagation of nodes

Localisation algorithms can be classified as having either “incremental” or “concurrent” approaches, based on how information about each node's location propagates in the network. *Incremental* algorithms [24] start with a low number of beacons. As soon as the unknowns estimate their position, they may serve as new reference points. This process can be applied incrementally till all (or most of) the sensor nodes estimate their position. In a *concurrent* approach (also called multi-hop localisation) [25, 26], on the other hand, many pairs of sensor nodes communicate and share measurements to estimate the location of all sensor nodes. All sensor nodes' positions are estimated simultaneously rather than each sensor node position being solved one at a time. This approach allows unknowns to make measurements with other unknowns in order to gain additional information that could enhance the accuracy and robustness of the localisation system. However, the measurement of multi-hop could suffer inevitable error, due to the compounding of error from the approximated measurement at each hop.

2.1.3 Granularity of information

Localisation algorithms can be classified as having either “fine-grained” or “coarse-grained” approaches, based on the granularity of information acquired by sensor nodes. *Fine-grained* approaches [27] use accurate information in location estimation, for example measuring the distance to beacons using received signal strength (RSS) or time of arrival (ToA) techniques. *Coarse-grained* approaches [28] use less accurate information by using rough techniques, such as hop-count, to measure the distance to beacons. This approach reduces the number of required beacons, but it could lead to less accurate position estimation than fine-grained approaches.

2.1.4 Computational distribution

Localisation algorithms can be classified as having either "centralised" or "decentralised" (distributed) approaches based on whether the computation of the position is performed at each node or at a central unit. A *centralised* system [29] requires global knowledge in the sense that all measured data are available, while in a distributed system [22] data are provided by a set of neighbouring nodes. Theoretically, a centralised system may outperform a distributed one, because the central unit has global knowledge. However, this system also requires that all the raw data (or processed estimates) be transmitted from the nodes to the central unit. Such a high volume of communication might not be practical and might consume too many system resources.

In a *decentralised* or distributed system, each node has its own processing facility to perform position estimation based on local observation and the information received from neighbouring nodes. The main advantages of a distributed system are that it reduces the communications overheads and thus overcomes the problem of limited communication bandwidth; it eliminates the effect of centralised computational bottlenecks, which makes this approach scalable. It can also adapt to the dynamic changes in the network structure and to the addition, or loss, of sensing nodes. These advantages, in view of the very nature of WSNs, with their limited resources and bandwidth, make the distributed algorithms more attractive and preferable to centralised algorithms.

2.1.5 Number of estimations

Localisation algorithms can be classified as either "single-estimation" or "successive-refinement" algorithms, based on whether the nodes estimate their position only once or iteratively. In the *single-estimation* type, when the node gets the required information it estimates its position, considers it as a final solution and stops requesting location information. Enhancing the accuracy of single-estimation algorithms could require special conditions. For example, the algorithm proposed in [30] requires a triangle placement of beacons in a certain location, which conflicts with the self-organising design objective that assumes that the localisation algorithm should be independent of global infrastructure and beacon placement [15], otherwise it could increase the computation cost, as indicated in [31].

In the second type (*successive-refinement* algorithms), localisation algorithms [32, 33] consist of two main phases: the initialisation phase, in which a node can get a rough estimation of its position, and the refinement phase, where each node iteratively broadcasts its initial position, then repeats the estimation using the new information to estimate a refined position. Successive-refinement algorithms could significantly improve the accuracy of the estimated position. On the other hand, they increase the messages propagated between nodes and the complexity of computations, and so the nodes consume more energy than the first type. Recently, several localisation algorithms have been proposed to overcome the drawbacks of the refinement approach, such as those given in [34, 35].

2.1.6 The set of references used

Localisation algorithms can be classified as either “all-references” or “subset-references” based on whether all references are used or not. Several optimisation techniques have been proposed, which are based on using all the available references to estimate the position of nodes, assuming that this approach (*all-references*) should lead to the most accurate estimation [36]. In contrast, significantly fewer algorithms have adopted the *subset-references* approach to optimise location accuracy, where a node uses only a subset of the available references. Increasing the number of references used will of course increase the complexity of the localisation algorithm[13], which conflicts with one of the most important design objectives of WSNs, namely minimising the computation cost in order to reduce energy consumption.

2.1.6.1 Selection of subsets

Different techniques have been proposed to select a subset of references, each of them aiming to fulfil one or more design objectives. One of the simplest techniques is to select the closest references as a subset [37], assuming that the estimation error would be lower for the nearest references. However, this assumption can only be true if the location estimation error comes from distance measurement alone, and the references have no location error. In fact, references further away could contribute to more accurate position estimation than closer ones. A more accurate approach that considers the two types of error (distance-measurement error and location error) has been adopted by many algorithms to select a subset of references with the lowest error [38, 39]. These algorithms enhance the accuracy of the estimated position; however, they require more computation and/or communication.

Some localisation algorithms select a subset of references based on the references' consistency by excluding the inconsistent references in order to increase the robustness and accuracy of location estimation [40]. The two algorithms proposed in [41] follow this approach to enhance security by detecting and removing malicious references. However, these algorithms require large memory space and the cost of computation is high.

The cardinality of the subset references can be specified either manually or dynamically. In the *manual* type the number of references is predefined manually at the time of design [13, 37], where a trade-off should be made between the simplicity and the accuracy of the algorithm. Simplicity requires a low number of references, but that could reduce accuracy, which can be improved by using more references. In the *dynamic* approach. the cardinality of the set of references used is specified at the run time, based on specific criteria. Each node may use a different number of references based on its neighbour references. For instance, a sensor node close to references with high localisation accuracy could use a low number of references, while sensor nodes surrounded by references with high localisation error need more references to handle this error.

The advantages of the dynamic approach are these. Firstly, it could make the selection process "smart" (i.e. enable each sensor node to specify the proper number of references that should be used to achieve a certain level of accuracy). Secondly, it enhances the robustness of the localisation algorithm because in a noisy environment the sensor node will dynamically increase the number of references used to overcome the existence of error. Thirdly, it improves the energy efficiency of the localisation algorithm; instead of continuing to use the same number of references, several sensor nodes will be able to use a lower number of references, which could reduce the computational and communication overheads.

In fact, most of the existing algorithms do not use selection methods but rather eliminate some references that satisfy (or do not satisfy) certain conditions. For example, in the algorithms proposed in [42, 43], the node eliminates the references that are out of its transmission range, while in [41] the node eliminates the references that could be malicious nodes. The main disadvantage of the elimination method is that the node could end up using all the available references without any elimination, or only an insignificant reduction. In contrast, the selection method initially selects the minimum number of the best references and then adds more references gradually until a stopping condition is achieved. For example, a sensor node could select only those references whose location error falls below a predetermined threshold [38]. The objective of the selection method is not only to achieve good accuracy but also to select the minimum number of references, which could assist in achieving several design objectives.

2.2 LOCALISATION SYSTEMS

2.2.1 Components of localisation systems

In a beacon-based localisation system, special nodes called beacons are required for location discovery. Beacons know their location through a GPS receiver or manual configuration. The rest of the nodes that have no knowledge about their location are called unknowns. As shown in Figure 2.1, localisation systems consist of three major components: distance/angle estimation, position computation and a localisation algorithm [18].

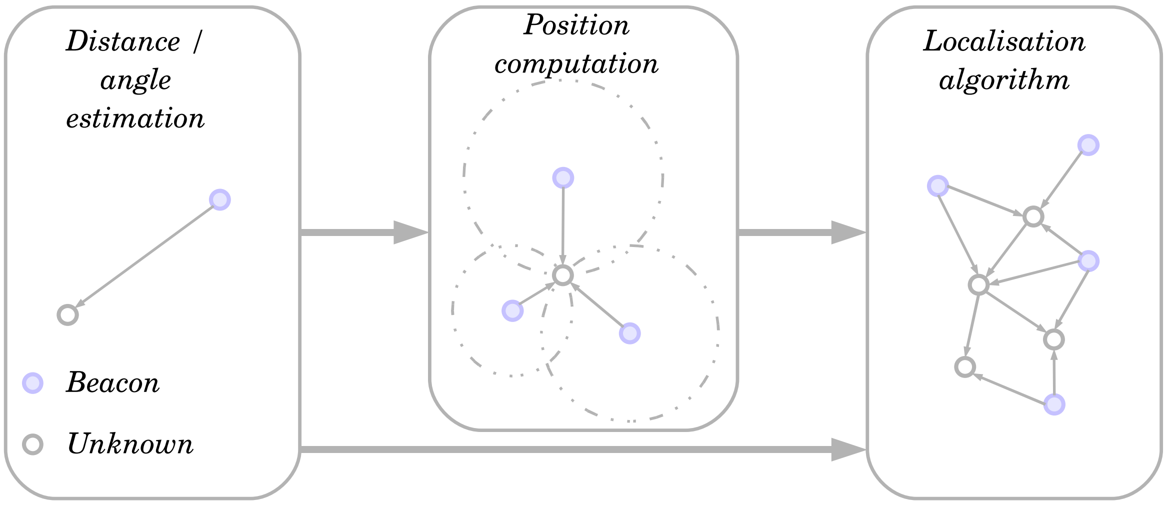


Figure 2.1. Components of localisation systems

2.2.1.1 Distance/angle estimation

This component is responsible for determining the physical relationship between two nodes, which can later be used to compute a node's location. Different approaches can be used for this purpose, such as directional antennas [44], radio frequency (RF) fingerprinting (communication neighbour authentication) [45], connectivity (in range) [46], and distance bounding [47]. Practically, these approaches use several techniques, including RSS, ToA, time difference of arrival (TDoA), angle of arrival (AoA) or round-trip time (RTT). This component will be discussed in more detail in Chapter 5.

2.2.1.2 Position computation

This component is responsible for computing the position of a node based on available information about the distance estimated from the previous component and position of references. Recognised techniques used in this component include triangulation [37], trilateration [48] and multilateration [24]. In the *triangulation* technique, an unknown measures AoA of at least three beacons and then uses the simple geometric relationships to estimate its position. One potential problem of the AoA approach is the expense of equipment to obtain precise angle estimates [49]. *Trilateration* also uses the geometry of a triangle to estimate nodes' position. However, instead of using AoA, it uses the location of and the distance to at least three beacons. The *multilateration* technique estimates location by solving the mathematical intersection of multiple hyperbolas [12]; it is also based on the location of and the distance to three or more beacons. The proposed localisation algorithm follows the multilateration technique.

2.2.1.3 Localisation algorithm

This is the main component of a localisation system. It determines how the available information will be manipulated in order to enable most or all of the nodes of the WSN to estimate their position. These algorithms can be centralised (global) or distributed. The *centralised* algorithms [50-52] are powerful and estimate the nodes' position with high accuracy. However, they have a high communication and computational requirement, which is usually not available in WSNs. To reduce the communication overhead, various *distributed* localisation algorithms have been proposed, which decompose the global estimation system into sub-systems and then iterate over these sub-systems. Several iterative techniques have been followed. For instance, [53] uses references' location information and local computation to localise unknown nodes iteratively, while [28] uses shortest-path approximation to the reference node to approximate Euclidean distances. The third technique uses local refinement [42], which requires an initial solution. The disadvantage of iterative techniques is the effect of error propagation and accumulation, which is less prominent in centralised algorithms.

2.2.2 Multilateration method

By using the multilateration method, a node within the range of at least three beacons can estimate its position by minimising the differences between the measured distances and the estimated Euclidean distances in order to obtain the minimum mean square estimate (MMSE) from the noisy distance measurements. As shown in Figure 2.2, a sensor node has a set of m reachable beacons with the following information (x_j, y_j, d_j), where (x_j, y_j) is the location of beacon j and d_j is the measured distance to it. Assuming that $(\hat{x}, \hat{y})$ is the estimated position of the sensor node, the error of the measured distance to beacon j $(1 \leqslant j \leqslant m)$ can be represented as

$$d_j - \sqrt{(\hat{x} - x_j)^2 + (\hat{y} - y_j)^2} \quad . \tag{2.1}$$

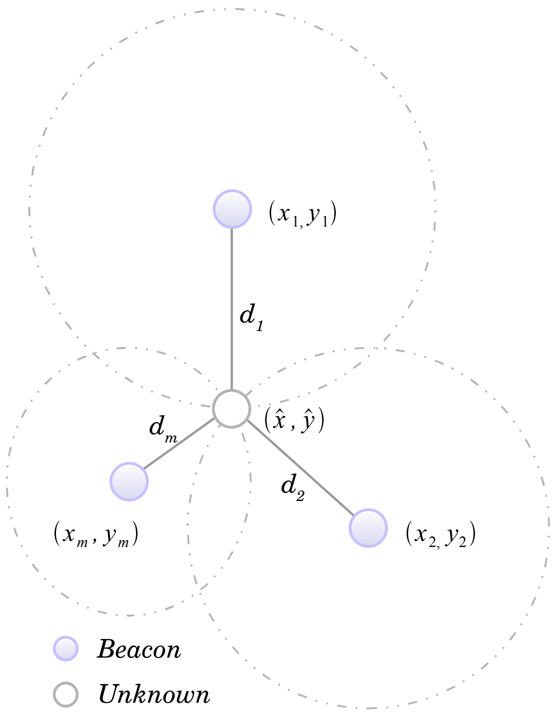


Figure 2.2. Multilateration

This system of equations can be solved to estimate the location $(\hat{x}, \hat{y})$ by using the matrix solution for MMSE [36] given by:

$$b = (X^T X)^{-1} X^T Y \tag{2.2}$$

where

$$b = \begin{bmatrix} \hat{x} \\ \hat{y} \end{bmatrix}$$

$$X = \begin{bmatrix} 2(x_1 - x_2) & 2(y_1 - y_2) \\ 2(x_1 - x_3) & 2(y_1 - y_3) \\ \vdots & \vdots \\ 2(x_1 - x_m) & 2(y_1 - y_m) \end{bmatrix}$$

$$Y = \begin{bmatrix} t - x_{٢}^{٢} - y_{٢}^{٢} + d_{٢}^{٢} \\ t - x_{٣}^{٢} - y_{٣}^{٢} + d_{٣}^{٢} \\ \vdots \\ t - x_m^{٢} - y_m^{٢} + d_m^{٢} \end{bmatrix}$$

$$t = x_1^2 + y_1^2 - d_1^2 \quad .$$

2.2.3 Assumptions and variables

From the perspective of localisation systems, there are four types of sensor nodes: beacons (B) with *a priori* known location; unknowns (U) to be localised; knowns (K) that have already estimated their position; and references (R), which are willing to help other nodes to estimate their position. These four set of nodes can be defined as follows:

$B = \{b_j, \text{where } j \in \{1,2,\ldots,C(B)\}\}$

$U = \{u_i, \text{where } i \in \{1,2,\ldots,C(U)\}\}$

$K = \{k_i, \text{where } i \in \{1,2,\ldots,C(K)\}\}$

$R = B \cup \bar{K}, \text{where } \bar{K} \subseteq K \ . \quad R = \{r_j, \text{where } j \in \{1,2,\ldots,C(R)\}\}$

where $C(\ )$ is the cardinality of a specific set. Several localisation algorithms assume that $R = B \cup K$; however, in the proposed algorithm the known node cannot act as a reference unless it satisfies certain conditions. The notation n_i will be used to refer to either an unknown sensor node that would like to estimate its position or a known sensor node that would like to refine its position $(n_i \in \{U \cup K\})$. Without loss of generality, the localisation will be employed for a network in a 2-D plane. It is assumed that the sensor nodes are range nodes producing distance measurements $\hat{d}_{i,j}$ (between node n_i and reference r_j) by measuring the RSS of radio signals, while the actual distance is $d_{i,j} = \|z_i - z_j\|$, where $\|\ \|$ is the Euclidean norm and z is the actual location. The node n_i can estimate its position $\hat{z}_i = (\hat{x}_i, \hat{y}_i)$ if it knows the location of at least three references and the distance to them, which could be different from the actual location, $z_i = (x_i, y_i)$, then it could act as a reference for other nodes. Since only the local information is considered, the node n_i will consider only the reachable references within its range, i.e.

$$R_i = \{r_j, \text{where } d_{i,j} \leqslant r_{tx}\} \tag{2.3}$$

where r_{tx} is the transmission range of the reference node. After a period of time this set (R_i) will consist of a large number of references. Using all of them could improve the accuracy of location, but on the other hand it will increase the complexity, computation cost, required time and energy consumption of the localisation process. This situation leads to the need for using the best subset of references S_i, where $S_i \subseteq R_i$, in terms of localisation accuracy, without compromising the security, simplicity, applicability, resource constraint and communication cost of the localisation algorithm.

2.2.4 Localisation errors

Location discovery is based on physical measurements, which may be significantly inaccurate owing to several types of errors. Therefore it is crucial to consider error sources and error propagation in order to design an accurate location-discovery method. Five sources of error that influence the localisation performance in WSNs were identified in [16], namely:

1. Measurement
2. Finite precision
3. Objective function specific
4. Intractable optimisation tasks
5. Localised algorithms.

Measurement errors arise from limitations of sensing technology, the instability of phenomena and environmental noise. Finite precision is present in all computing systems, and it is important in WSNs because of their constrained resources. This leads to the need for a simple algorithm which will reduce the error from this source. Errors 3 and 4 are caused by optimisation issues. The final error is unique to localised algorithms because of lack of global knowledge.

In fact, these sources can cause mainly three types of error, as shown in Figure 2.3. Firstly, computation error (e_i^c) comes from the node that performs the estimation; secondly, location error (e_j^l) arises from the references used; and thirdly, distance-measurement error $(e^{d}_{i,j})$ occurs between the node and the references used. When the node n_i estimates its position using R_i set of references, the resulting total error (e^{t}_{i}) can be represented as a function of these three errors as $e^{t}_{i}=f\,(e^{c}_{i}, e^{d}_{i,j}, e^{l}_{j}),\ where\ j \in R_i$.

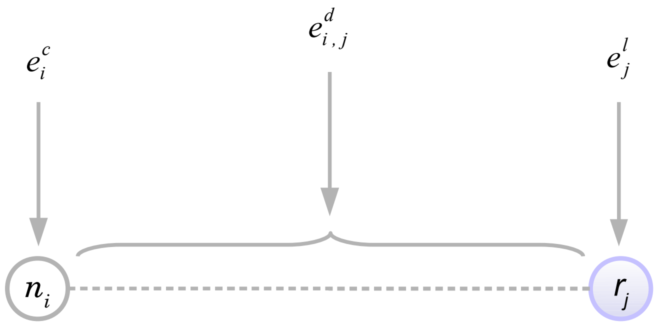


Figure 2.3. Localisation errors

This total error represents the location error of node n_i $(e^{l}_{i}=e^{t}_{i})$. Iterative localisation methods may suffer from the impact of error accumulation and propagation. Node n_i could become a reference r_i for other neighbouring nodes. Its error will affect not only these neighbours but could also affect those nodes using these neighbours as references. If there is no error-control mechanism, this could lead to unbounded localisation error for large WSNs. To illustrate the effect of error propagation, one can consider the simple network shown in Figure 2.4.

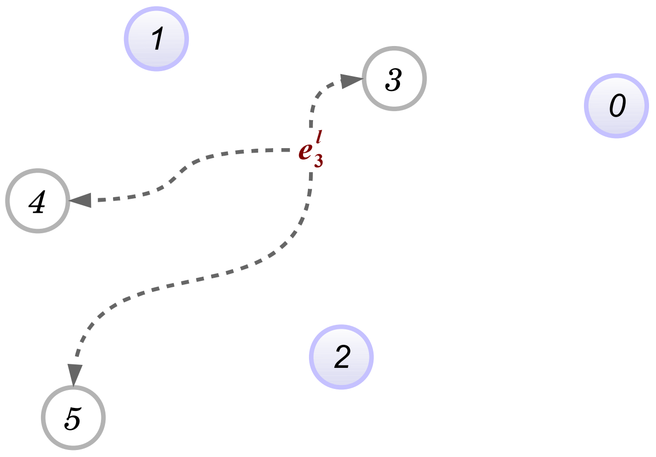


Figure 2.4. Error propagation

From Figure 2.4, R and U can be defined as $R=\{r_{0,} r_{1,} r_{2}\}$ and $U=\{u_{3,} u_{4,} u_{5}\}$. Node u_3 first estimates its position using $R_{٣}=\{r_{٠,} r_{١,} r_{٢}\}$, u_4 uses $R_4=\{r_{1,} r_{2,} r_{3}\}$ and then u_5 uses

$R_5 = \{r_1, r_2, r_4\}$. For the sake of simplicity, the total error can be considered as the summation of the three errors. Then the errors of position estimation are:

$$\boldsymbol{e_3^l} = e_3^c + \sum_{j=0}^{2} e_j^l + e_{3,j}^d \tag{2.4}$$

$$e_4^l = e_4^c + (\boldsymbol{e_3^l} + e_{4,3}^d) + \sum_{j=1}^{2} e_j^l + e_{4,j}^d \tag{2.5}$$

$$e_5^l = e_5^c + (\underbrace{e_4^l}_{e_4^c + (\boldsymbol{e_3^l} + e_{4,3}^d) + \sum_{j=1}^{2} e_j^l + e_{4,j}^d} + e_{5,4}^d) + \sum_{j=1}^{2} e_j^l + e_{5,j}^d \; . \tag{2.6}$$

The location error of node *u₃* (e_3^l) also affects the position estimation of nodes *u₄* and *u₅*. Therefore, in order to get an accurate localisation system, one should develop a localisation method that takes all three types of error and their impact into consideration and does not deal with only one of them.

The purpose of this section is to classify and introduce the three types of error that could affect the position estimation. In addition, it is to show the impact of error accumulation and propagation on the iterative localisation algorithms. However, investigating error characteristics and modelling is beyond the scope of this study. Readers who wish to do so can refer to the literature on this type of investigation, such as [16, 24, 27].

2.3 APPROACHES TO SELECTING A SUBSET OF REFERENCES

As mentioned earlier in the previous chapter, designing an efficient localisation algorithm for WSNs does not encourage using all of the available references. A localisation algorithm should first select those references with the potential of contributing more to high accuracy. Different approaches have been used to select a subset of references. This section will analyse only a number of existing approaches, highlight their merits and weaknesses and then compare these approaches.

2.3.1 Nearest references

This is a very simple approach, which is based on choosing the nearest references as a subset to estimate a node's position, assuming that the estimation error would be higher for distant references than for near ones. This approach could improve the accuracy of position estimation in WSNs. Cheng *et al.* [37] propose a localisation algorithm called APS (Near-3), which is a modification of the original ad-hoc positioning system (APS) [53], which considers all the available beacons during the position estimation. The new, improved APS algorithm simply chooses the nearest three beacons to the unknown node inside the original APS computation (i.e. the triangulation mechanism and least square method) in order to estimate the unknown node position. The simple heuristic used to select the best beacons requires much fewer communication overheads than to the original APS approach. [54, 55] assign a different weight to each reference, depending on its estimated distance from the unknown node, with a higher weight to the near references. However, these algorithms can be modified to select a subset of weighted references by assigning a weight equal to zero for distant references.

This approach assumes that the estimation error would be higher for distant references than for near ones and that the estimation error comes only from the distance measurement and ignores neighbour location error (because it only uses beacons that have no, or low, location error). Logically, if near references with location estimation errors are to be used, this assumption will not be valid and distant references could make a better contribution to position estimates than near ones.

2.3.2 Low-error references

A localisation error results mainly from two sources: location error, which is the error in neighbouring nodes' position, and distance error, which is the error in the distance measurement. Iterative techniques that may be used by localisation algorithms propagate this error, and so references that have large errors contaminate their neighbours' location estimate. Using a reliable subset of references that consists of references with a low error rate will prevent this type of contamination.

This technique has been used by Liu *et al.* [24], where an unknown node computes the total error of its neighbour references, which is the sum of the location error and distance error. Then it ranks references in an ascending order based on their error. Finally, it selects references with an error below a certain threshold and discards the others. Sinha and Chowdhury [38] propose that a localisation algorithm should choose a subset of three references in such a way that the error in the estimated location is within a certain limit. However, this algorithm requires high computational complexity. Selecting references in [39] is also based on this approach.

2.3.3 Malicious node removal

An attacker may provide an incorrect location reference to unknown nodes, which will then estimate their locations incorrectly. The malicious node removal approach aims to keep as many benign location references as possible, while the malicious ones are removed, resulting in a more accurate position estimation. The authors of [41] investigated two types of attack-resistant techniques to target malicious attacks against range-based location discovery in WSNs. In the first technique, the unknown nodes defeat malicious attacks by checking the consistency of references and then removing the inconsistent malicious references. This technique starts by using the entire set of references and then it gradually removes the most suspicious references till it reaches a certain level of consistency, which depends on the measurement error of an estimated location. The authors developed an incremental MMSE approach to reduce the computation cost, but it increases the size of the required memory.

The second technique is called voting-based location estimation, which quantises the deployment field into a grid of cells, and then the unknown node determines how likely it is to be correct in each cell, based on each reference. After the unknown node has processed all references, it chooses the cell(s) with the highest vote, and uses its (their) geometric centroid as the estimated location of the sensor node. However, specifying the voting by each reference at each cell of the grid requires a high computation cost. Liu *et al.* [56] follow the same approach in their localisation algorithm.

2.3.4 Consistency of references

This approach selects a subset of references based on their consistency with each other and excludes the inconsistent ones in order to increase the robustness and accuracy of the location estimate. One of the techniques to find the degree of consistency of each reference is to find the reference location error with respect to other references. This is the sum of the squared differences between the calculated distance and the estimated distance from one reference to the rest of them.

Albowicz *et al.* [40] propose a localisation algorithm for choosing a reliable subset of references based on a reference consistency approach. The algorithm starts when the unknown node gathers information from neighbour references, which includes their degree of consistency (in [40] termed “residual value”), and then the unknown node chooses only those references with the highest degree of consistency to estimate its location. While most of the unknown nodes should manage to get their position estimate, only the most accurate should extend system coverage and become references, in order to prevent incorrect convergence and divergence. Liu *et al.* [41] also use this approach to identifying the malicious references.

2.3.5 Impact of geometry

This approach excludes insignificant references from participating in the localisation estimate, based on the geometry of references. Geometry could have a greater impact on accuracy of localisation than distance between references and unknowns. The Cramer-Rao-Lower-Bound (CRLB), which was defined by Patwari *et al.* [57], can be used to specify the impact of geometry in order to quantify and compare the contribution of each reference to the accuracy of localisation and then to be able to choose a subset of references that contribute most to the accuracy.

The Local-CRLB algorithm, which is proposed in [13], considers the impact of geometry. Local-CRLB starts when an unknown broadcasts a request for localisation. The neighbour references receiving the request estimate their distance to the unknown, which can be used in addition to the CRLB to assign beacons a probability of response. Responses, which include the originator’s address, location and distance estimate to the unknown, are broadcast. Subsequent beacons can use the additional information provided by the former responses. Local-CRLB constitutes a significant improvement over the algorithms selecting the nearest beacons as a subset. However, this algorithm assumes ideal estimation of distances, which is a strong assumption that would never be available in real-life applications. Lieckfeldt *et al.* [58] investigate the Local-CRLB algorithm by considering energy consumption and impact on accuracy of localisation, using a maximum-likelihood estimator (MLE).

2.3.6 Noisy distance estimate

In the realistic case, the distance estimate is corrupted by noise and so localisation algorithms using only a distance estimate (e.g. those based on the nearest-references approach) to select neighbouring references could tend to select references whose estimate distance is shorter than the true distance. This approach considers a noisy distance estimate in order to remove bias from location estimates even in high-noise environments. Costa *et al.* [42] propose a localisation algorithm called distributed weighted multidimensional scaling (dwMDS). dwMDS selects a subset of references based on a noisy RSS distance estimate and small neighbourhoods in order to avoid the biasing effect of a noisy environment. The proposed algorithm consists of two steps. In the first step, it finds the estimated node location based only on a distance estimate. In the second step, it excludes neighbours with a high biasing effect to construct a subset of references that require fewer iterations to converge to an accurate position estimate. The authors of [43] modify the dwMDS algorithm by simplifying the computation and reducing the processing time. [23] also used this approach.

2.4 COMPARISON OF THE ANALYSED APPROACHES

Each of these approaches has advantages and disadvantages and it is not possible to consider one of them as the best approach for every application, scenario or network. The selection of one of these approaches to be implemented in WSNs is a little more complicated because of resource limitations. When deciding which approach will be used, several issues should be considered, such as available resources, security level, computational cost, time of convergence and accuracy level. For example, if the designer would like to use minimal resources and is concerned about the execution time and computational cost, then the nearest-references approach is a possible choice. If the localisation algorithm is to be used for WSNs in a hostile environment, then the security level is an important issue and so malicious node removal and references consistency can be considered. The noisy distance estimate approach can be selected for WSNs with high noise to avoid the biasing effect of a noisy environment. A designer who would like to estimate position with high accuracy could choose one of the following approaches: the low-error references or the noisy distance estimate approach. However, one who is also looking for lower time of convergence could select the low-error references approach. On the other hand, the designer should also consider the limitations of each approach. For instance, the nearest references approach is very simple but cannot achieve a high level of accuracy compared with other approaches. The malicious node removal and references consistency approaches require higher computational cost, and the noisy distance estimate approach requires higher time of convergence.

A comparative summary is provided in Table 2.1. This table highlights some of the advantages and disadvantages of the analysed approaches. The last two fields of this table (targets and limitations) could be used as a guideline to help the designer to select an applicable approach that would be more suitable for his specific system requirements. Targets represent the issues that can be achieved using the corresponding approach, while limitations indicate the issues that cannot be achieved (or not completely fulfilled).

Table 2.1. Comparison of the analysed approaches

Approach	Advantages	Disadvantages	Targets	Limitations
Nearest references	-Very simple -Low computation -Few references	-Does not consider references' location error	-Resources usage -Computation cost -Convergence time	-Security level -Accuracy level -Noise level
Low-error references	-Accuracy -Few references	-Computationally intensive	-Accuracy level -Convergence time	-Computation cost -Security level
Malicious node removal	-Works in hostile environment -Accuracy	-Computationally intensive -Large memory -Elimination criteria	-Security level -Accuracy level	-Resources usage -Computation cost
References consistency	-Works in hostile environment -Accuracy -Few references	-Computes the consistency of each reference	-Security level -Accuracy level -Convergence time	-Computation cost
Impact of geometry	-Accuracy -Few references	-Assumes ideal estimation of distances	-Accuracy level -Convergence time	-Security level -Noise level
Noisy distance estimate	-Works in noisy environment -Accuracy	-Elimination criteria	-Accuracy level -Noise level	-Security level -Convergence time

2.5 CHAPTER CONCLUSIONS

Various categories of localisation algorithm were analysed to show the motivation behind adopting specific categories for the proposed localisation algorithm. That does not mean the other categories do not have advantages. However, the categories adopted here will help to accomplish several design objectives, as will be seen in the next three chapters. It is emphasised that localisation algorithms for WSNs should use a subset of references, rather than using *all* of the available ones. However, selecting the proper subset of references is a very challenging task. Several localisation algorithms use various approaches to select a subset of references. A comparison of these approaches was briefly presented, highlighting some of their strengths and weaknesses. The main objective of localisation algorithms is to estimate nodes' position with high accuracy without compromising other design objectives.

Therefore, the low-error references approach was adopted in the proposed selection method. This approach was modified in order to overcome its limitations. Finally, it can be concluded that, despite significant research into the development of localisation systems, developing a localisation algorithm for WSNs by carefully selecting a sufficient number of the best references in order to enhance the accuracy of position estimate at reduced cost, is still a challenge and an open area for future investigation.

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

Chapter 3

ACCURATE LOCALISATION SYSTEM

Several localisation algorithms rely on using all or most of the available references to enhance their performance. In contrast, the ALWadHA (**a**n efficient **l**ocalisation algorithm for **w**ireless **ad** hoc sensor networks with **h**igh **a**ccuracy) does not rely on using a high number of references to enhance the accuracy of estimation; rather it relies on using a smart reference-selection method. ALWadHA selects almost the minimum possible number of references that could contribute most to high accuracy. In order to evaluate and compare ALWadHA with other localisation algorithms, the network simulator, ns-2, was used. The current version of ns-2 was extended to simulate wireless sensor networks and, mainly, the localisation system by adding new modules. The extended ns-2 has a user-friendly interface, which enables a normal user, who has basic knowledge of simulating a simple wireless network, to simulate the proposed localisation system without any extra knowledge. The ALWadHA algorithm was evaluated with respect to the effects of network size and the deployment and density of nodes on the location estimation error and the number of references used. The researcher also examined the impact of increasing the distance-measurement error on the accuracy of the ALWadHA.

3.1 ALWADHA ALGORITHM

An ALWadHA has been developed to enhance the accuracy of position estimation. The idea is to select those references that are willing to help other nodes estimate their position. Based on this method, the node will only select those references that could contribute most to an accurate position estimate, and will eliminate irrelevant references from participating in the final position estimate.

Table 3.1. ALWadHA localisation algorithm

```
1. Initialisation
   If (final = true) then exit
   Broadcast “location request” messages
   Receive “location response” messages from neighbouring references ($R_i$)
   If $(C(R_i) < 3)$ then exit.
2. Initial position estimation
   Select a subset of references $S_i$ from $R_i$
   Measure distance to the references in $S_i$
   Apply MMSE to determine an initial position $\hat{z}_i^0$ .
3. Refined position estimation
   for $(j = 1$ to $C(S_i))$
       $\hat{e}_{i,j}^d = | \|\hat{z}_i - \hat{z}_j\| - \hat{d}_{i,j} |$
       if $(\hat{e}_{i,j}^d > e_{max}^d)$ then (enhancement = true); break.
   If (enhancement = true)
       for $(j = 1$ to $C(R_i))$
           $\hat{e}_{i,j}^d = | \|\hat{z}_i - \hat{z}_j\| - \hat{d}_{i,j} |$
           if $(\hat{e}_{i,j}^d > e_{max}^d)$ then eliminate $r_j$.
       Estimate refined position $\hat{z}_i$ as shown in 2
   else
       $\hat{z}_i = \hat{z}_i^0$ .
4. Position update
    $D_{acc} = \sum_{j \in S_i} | \|\hat{z}_i - z_j\| - \hat{d}_{i,j} |$
   if $(D_{acc}^k < D_{acc}^{k-1})$
       $\hat{z}_i$ will be accepted
       if $(D_{acc} < T_{acc})$ then (final = true).
```

The ALWadHA localisation algorithm consists of four phases, as shown in Table 3.1. In the first phase, the node collects information from nearby references. In the second phase, the node selects a subset of references to estimate its initial position. In the third phase, the node checks the possibility of improving the current position. In the final phase, the node decides if it will accept this position, and if the accepted position can be considered as a final estimate.

The ALWadHA algorithm follows two types of optimisation. Firstly, the node increases the number of participating references till it reaches a certain level of accuracy based on location error. Secondly, the node eliminates irrelevant references, based on distance error, which could bias the estimated location toward incorrect references.

3.1.1 Initialisation

Beacons assign their probability of accuracy to one $(P_{acc} = 1)$ (the probability of accuracy will be introduced in Section 3.1.2.1). Each reference holds the information required to localise other nodes, which is the location (or location estimate) and the probability of accuracy. A node initiates localisation by broadcasting a "location request" message, which includes the required accuracy level (L_{acc}), to the first neighbours (one-hop neighbours). The accuracy level of unknowns is equal to zero, while for knowns it is equal to the minimum probability of accuracy in the subset S that they used to estimate their position.

$$L_{acc} = \begin{cases} 0 & \text{unknowns} \\ \min\limits_{j \in S} P_{acc}^{j} & \text{knowns} \end{cases} \tag{3.1}$$

References receiving a request from an unknown will respond with a "location response" message containing the tuple $\{z \text{ (or } \hat{z}), P_{acc}\}$. However, if the request came from a known, then references will not respond unless their probability of accuracy is higher than the required accuracy level and also higher than the probability of response $(P_{acc} \geqslant L_{acc} \text{ AND } P_{acc} \geqslant P_{res})$; this response mechanism is shown in Figure 3.1.

The advantages of a response mechanism include:

- In addition to beacons, only knowns with a high probability of accuracy will act as references, which could minimise incorrect convergence.
- The time of response is specified to be only when it could contribute to a more

accurate estimate.

- The number of response messages is reduced, which reduces the communication cost.

- Knowns will not re-estimate their position unless the new estimate could enhance the accuracy of their current position, thus reducing the computation cost.

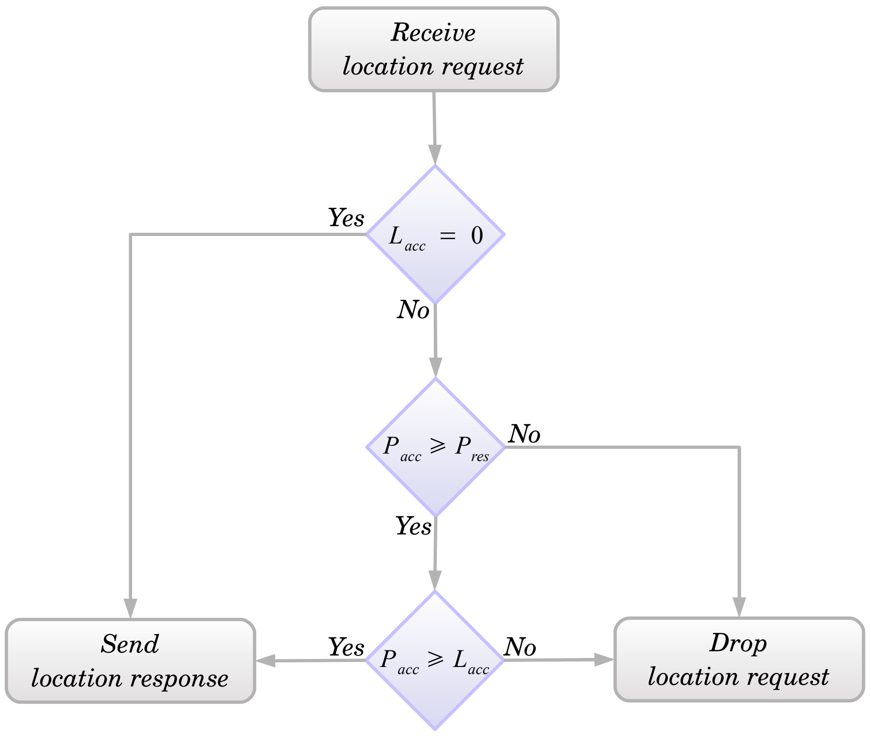


Figure 3.1. Response mechanism

The requesting node could receive several “location response” messages from neighbouring references (R_i), which it ranks in a descending order list based on their probability of accuracy. Each record of this list consists of a reference's *id*, location and probability of accuracy

$$R^i_{list} = \{id_j,\ z_j\ (\text{or } \hat{z}_j),\ P^j_{acc}\} \tag{3.2}$$

which will be used in the next phase to select a subset of references S_i.

3.1.2 Initial position estimation

This phase consists mainly of two parts. In the first part, the node selects the most accurate subset of references S_i using smart reference-selection method. In the second part, it measures the distance to each reference on this subset using RSS. Then the node applies the MMSE method to estimate its initial position $(\hat{z}_i^{\cdot})$.

3.1.2.1 Smart reference-selection method

The key idea is to select a subset of references S where $S \subseteq R$ could contribute to an accurate location estimate, i.e. the selection method should satisfy the following two conditions. Firstly, it should not cause a high computation error. Secondly, it should select only those references that have relatively low location and distance errors. To satisfy the first condition, the selection method should have the following characteristics:

- Is very simple
- Selects the minimum possible number of references.
- Deals with location and distance error separately.

Because of these characteristics the computation cost of the selection method is low, which could reduce the *computational error*. To satisfy the second condition, the references are selected on the basis of what are called accuracy levels, where the node selects a subset of references that could have the minimum *location error*. In addition, the next phase (refined position estimation) deals with *distance-measurement error*.

Initially, one can ignore the distance error $(e^d = 0)$ and the total error can then be represented as:

$$e^l = e^c + \sum_{j \in R} e_j^l \quad . \tag{3.3}$$

The goal is to select a specific subset of references S that minimises the location error over all possible subsets. If R is the set of available references with cardinality m, then $\varsigma = \{S^1, S^2, S^3, \ldots, S^C\}$ is the set of all possible subsets from R, where C is the cardinality of ς and the cardinality of $S^k \in \varsigma$ is c, where $c \geqslant 3$. The number of combinations to select a subset S^k, with cardinality c, from R is:

$$\frac{m!}{c!\,(m-c)!} \tag{3.4}$$

and so the total number of possible subsets in ς is:

$$C(\varsigma) = \sum_{c=3}^{m} \frac{m!}{c!\,(m-c)!} \quad . \tag{3.5}$$

After finding all of these subsets, S can be specified as follows:

$$S = S^k, \;\text{where}\, S^k \in \varsigma \;\text{ and }\; S^k \text{ has } min\ (\sum_{j \in S^k} e_j^l) \tag{3.6}$$

where S represents the subset of minimum location error that could contribute most to accurate location. However, finding this subset requires an exhaustive search and expensive computations, which makes this approach not practically feasible for very resource-constrained networks such as WSNs. Therefore, a different approach is applied to find this subset.

The references are divided into several levels based on their location accuracy, starting with level 0, which includes all beacons with a location error equal to zero $(e^l = ٠)$. Level 1 consists of nodes that use only beacons to estimate their position, i.e. their location error is caused by computation error only $(e^l = e^c)$. Level 2 consists of nodes that use one reference from level 1 and the rest of the references are from level 0; their location error is $e^l = e^c + e_j^l$, where $j \in \text{level } 1$. This carries on with the rest of the nodes based on the reference level used in their position estimation. To take advantage of these levels, a probability of accuracy (P_{acc}) will be assigned to each level. Beacons at level 0 have the maximum probability of accuracy $(P_{acc} = 1)$, while the probability of accuracy of other references can be estimated as follows:

$$P_{acc} = ١ - \frac{١}{\sum_{j \in S} P_{acc}^j} \tag{3.7}$$

where P_{acc}^j is the probability of accuracy of reference r_j, and so each reference keeps not only its location but also the corresponding probability of accuracy. The probability of accuracy will not be used in the position estimation, but it is used to distinguish between the levels of references and to allow nodes to rank the references on the basis of these levels.

As mentioned in the previous section, after receiving the response messages, the node ranks references in a descending order list (R^i_{list}), based on their probability of accuracy. The node selects the first three references in R^i_{list} and then calculates the probability of the accuracy of S_i using Equation (3.7). If P_{acc} is less than a certain value, the next reference will be added to S_i and P_{acc} is recalculated. The node carries on with this process till the subset S_i satisfies a certain value of probability of accuracy $(P_{acc} \geqslant P_{min})$, otherwise it will stop and postpone the estimation to the next iteration, where more accurate references could be available. Figure 3.2 shows the proposed smart reference-selection method.

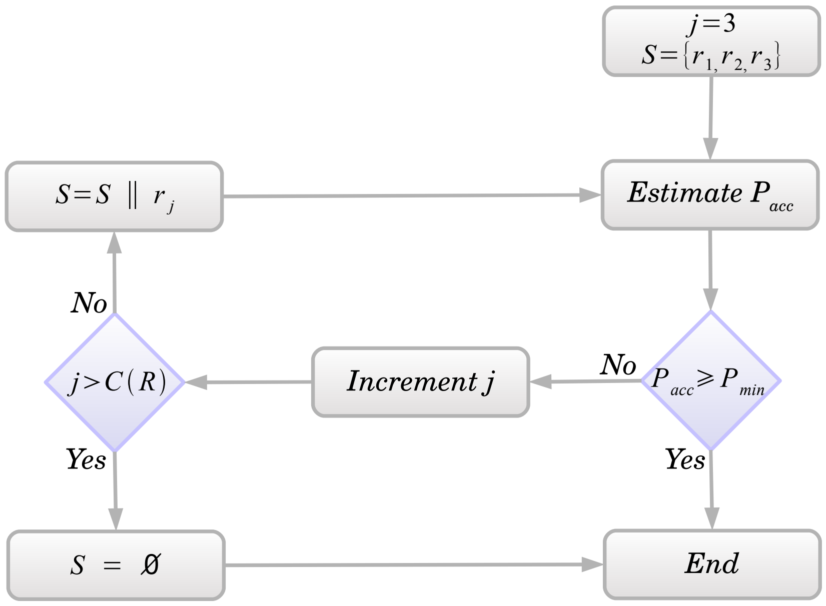


Figure 3.2. The smart reference-selection method

The smart reference-selection method has the following characteristics:

- It selects references that could have the lowest location error.
- The estimation of P_{acc} requires very simple computation.

- It is not based on measured distance, which could be corrupted by multiplicative noise.

- It specifies the time of estimation; nodes that have a more accurate subset will estimate their position first. The technique followed for this purpose does not require any interaction or message broadcasting between nodes; rather, it is based on available information.

- Rejecting a position estimate with low accuracy could delay the time required by a node to know its position, but on the other hand it could enhance the position accuracy and minimise the incorrect convergence due to cumulative error.

3.1.2.2 Cardinality of subset references

Several localisation algorithms that are based on the subset-references approach focus on improving the selection method. However, specifying the proper number of references that should be used to guarantee a certain level of accuracy has rarely been studied in the literature. The cardinality of the subset references can be specified either manually or dynamically, as explained in Section 2.1.6.1. The proposed algorithm is based on the dynamic approach, where the node starts by using the minimum possible number of references $(c=3)$, and then increases this number gradually until the selected subset of references achieves the minimum level of accuracy

$$S = S^k \text{ , where } S^k \text{ has } min(c) \text{ and } \left(1 - \frac{1}{\sum_{j \in S^k} P^j_{acc}}\right) \geqslant P_{min} \quad . \tag{3.8}$$

The cardinality of S could vary from one level to another, for instance at level 1 the cardinality of S is equal to three $(c=C(S)=3)$, while it will be increased in the higher levels, because at level 1 the node uses only beacons with no location error, while on the higher levels references with location error start to participate in the estimation and so the node increases the number of references slightly to handle this error.

3.1.3 Refined position estimation

In the previous phase the node selected the subset of references based only on location error. In this phase the node enhances the accuracy of position estimation by considering the distance error, where references with high distance error are eliminated from R^i_{list} . This phase starts by checking the possibility of enhancement based on the estimated distance error $(\hat{e}^d_{i,j})$, which is the difference between the calculated distance (between the node's initial position $(\hat{z}^0_i)$ and references position $(z_j$ or $\hat{z}_j)$) and the measured distance $(\hat{d}_{i,j})$

$$\hat{e}^d_{i,j} = |\ \| \hat{z}^{\cdot}_i - \hat{z}_j \| - \hat{d}_{i,j}\ |, \quad \text{where } j \in S_i \ . \tag{3.9}$$

If the estimated distance error for at least one reference in S_i is greater than a certain value $(\hat{e}^d_{i,j} > e^d_{max})$, this indicates that position enhancement is required, otherwise the initial position is regarded as an accurate position $(\hat{z}_i = \hat{z}^{\cdot}_i)$ and the process continues to the next phase. To enhance the position estimation, the node eliminates those references that have $\hat{e}^d_{i,j} > e^d_{max}$, where $j \in R_i$ from R^i_{list} , then a new subset of references will be selected to estimate a refined position $(\hat{z}_i)$ as described in the previous phase. It is remarked that position refinement is not required at each iteration; it is only required when at least one of the references in the subset S_i has a high distance error. Checking the estimated distance error not only enhances the accuracy of estimation but could also be used to detect the existence of malicious nodes.

3.1.4 Position update

In the final phase, the node performs two tasks. Firstly, it checks the acceptance of the estimated position. Secondly, it applies the termination criterion. To check the acceptance of the estimated position, the node computes the position's degree of accuracy (D_{acc}) as follows:

$$D_{acc} = \sum_{j \in S_i} |\ \| \hat{z}_i - z_j \| - \hat{d}_{i,j}\ | \ . \tag{3.10}$$

If the new degree of accuracy is better than the one from the previous iteration

$(D_{acc}^{k} < D_{acc}^{k-1})$, it will accept the new estimated position, otherwise it will be rejected. This test could ensure that the new accepted position is more accurate than the previous one, and it furthermore screens out incorrect convergence and divergence.

Finally, if the position is accepted and its degree of accuracy is less than the accuracy target $(D_{acc} < T_{acc})$, the node considers this position as a final one and stops sending "location request" messages. Following this approach for termination has the following advantages:

- Nodes that have a subset of references with high accuracy will terminate their position estimation at an early stage, which could reduce the computation and communication cost.

- More time will be allowed for those nodes with a low accuracy subset of references to enhance their position estimate.

Compared with the time-based or iteration-based termination approaches, where the nodes terminate the localisation process after a certain time or a specific number of iterations, the approach used could outperform these two approaches in terms of computation, communication and the accuracy of final position estimate.

3.2 IMPLEMENTATION

The difficulties of setting up a WSN with real nodes and the infeasibility of analysis make simulation an essential tool to study WSNs. Simulation is widely used in system modelling for applications ranging from engineering research, business analysis and manufacturing planning to biological science research [59].

3.2.1 Network simulator (ns-2) overview

Ns-2 [60] is an open-source event-driven simulator designed specifically for research in networks. Ns-2 was developed in C++ and uses Object-oriented Tool command Language (OTcl) as configuration and script interface (i.e., a front-end). Each language has two types of classes. The first type includes the standalone C++ and OTcl classes that are not linked together. The second type includes classes that are linked between the two languages.

These C++ and OTcl classes are called *compiled hierarchy* and *interpreted hierarchy*, respectively. These two hierarchies are linked together using an OTcl/C++ interface called TclCL [61].

Ns-2 provides substantial support for simulation of TCP, routing, and multicast protocols over wired and wireless networks. Recently, Morávek [62] investigated the ns-2 capabilities in node localisation in wireless networks. This investigation shows that ns-2 supports simulations of different localisation techniques (such as TOA and RSS). Ns-2 has several tools and modules that researchers can use to develop localisation schemes. The researchers can modify the existing modules, or create new modules from the very beginning, and ns-2 allows researchers a high level of independence from the designed framework of the simulator. Therefore, the ns-2 simulator will be used to implement and to evaluate the proposed localisation algorithm.

3.2.2 The extended ns-2

Ns-2 contains several flexible modules for energy-constrained wireless ad-hoc networks, which encourages researchers to use ns-2 to investigate the characteristics of WSNs. However, to implement and evaluate the proposed localisation algorithm, the current ns-2 version (ns-2.34) should be extended and new modules should be added. Figure 3.3 shows the new classes that were added to the ns-2. These classes can be divided into two types. Firstly, there are *standalone* classes, which are MMSE, Position and AlwadhaPosition classes. These classes are used only from the C++ domain. Secondly, there are *compiled hierarchy* classes, which are LocDisApp, LocReqAgent and LocResAgent classes. In order to access these classes from the OTcl domain, they should be linked to the corresponding *interpreted hierarchy* classes, Application/LocDis, Agent/LocReq and Agent/LocRes, respectively.

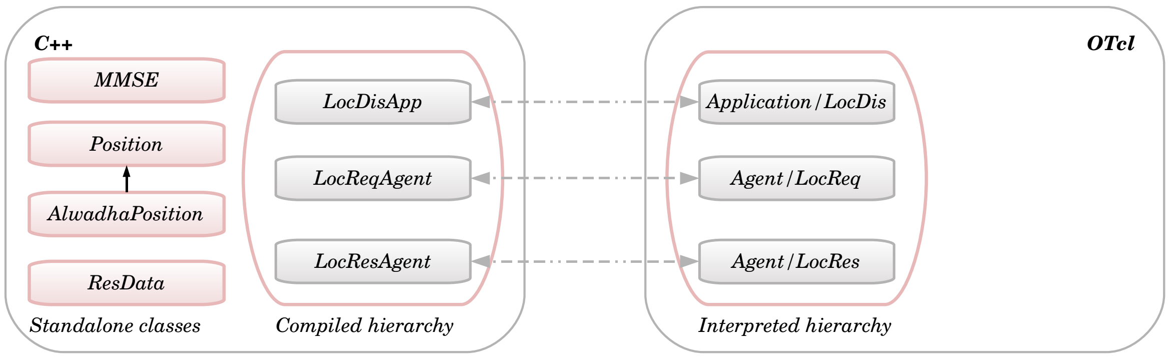


Figure 3.3. The new modules added to the ns-2

3.2.2.1 MMSE class

This class is responsible for all the mathematical matrices operations required to solve Equation (2.2). Instead of using a general matrices multiplication and matrix inverse, optimised methods dedicated mainly to MMSE were implemented. These optimised methods require less computation and shorter execution time.

3.2.2.2 Position class

This class performs the general multilateration method, which was explained in Section 2.2.2, to estimate the node position. This method uses all of the available references, does not distinguish between beacons and references, does not weigh the references used and performs the estimation only once. This class is the base class for ALWadHA and other localisation algorithms that will be explained later.

3.2.2.3 AlwadhaPosition class

This class is derived from Position class. It includes the implementation of an ALWadHA localisation algorithm, which was explained in Section 3.1. Compared with the Position class, AlwadhaPosition has more functionalities such as using a smart reference-selection method, specifying the number of references used, applying a termination criterion and other features as explained earlier.

3.2.2.4 ResData class

This class is responsible for storing and retrieving the location information included in the “location response” packets received from the neighbouring references. This information is the address, the location, the probability of accuracy of the sending references and the power with which the packet is received.

3.2.2.5 LocReqAgent class

LocReqAgent is derived from the Agent class. This agent is responsible for constructing a “location request” packet (PT_LOCREQ) and then broadcasting it to neighbouring nodes. This agent should be attached only to unknown nodes because beacons already know their location.

3.2.2.6 LocResAgent class

This class is a child of class Agent. It is responsible for receiving packets. This agent should be attached to all nodes (unknowns and beacons). Two types of packets could be received. The first is a *location request* packet (PT_LOCREQ). If this type of packet is received by a beacon or reference node it constructs a “location response” packet (PT_LOCRES) that includes location information and then sends it to the requesting node. Unknown nodes receiving this type of packet simply deallocate it. The second is a *location response* packet (PT_LOCRES). The requesting node that receives this packet sends it to the application layer (LocDisApp), which processes the included location information to estimate the node's position.

3.2.2.7 LocDisApp class

This class is derived from the Application class. Each node in the network uses an object from this class by attaching it to its agent(s). The LocDisApp class performs several functions, such as periodically invoking the broadcast method of LocReqAgent to broadcast a "location request" packet, processing the received “location response” packet and estimating the node location.

3.2.2.8 Interpreted hierarchy

In fact, no OTcl modules were created. However, in order to be able to access the newly compiled hierarchy classes LocDisApp, LocReqAgent and LocResAgent from the OTcl domain, these classes were mapped and linked to corresponding OTcl classes, which are Application/LocDis, Agent/LocReq and Agent/LocRes, respectively. In this way the users are able to create an object of the compiled hierarchy classes from the OTcl domain. For example, the OTcl command “set lreq [new Agent/LocRec]” will create a new object of class: LocReqAgent.

3.2.3 Class hierarchy

The Doxygen documentation system [63] was used to illustrate the class hierarchy of the new classes. For the sake of simplicity, only the new classes, the classes they are derived from (i.e. parent classes) and the classes used by these new classes were included. Solid lines show where a class is “inherited”, or derived, from another class; for example $A \rightarrow B$ means class A is derived from class B. Dotted lines show when a class is using a method and/or member of another class.

3.2.3.1 Position and AlwadhaPosition classes

The Position class represents the general multilateration method for estimating the node position, which is called the M_Single algorithm. As shown in Figure 3.4, Position class is the base class of AlwadhaPosition class and the other localisation algorithms, which were implemented for the sake of comparison.

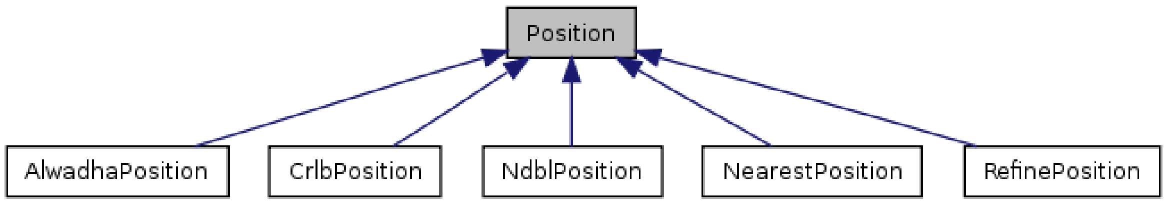


Figure 3.4. Inheritance diagram for Position class

Figure 3.5 shows the collaboration diagram for AlwadhaPosition class. Position class uses the MMSE and LocDisApp classes and ReferenceNode structure, which consists of two members, location variable and double variables, to store the measured distance. AlwadhaPosition class uses an array of this structure (ref_nodes_) to store the location information (location and distance) of neighbouring references. The Location class represents the X, Y and Z coordination of sensor nodes.

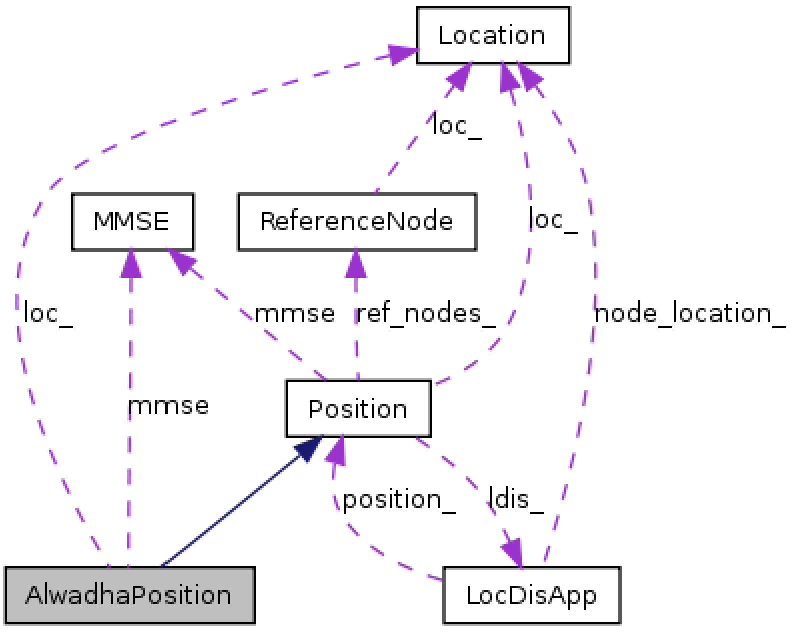


Figure 3.5. Collaboration diagram for Position and AlwadhaPosition classes

3.2.3.2 The Timer classes

Three timer classes were created, which are the ReqTimer, EstimateTimer and OutputTimer classes, as shown in Figure 3.6. These classes are derived from the TimerHandler class. These timer classes collaborate with LocDisApp to schedule several tasks during the run time. The ReqTimer is used to moderate how frequently sensor nodes broadcast a "location request" packet. After sending the "location request" packet, the EstimateTimer is used to schedule the estimation process after a specific delay, which is required to give "location response" packets enough time to receive from neighbouring references. The OutputTimer is used to schedule the action of recording the result, such as location error, number of references used and remaining energy, to the trace file.

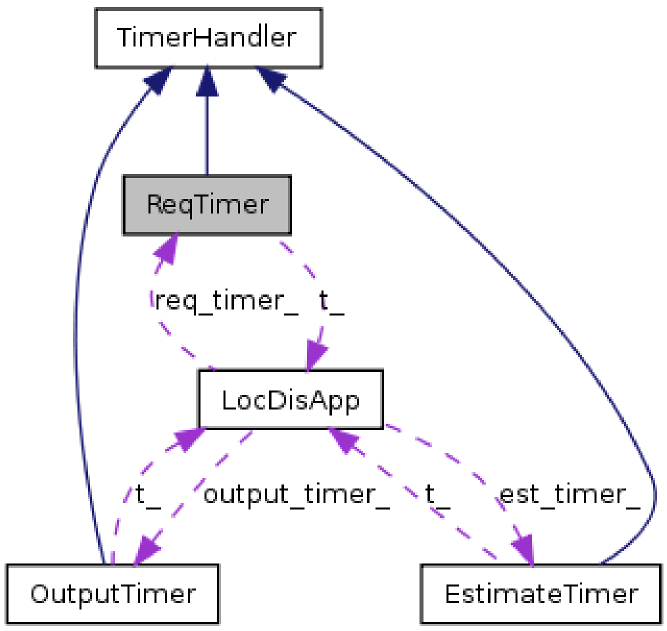


Figure 3.6. Collaboration diagram for the ReqTimer, EstimateTimer and OutputTimer classes

3.2.3.3 LocReqAgent and LocResAgent classes

As shown in Figure 3.7, these classes are derived from the Agent class. LocReqAgent constructs and broadcasts a "location request" packet. LocResAgent is responsible for handling the packets received, which could be "location request" or "location response" packets. Two new types of packet were created: firstly, a "location request" packet (PT_LOCREQ), which uses the new protocol-specific header (PSH) defined in the structure hdr_locreq, and secondly, a "location response" packet (PT_LOCRES), which uses the new PSH defined in the structure hdr_locres. LocReqAgent uses only hdr_locreq to construct "location request" packets. LocResAgent uses both of the headers' structures: it uses hdr_locreq to gain access and to process the received "location request" packets, while it uses the hdr_locres to construct the "location response" packets.

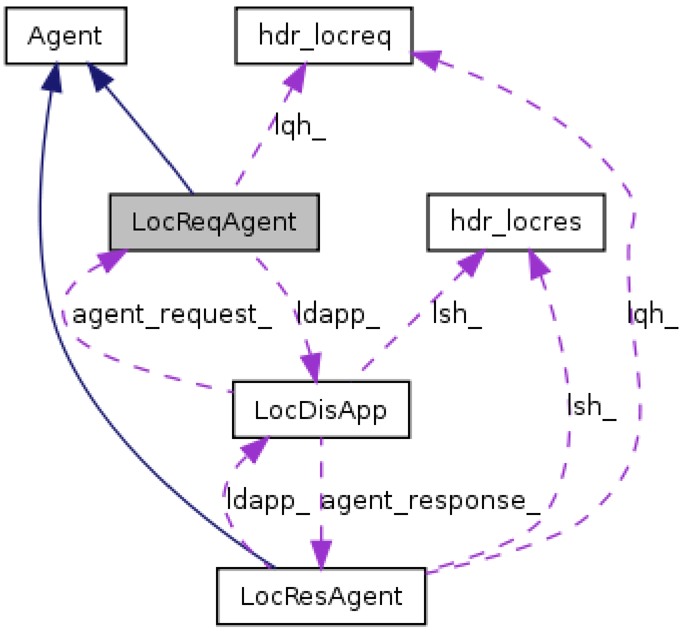


Figure 3.7. Collaboration diagram for LocReqAgent and LocResAgent classes

3.2.3.4 LocDisApp class

The LocDisApp class is derived from the Application class. As shown in Figure 3.8, this class collaborates with several classes, which are the three timers, the two agents, Location and Position classes. It uses the hdr_locres header structure to gain access to the received "location response" packets in order to process the included location information and estimate the node location. LocDisApp uses a vector of class ResData, which is used to store the location information received from neighbouring references. This information is the address, location and the probability of accuracy of the sending node and the power with which the packet is received.

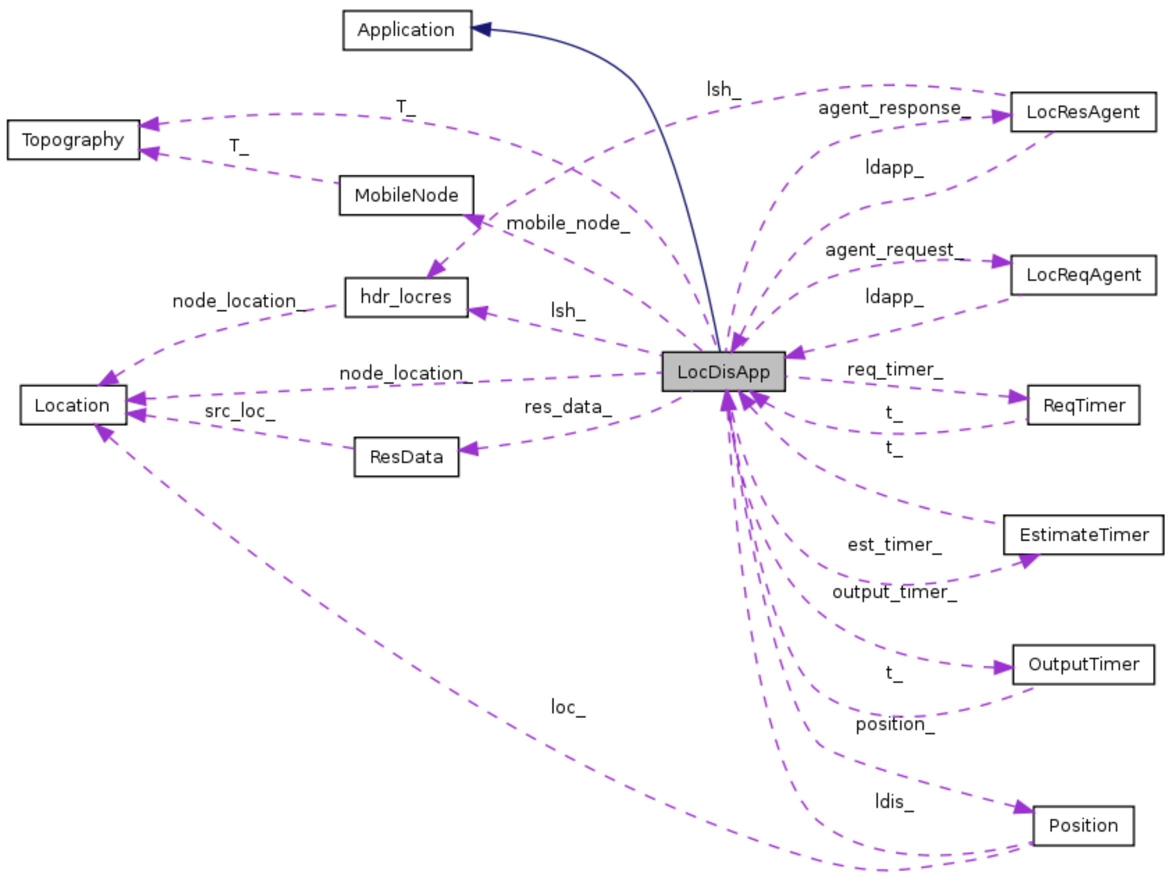


Figure 3.8. Collaboration diagram for LocDisApp class

3.2.3.5 The complete class hierarchy of the new modules

Figure 3.9 shows the complete collaboration diagram for the new classes. The complete procedures of the localisation process are as follows:

- LocDisApp schedules the OutputTimer with a specific delay (for example 1.0 second) to record the result into a trace file.
- LocDisApp schedules the ReqTimer with a specific delay, which determines how frequently the node broadcasts a "location request" packet.
- At the expiration time of ReqTimer, the LocDisApp invokes the LocReqAgent's method called broadcast( ) in order to broadcast a “location request” packet, schedules the EstimateTimer to start location estimation after a specific delay and reschedules the ReqTimer.

- LocReqAgent constructs a “location request” packet (PT_LOCREQ), which includes the required accuracy level, and then it broadcasts the packet to the neighbouring nodes.

- The LocResAgent of the references that received the “location request” packet requests from the LocDisApp the location information of the node, which is the node's location and the node's probability of accuracy. LocResAgent constructs a new “location response” packet (PT_LOCRES), which includes this information, and sends it back to the requesting node.

- The LocResAgent of the requesting node receives the “location response” packets from neighbouring references and then sends them to LocDisApp for more processing.

- LocDisApp extracts the required information from the packet received, namely the address, location and the probability of accuracy of the sending reference node and the power with which the packet is received, and then stores this information in a ResData vector.

- At the expiration time of EstimateTimer the LocDisApp invokes the AlwadhaPosition's method called estimate( ).

- AlwadhaPosition estimates the node location using the data stored in the ResData vector, based on the procedures explained in Section 3.1.

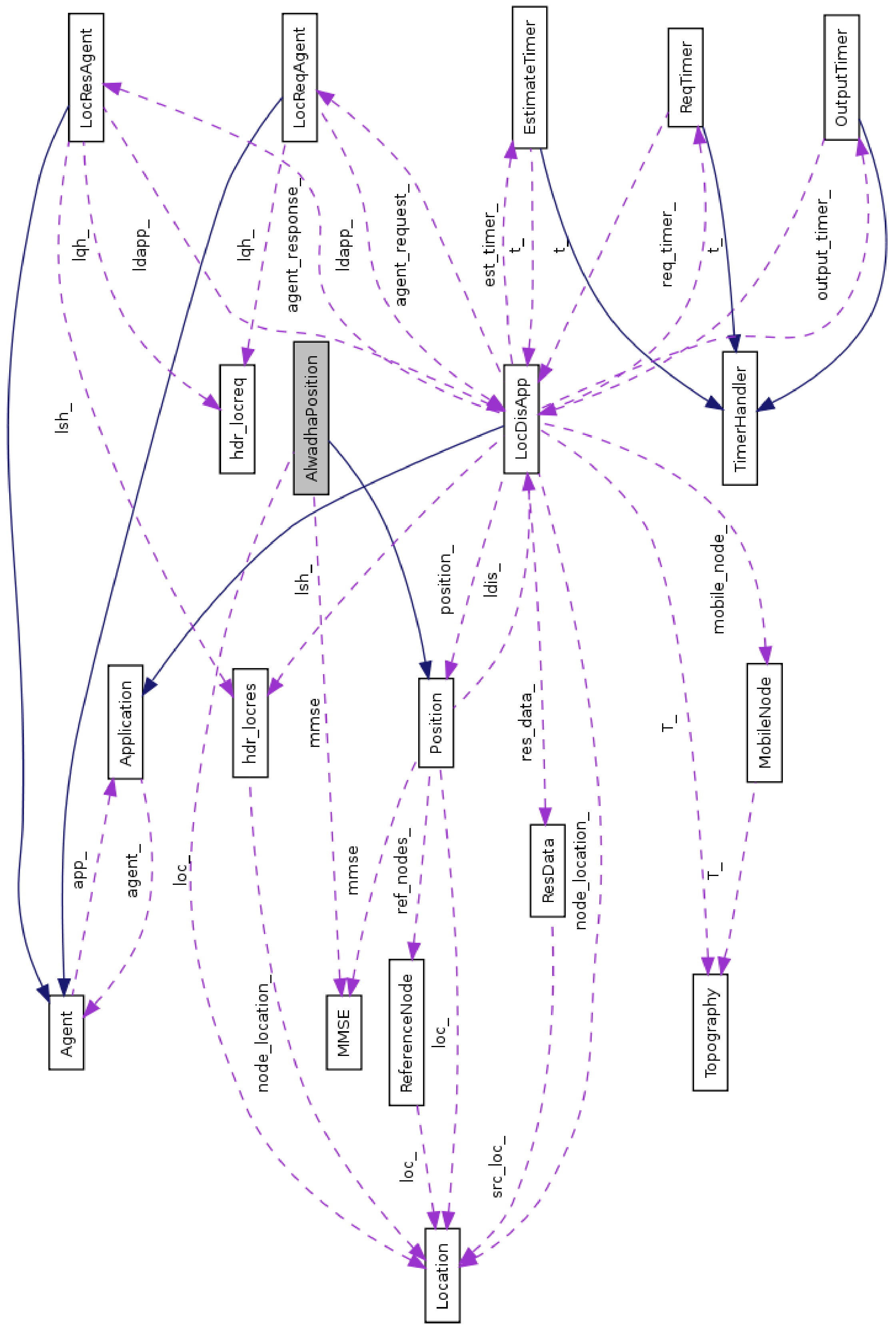


Figure 3.9. Collaboration diagram for the new classes

3.2.4 The structure of the new ns-2

Figure 3.10 Shows the structure of the new ns-2, where the files under the "location" directory represent the new files that were added to ns-2, while the other files (left-hand side) are the modified files.

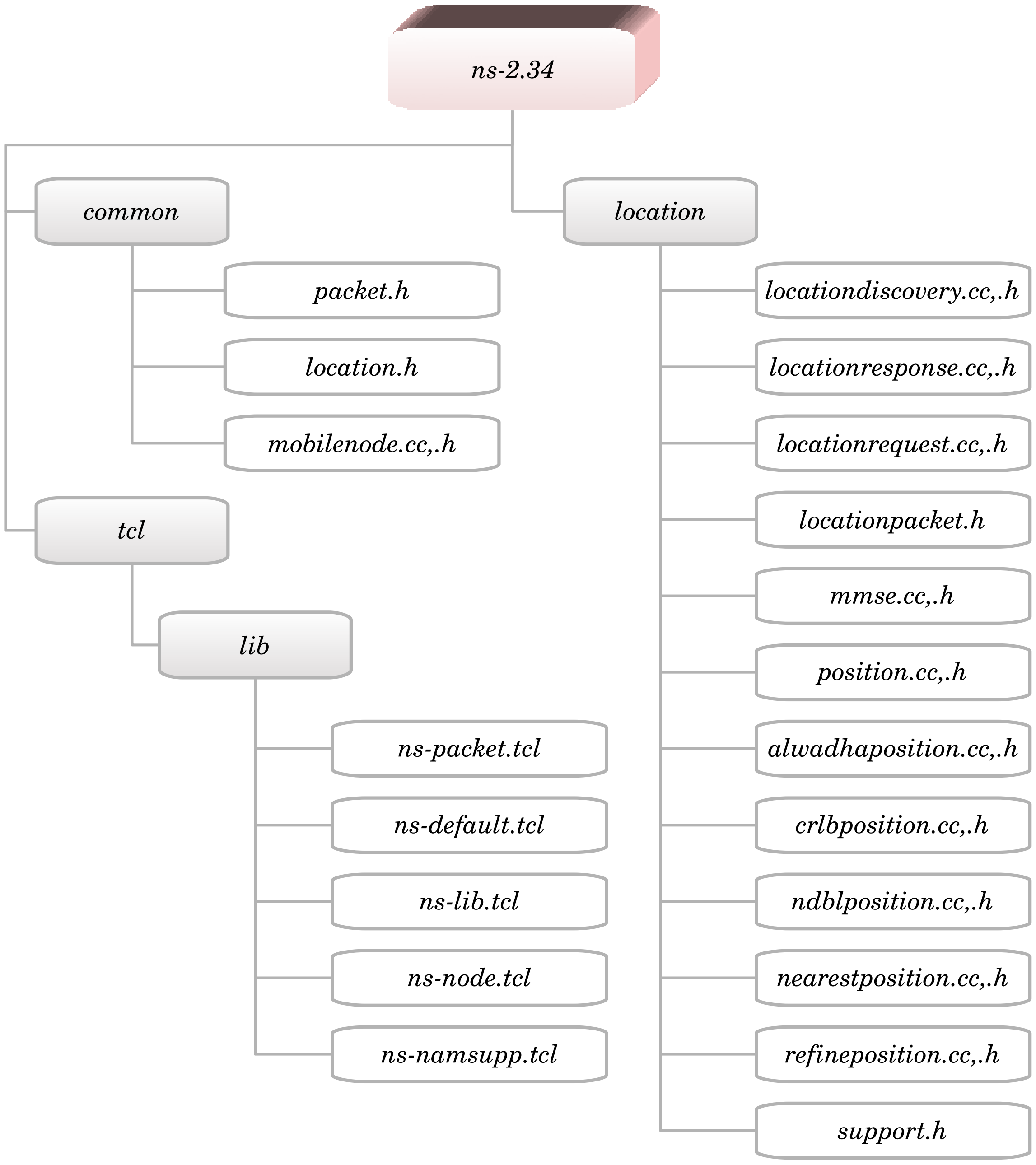


Figure 3.10. The structure of the new ns-2, showing the new files added to ns-2 (right-hand side) and the modified files (left-hand side)

The new classes that were discussed in the previous two sections are implemented in the files under the "location" directory. In addition to these new files, some other files were modified as follows:

- *common/packet.h:* Two packet types were created in the locationpacket.h file using two structures (hdr_locreq and hdr_locres). In order to use these two types of packet, their corresponding packet types (PT_LOCREQ and PT_LOCRES) were defined in the packet.h file.

- *common/location.h:* This file contains the Location class, which is used to represent the location coordination (X, Y and Z) of nodes. Some methods were added to this class, such as the getter and the setter of an individual coordinate, is_equal( ) method to check if two locations are the same and distance( ) method to find the distance between two locations.

- *common/mobilenode.cc, .h:* Two methods were added to these files. The first is to get an object to the topography. The second is to record the result in the trace file. The result could be the location error, number of references used, probability of accuracy and remaining energy.

- *tcl/lib/ns-packet.tcl:* In order to activate the new header classes, the OTcl class names were included in this file. The new OTcl header classes are "PacketHeader/LocReq" and "PacketHeader/LocRes", and so only "LocReq" and "LocRes" were added to the active protocol list.

- *tcl/lib/ns-node.tcl:* As mentioned before, from the localisation perspective, the node could be a beacon, reference or unknown. In order to specify the type of nodes, a new instvar, called "nodeAttribute_", and a new instproc to get the node attribute, called "attribute", were created.

- *tcl/lib/ns-lib.tcl:* In order to enable the simulator to deal with the node attribute, an instproc was created to set the instvar "attribute_". Within the "Simulator instproc create-wireless-nodes" the node is allowed to set its attribute ($node set nodeAttribute_ $attribute_).

- *tcl/lib/ns-namsupp.tcl:* During the simulation, when the unknown nodes estimate their position they change their colour (for instance to red). In order to enable the nodes to change their colour after running the simulator, the "Node instproc color" was modified within this file.

- *tcl/lib/ns-default.tcl:* Sometimes it is necessary to bind some variables in both hierarchies (i.e. interpreted and compiled hierarchies). The default value of these bind variables is initialised in the ns-default.tcl file. Several variables were bound, such as the packet size, the request frequency (reqFreq_), the showColor_ variable to enable showing the colour of the nodes based on their attribute and the subset_ variable to specify which localisation algorithm should be applied (e.g. ALWadHA, CRLB or Nearest).

3.2.5 Guidelines for running the simulation

Using the extended ns-2 does not require new knowledge or writing a specific code to run the simulator. Normal users who have the basic knowledge to run a simple wireless network using ns-2 are able to write a simple OTcl script to simulate the proposed localisation system. This section gives some guidelines for configuring the localisation simulation. It assumes that the reader is familiar with setting up wireless mobile network simulations in ns-2. Therefore, it will not explain the entire simulation procedures; rather, it will show how to configure nodes to simulate localisation. However, the reader is referred to [60] for some tutorials about configuring wireless networks.

At the beginning of the simulation there are only two types of nodes: beacons and unknowns. Each of them has a different configuration, as shown in Figure 3.11. Beacons already know their location, so they should be attached only with LocResAgent. Unknowns should be attached with LocReqAgent in order to allow them to broadcast "location request" messages. After estimating their position, unknowns could become a reference for other nodes, and so they are also attached with LocResAgent. These two types of agents should be attached to LocDisApp.

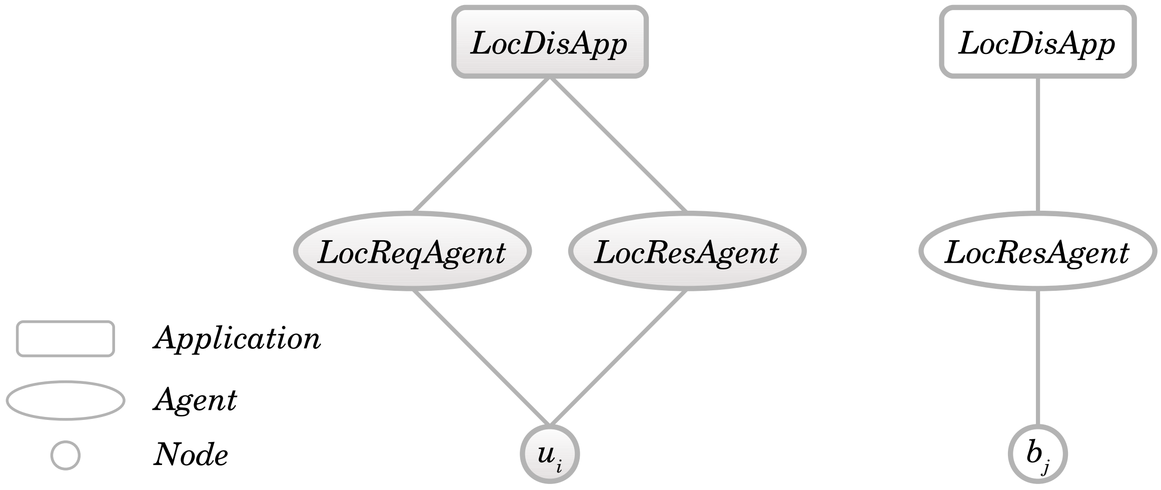


Figure 3.11. Node configuration, where u_i is an unknown node and b_j is a beacon node

Before creating the nodes it is necessary to specify the nodes' configuration, such as the routing protocol, MAC type and so on. In addition to these configurations it is also necessary to specify the attribute of the nodes, which can be done with the help of the command “node-config”

```
set val(nn)      10               ;# number of mobilenodes
set val(nu)      7                ;# number of unknown nodes
set val(nb)      3                ;# number of beacon nodes

# Beacon nodes:
# Nodes configuration
set val(attr)    BEACON           ;# node attribute
$ns_ node-config -attribute $val(attr)
# Nodes creation
for {set i 0} {$i < $val(nb)} {incr i} {
      set node_($i) [$ns_ node]
}

# Unknown nodes
# Nodes configuration
set val(attr)         UNKNOWN
$ns_ node-config -attribute $val(attr)
# Nodes creation
for {set i $val(nb)} {$i < $val(nn) } {incr i} {
      set node_($i) [$ns_ node]
}
```

The next step is to create the required agents and application based on the configuration shown in Figure 3.11. Beacon nodes require only LocResAgent and LocDisApp; this can be done as shown below:

```
# Beacon nodes have only response agent
for {set i 0} {$i < $val(nb)} {incr i} {
     # Setup the response agent
     set lres_($i) [new Agent/LocRes]
     $ns_ attach-agent $node_($i) $lres_($i)

     # Setup the location-discovery application
     set ldis_($i) [new Application/LocDis]
     $ldis_($i) attach-agent $lres_($i)
}
```

Unknown nodes also require LocReqAgent, as shown below:

```
# Unknown nodes have both request and response agent
for {set i $val(nb)} {$i < $val(nn) } {incr i} {
     # Setup the request agent
     set lreq_($i) [new Agent/LocReq]
     $ns_ attach-agent $node_($i) $lreq_($i)

     # Setup the response agent
     set lres_($i) [new Agent/LocRes]
     $ns_ attach-agent $node_($i) $lres_($i)

     # Setup the location-discovery application
     set ldis_($i) [new Application/LocDis]
     $ldis_($i) attach-agent $lreq_($i)
     $ldis_($i) attach-agent $lres_($i)
}
```

Finally, the location-discovery applications should be started at a specific time:

```
# Start the locdis applications
for {set i 0} {$i < $val(nn)} {incr i} {
     $ldis_($i) set random_ 1
     $ldis_($i) set subset_ 3
     $ldis_($i) set showColor_ 1
     $ns_ at 0.0 "$ldis_($i) start"
}
```

If the user does not want to use the default value of the bind variables he can change the setting of this variable before starting the applications. For instance, in the previous code the random_ variable was set to 1 to all the LocDis applications to start broadcasting the "location request" messages at a random time instead of starting immediately (random_ = 0). The subset_ variable specifies the localisation algorithm that should be applied to estimate the node location; the value of three refers to the ALWadHA algorithm. The variable showColor_ is used in all the unknowns to change their colour after they estimate their location; setting this variable to zero disables this feature.

3.2.6 Manipulate output files

After simulation, ns-2 outputs either text-based or animation-based simulation results. As shown in Figure 3.12, several tools could be used to interpret these results graphically or interactively. The Network AniMator (NAM) [60] is an animation tool that uses the animation-based results to view the network simulation traces and real-world packet traces. NAM supports topology layout, packet level animation and various data inspection tools.

Text-based results consist of a lot of details on events that occur at the network, such as sending or receiving packets, nodes' movement and nodes' remaining energy. A new method was written that records the localisation-related information to the trace file. This information includes the localisation error, number of references used, remaining energy and number of iterations. To analyse particular data such as localisation error, the relevant information needs to be extracted from the traces and transformed to a more easily conceived presentation. Perl language [64] was used for this purpose. Perl stands for “Practical Extraction and Report Language”. Perl can be used to filter and to process the ASCII trace files in Unix. For example, a simple Perl script could be written to estimate the average localisation error of all nodes at each second.

The Gnuplot software was used to represent the relevant results graphically. Gnuplot [65] is a portable command-line-driven graphing utility for Linux, as well as many other platforms.

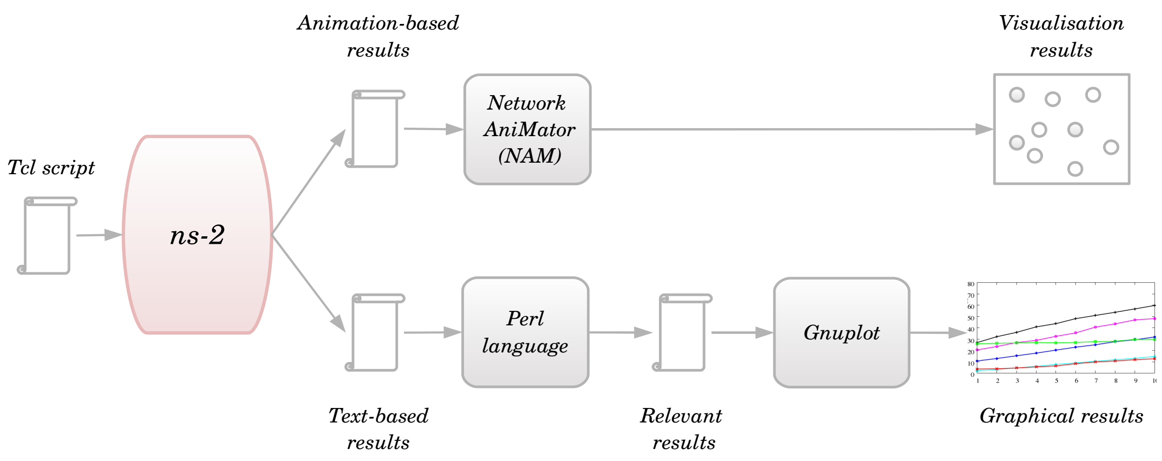


Figure 3.12. Tools used to manipulate the result

3.3 SIMULATION

This section evaluates the ALWadHA algorithm, looking at the effects of nodes' deployment, nodes' density and network size on the estimation error and the number of references used. It will examine the impact of increasing the distance-measurement error on the accuracy of the ALWadHA algorithm. It will compare the performance of the ALWadHA algorithm with other popular schemes.

3.3.1 Localisation algorithms

Five localisation algorithms, in addition to the ALWadHA algorithm, have been implemented for the performance comparison, using the same assumptions. These algorithms will be explained briefly and then a comparison of them will be presented in Table 3.2, where each algorithm is classified based on the number of estimations and the set of reference categories used. The table also shows how each algorithm selects the subset of references (S_i) and how it specifies the number of references ($C(S_i)$), which is either predefined manually or specified dynamically, based on certain conditions at run time.

3.3.1.1 General multilateration

The node uses all the available references to estimate its position $(S_i = R_i)$ using the basic multilateration method. Two algorithms based on this method were implemented.

The first one is based on the *single-estimation* approach (M_Single), while the second one is implemented based on the *successive-refinement* approach (M_Refine).

3.3.1.2 Nearest three references

The node selects a subset of three references to estimate its position [37]. The selected references have the minimum measured distance to the node. The selected subset is:

$$S_i = S_i^k, \text{ where } S_i^k \in \varsigma \text{ and } S_i^k \text{ has } max \left(\sum_{j \in S_i^k} 1 - \frac{\hat{d}_{i,j}}{r_{tx}} \right) \quad . \tag{3.11}$$

3.3.1.3 CRLB-based algorithm

The proposed algorithm in [13] tries to minimise the localisation error by selecting a subset of references based on the measured distance and CRLB on localisation error. The selected subset is

$$S_i = S_i^k, \text{ where } S_i^k \in \varsigma \text{ and } S_i^k \text{ has } max \left(\sum_{j \in S_i^k} P_{crlb}^j \right) \quad . \tag{3.12}$$

3.3.1.4 Node distribution-based localisation algorithm

The authors of [43] modify the refinement dwMDS algorithm [42] by simplifying the computation and reducing the processing time. The proposed algorithm is called the node distribution-based localisation (NDBL) algorithm. In the NDBL algorithm each node selects the references that are within a measurable range

$$S_i = \{ r_j, \text{ where } \hat{d}_{i,j} \leqslant r_{tx} \} \quad . \tag{3.13}$$

In fact, this selection criterion does not exclude many references and it could be equivalent to the all-references set $S_i = R_i$. The algorithm starts by estimating the multi-hop distances to the beacons and then using the trilateration method to estimate a rough initial position. This inevitably implies error, due to compounding of error from the approximated measurement at each hop. In the next refinement phases, all the computation is based on the measured distances to the one-hop references. For the sake of comparison and enhancing the accuracy of the initial position, the multilateration method is used based on the measured distances to the one-hop references to estimate the initial position, without any modification in the refinement phases.

Table 3.2. Implemented localisation algorithms

Localisation	**Estimations**		**References**		**S$_i$**	**C(S$_i$)**
algorithms	**Single**	**Refinement**	**All**	**Subset**		
M_Single	√		√		R$_i$	-
M_Refine		√	√		R$_i$	-
Nearest	√			√	$max\ (\sum_{j\in S_i^k} 1-\frac{\hat{d}_{i,j}}{r_{tx}})$	Manual
CRLB	√			√	$max\,(\sum_{j\in S_i^k} P_{crlb}^j)$	Manual
NDBL		√		√	$\{r_j, \text{where } d_{i,j} \leqslant r_{tx}\}$	Dynamic
ALWadHA		√		√	S_i^k with $min(c)$ and $(1-\frac{1}{\sum_{j\in S_i^k} P_{acc}^j}) \geqslant P_{min}$	Dynamic

3.3.2 Performance comparison

The performance of each algorithm was then evaluated based on two metrics: The first was the localisation error, which reflects on the location accuracy. The second was the number of references used in the position estimation. (This has a significant impact on the computation cost, since the complexity of the localisation algorithms is proportional to the number of references used.)

3.3.2.1 Localisation error

The mean error is estimated every second for all knowns as a ratio of transmission range. The mean error at a specific time *t* is equal to the summation of the location error of all knowns, divided by the number of these knowns, and then it is divided by the transmission range as follows:

$$Mean\ error_t = \left(\frac{1}{n} \sum_{i=1}^{n} \| \hat{z}_i - z_i \| \right) \frac{1}{r_{tx}}\ 100\% \tag{3.14}$$

where *n* is the total number of knowns at a specific time *t* $(n = C(K))$.

3.3.2.2 Number of references (*C*(*S*$_i$))

Increasing the number of references could enhance the accuracy of the position estimation. However, it increases the complexity of computations [13]. The multilateration method uses all the available references. The number of references for the Nearest and CRLB algorithms was predefined manually by three references. This number was selected to show the impact of using the minimum required number of references on the performance of these two algorithms, and also to make them comparable with the ALWadHA algorithm, which uses almost the same number of references $(C(S_i) \approx 3)$. The ALWadHA and NDBL algorithms specify the number of references dynamically at each iteration, based on a specific criterion, as shown in Table 3.2. The average number of references used at a specific time *t* is calculated as follows:

$$\#\,of\ References_t = \frac{1}{n} \sum_{i=1}^{n} C(S_i) \tag{3.15}$$

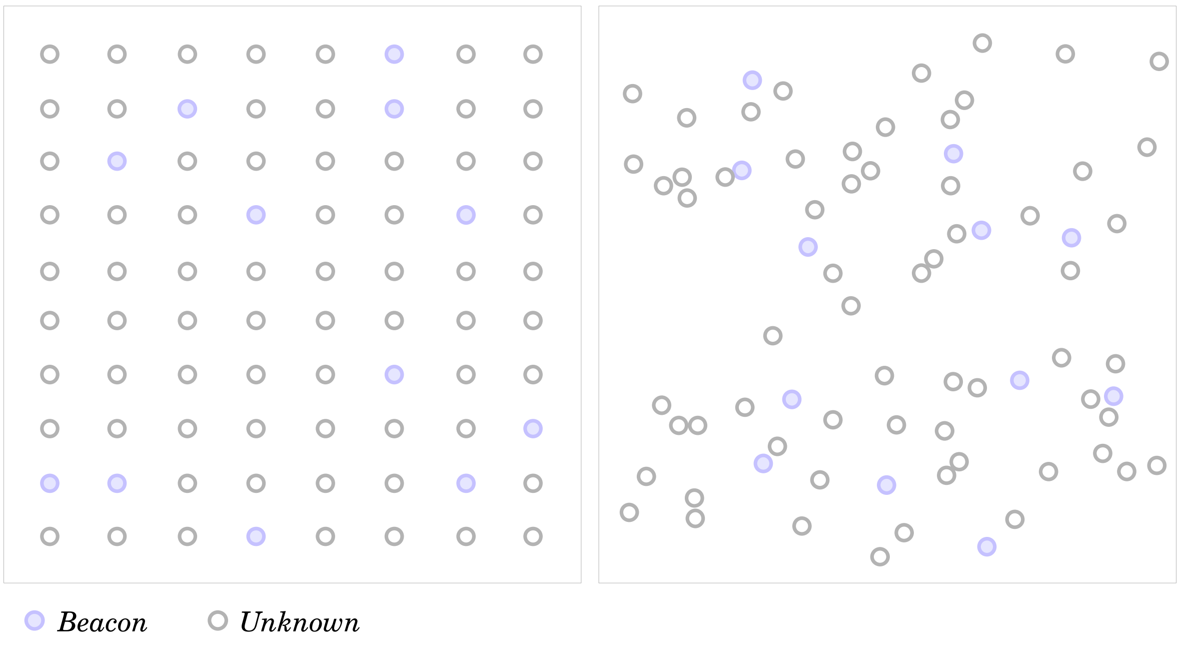
where *n* is the total number of knowns at a specific time *t*.

3.3.3 Setup and environment

The localisation algorithms were evaluated using the network simulator (ns-2). In order to satisfy the applicability of the proposed algorithm in large-scale WSNs, the field was divided into several sub-areas, each area containing the same number of unknowns and beacons without ignoring the randomness of node distribution. All nodes had a limited transmission range (r_{tx}) of 50 m. At each experiment the simulation was run 100 times; the duration of each run was 600 sec (the total duration was 60 000 sec), and at each run nodes were redistributed randomly in different places (using a different seed value). RSS was used to measure the distance between nodes. However, to simulate noise, each measured distance was disturbed by a normal random variant with the following settings: a mean of 0.1% of the measured distance and a standard deviation of 1% of the measured distance. To check the impact of increasing the distance-measurement error on the localisation algorithms, the standard deviation was varied from 1% to 10% of the measured distance.

Two types of deployment were used: grid and random deployment, similar to the networks shown in Figure 3.13. The grid deployment had the following characteristics:

- Deployment area: $200\,m \times 200\,m$ divided into four sub-areas
- Number of beacons: 12; each sub-area had three beacons (0.75 beacon per r_{tx}^2)
- Number of unknowns: 68; each sub-area had 17 unknowns (4.25 unknowns per r_{tx}^2)
- Beacon distribution: each sub-area was divided into 20 rectangles, then three rectangles were selected randomly where the three beacons were placed in their centres.
- Unknowns distribution: the unknowns were placed in the centre of the remaining empty rectangles.



a. Grid deployment b. Random deployment

Figure 3.13. Examples of node distribution

The random deployment was based on the following characteristics:

- Deployment area: $200\,m \times 200\,m$ divided into four sub-areas
- Number of beacons: 12; each sub-area had three beacons (0.75 beacon per r_{tx}^2)
- Number of unknowns: 64; each sub-area had 16 unknowns (four unknowns per r_{tx}^2)
- Beacon distribution: three beacons distributed randomly in each sub-area
- Unknowns distribution: each sub-area was divided into four regions $(50\text{m} \times 50\text{m})$, and then four unknowns were distributed randomly in each region.

Table 3.3. Experiments setup

Factors	Fig.	Deployment	Area	# B	# U	Notes	
Node deployment	3.14	Random	200 x 200	12	64	No error	
	3.15	Random	200 x 200	12	64	Sd = 1 %	
	3.16	Grid	200 x 200	12	68		
Node density	3.17	Random	200 x 200	12	64	4 *U* per r_{tx}^2	0.75 *B* per r_{tx}^{Υ}
	3.18	Random	200 x 200	12	96	6 *U* per r_{tx}^{Υ}	
	3.19	Random	200 x 200	12	160	10 *U* per r_{tx}^{Υ}	
	3.20	Random	200 x 200	12	64	0.75 *B* per r_{tx}^2	4 *U* per r_{tx}^2
	3.21	Random	200 x 200	16	64	1 *B* per r_{tx}^{Υ}	
	3.22	Random	200 x 200	20	64	1.25 *B* per r_{tx}^{Υ}	
Network size	3.23	Random	100 x 100	3	16	Small	0.75 *B* per r_{tx}^2
	3.24	Random	200 x 200	12	64	Normal	
	3.25	Random	600 x 600	108	576	large	4 *U* per r_{tx}^2
Standard deviation	3.26.a	Grid	200 x 200	12	68	Sd = 1 % to 10 %	
	3.26.b	Random	200 x 200	12	64		
	3.27	Grid	200 x 200	12	68	Sd = 10 %	

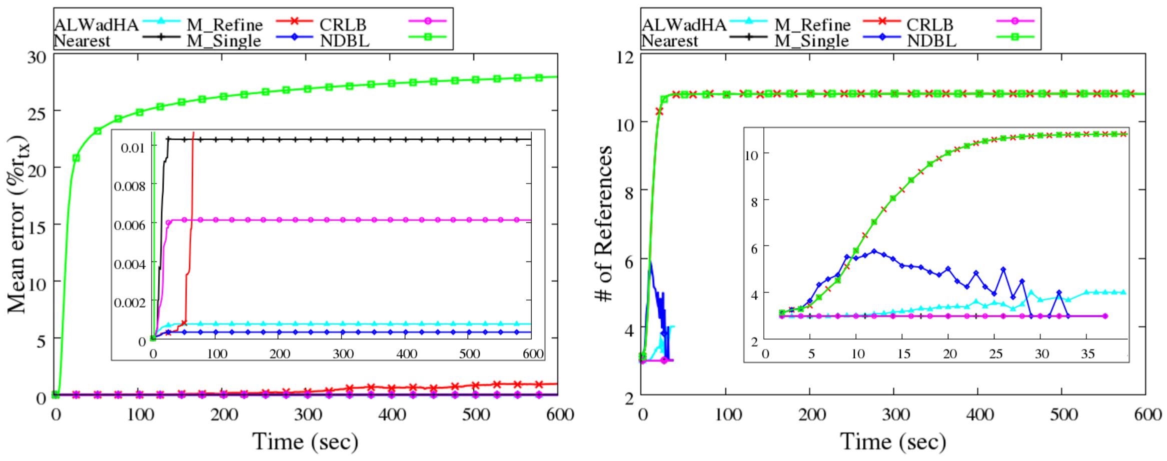
However, some of the random deployment characteristics could be different for some experiments, such as the number of beacons or unknowns or the network size. The exact setting for those experiments that have different characteristics will be provided. The reader can refer to Table 3.3 for the complete setup environment of each experiment that was performed in preparing this chapter.

3.3.4 Results and comparisons

Several experiments were performed to evaluate the localisation algorithms, based on three factors: node deployment, node density and network size. In each experiment, these localisation algorithms were evaluated and compared based on the two metrics mentioned above. Finally, the impact of increasing the distance-measurement error on the accuracy of these algorithms was examined.

3.3.4.1 Node deployment

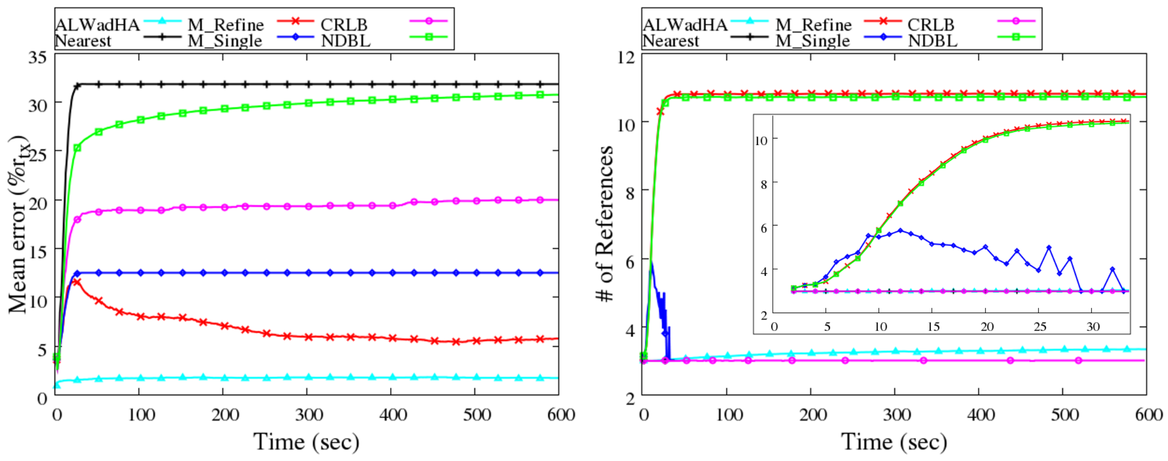
In the first two experiments, the nodes were deployed randomly, based on the characteristics mentioned earlier. In the first experiment, the nodes were deployed randomly without adding any error to the distance measurement, in order to check the error that could be introduced by the localisation algorithm itself. Figure 3.14.a shows the mean error of each algorithm versus the time, while Figure 3.14.b shows the average number of references used by each algorithm. Figure 3.14.a shows that the mean error of the M_Single, Nearest, CRLB and ALWadHA algorithms was close to zero, M_Single performed well in this experiment, but it performed much less well in the rest of the experiments. The mean error of the M_Refine algorithm increased gradually because of the computation error that accumulated during the refinement phases. This type of error does not affect the M_Single, Nearest and CRLB algorithms because they do not perform any refinement and the node stops directly after it gets its first position estimate. This type of cumulative computation error does not affect the ALWadHA algorithm because of the termination criterion used by the algorithm, where the nodes get their final positions within the first 39 seconds, as shown in Figure 3.14.b. The NDBL algorithm has a higher mean error compared with other algorithms. This error is caused by the algorithm itself and the cumulative error during the refinement phases. This result shows the importance of simplifying the localisation algorithm and using a proper termination criterion.



a. Mean error as a ratio of transmission range b. Average # of references

Figure 3.14. Random deployment without distance error

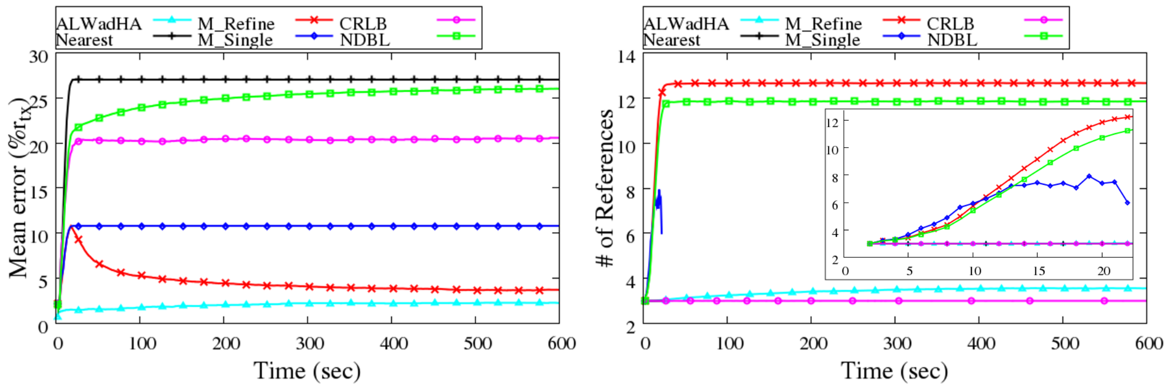
The previous experiment was repeated with distance error. Figure 3.15 shows that the ALWadHA algorithm outperformed other algorithms in terms of position accuracy. The multilateration algorithm (M_Refine) enhanced its estimation over time, because at the beginning a low number of references was available, then over time the nodes started to know their location, which increased the number of references. Using all these references enhanced the accuracy of the multilateration method. After about 25 seconds the M_Refine algorithm started to use more than 10 references to estimate the node position (Figure 3.15.b), while the ALWadHA algorithm kept using a low number of references with an average equal to 3.24, which is very close to the minimum possible number. It could be remarked that using more references could significantly increase the complexity of the algorithm. Figure 3.15.b also shows that the NDBL algorithm uses almost the same number of references as the M_Refine algorithm, which means that its reference-selection criterion does not always exclude references in this scenario.



a. Mean error as a ratio of transmission range b. Average # of references

Figure 3.15. Random deployment with distance error

The third experiment was performed with the existence of distance error based on a grid deployment. From Figure 3.16 it can be noticed that the ALWadHA algorithm has less mean error compared with other algorithms. If this result is compared with experiment two (Figure 3.15), it is noticeable that in the random deployment all algorithms apart from the CRLB and ALWadHA algorithms have a higher mean error. The reason is that in grid deployment most of the nodes have the same number of neighbours, while in random deployment nodes could have a different number of neighbours; this leads to lower accuracy caused by nodes having a lower number of references. This supplies the motivation to rely on proper selection of references, rather than on using a high number of references to enhance the performance of the localisation algorithm. Figure 3.16.b shows that the NDBL algorithm used a lower number of references than the M_Refine algorithm, because in this grid deployment several references are close to the boundary of the transmission range, and so the addition of distance error takes them out of the transmission range and allows the node to exclude them from its subset of references.



a. Mean error as a ratio of transmission range b. Average # of references

Figure 3.16. Grid deployment with distance error

The results of these three experiments are summarised in Table 3.4, where the mean error is recorded at the end of the run time, while the number of references represents the average number of references that were used during the run time.

Table 3.4. Performance comparison of different deployments

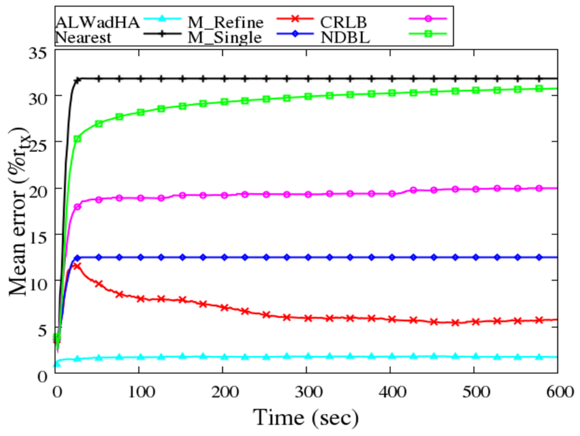
Deployment	**Random (no error)**		**Random (error)**		**Grid (error)**	
Algorithms	**Mean error**	**Average # Ref**	**Mean error**	**Average # Ref**	**Mean error**	**Average # Ref**
ALWadHA	0.0008	3.39	1.75	3.24	2.28	3.42
M_Single	0.0004	4.38	12.53	4.38	10.81	5.89
M_Refine	0.9699	10.67	5.77	10.67	3.73	12.46
CRLB	0.0061	3	19.99	3	20.57	3
NDBL	27.9454	10.67	30.77	10.58	26.01	11.66
Nearest	0.0103	3	31.86	3	27.02	3

3.3.4.2 Node density

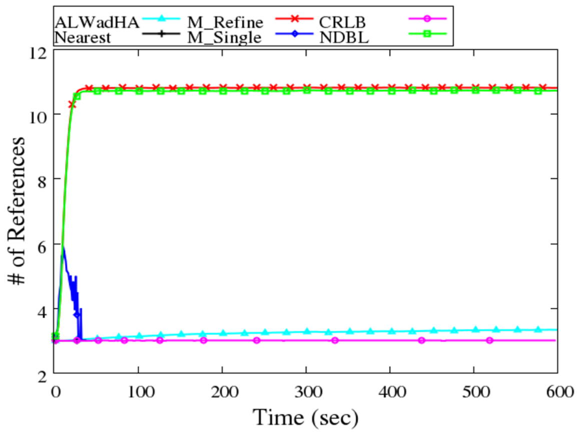
Initially, the network consisted of two types of nodes, beacons and unknowns. This section examines the impact of changing the number of these nodes on the performance of the localisation algorithms. First, a fixed number of beacons were used while changing the number of unknowns, then the number of unknowns were fixed while changing the number of beacons. In all the experiments done in this section, nodes were distributed randomly in a $200\,m \times 200\,m$ field, based on the same characteristics mentioned above for random deployment except for the number of nodes. To examine the unknowns' density, 12 beacons were used (0.75 beacon per r_{tx}^2) and the unknowns' density was varied as follows:

- 64 unknowns (four unknowns per r_{tx}^2)
- 96 unknowns (six unknowns per r_{tx}^2)
- 160 unknowns (10 unknowns per r_{tx}^2).

By comparing Figure 3.17, 3.18 and 3.19 it can be seen that the M_Refine and NDBL algorithms enhance the location estimation accuracy slightly by increasing the nodes' density. On the other hand, these two algorithms use around 25 references in the high unknown density (Figure 3.19.b). Using such a high number of references would dramatically increase the overheads for computation and communication. Figure 3.19.b shows that the NDBL algorithm uses almost the same number of references as the M_Refine algorithm. This result confirms the disadvantage of using the elimination criterion that was mentioned earlier, and shows the impact of using a real selection criterion that is able to select a low number of the references that contribute most to high location estimation accuracy. When using different unknowns' density, the smart reference-selection method used by the ALWadHA algorithm enables the nodes to estimate their location with the best accuracy, compared with other algorithms. The ALWadHA algorithm uses on average fewer than 3.37 references.

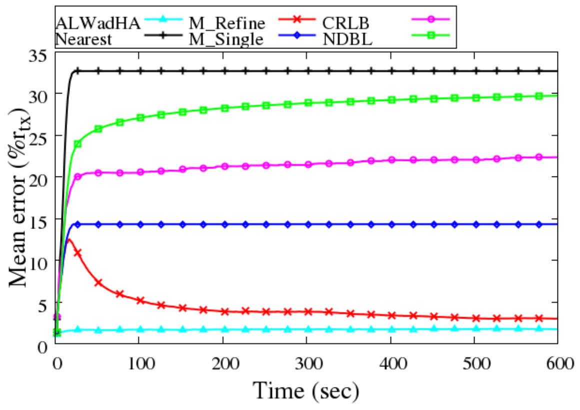


a. Mean error as a ratio of transmission range

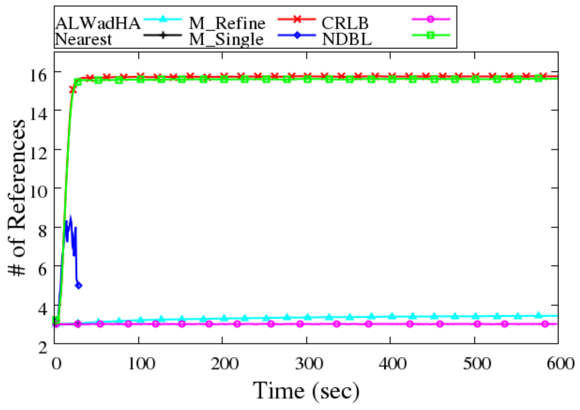


b. Average # of references

Figure 3.17. Unknowns' density: 4 unknowns per r_{tx}^2

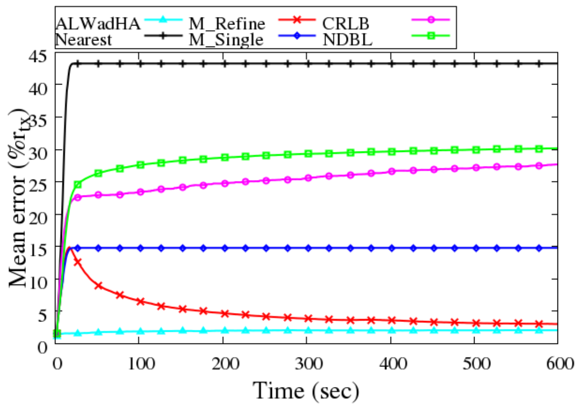


a. Mean error as a ratio of transmission range

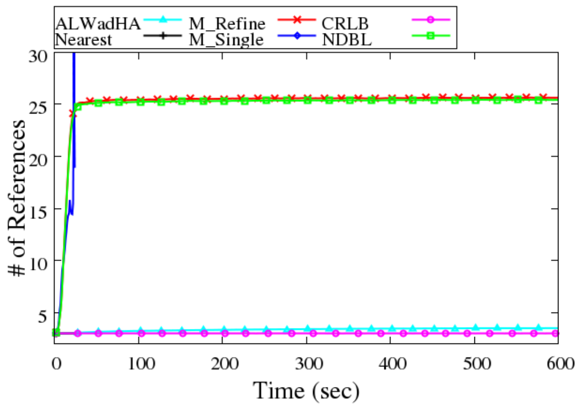


b. Average # of references

Figure 3.18. Unknowns' density: 6 unknowns per r_{tx}^2



a. Mean error as a ratio of transmission range



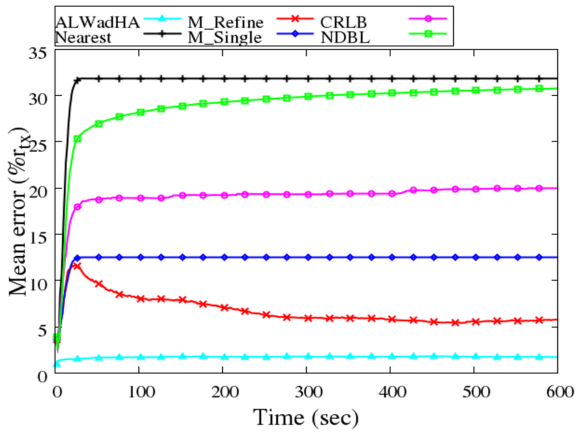
b. Average # of references

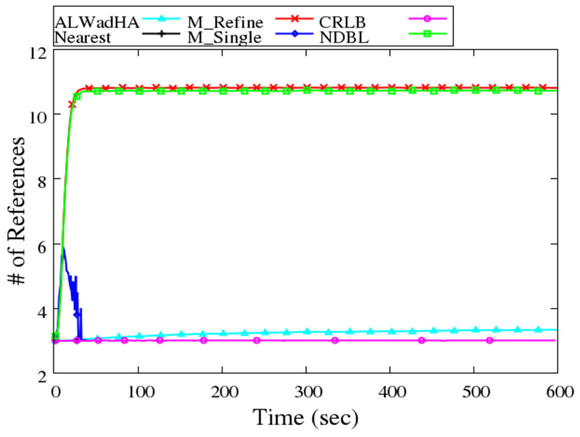
Figure 3.19. Unknowns' density: 10 unknowns per r_{tx}^2

In the second part of this section, 64 unknowns were used (four unknowns per r_{tx}^2) and the beacons' density was varied as follows:

- 12 beacons (0.75 beacon per r_{tx}^2)
- 16 beacons (1.0 beacon per r_{tx}^2)
- 20 beacons (1.25 beacon per r_{tx}^2).

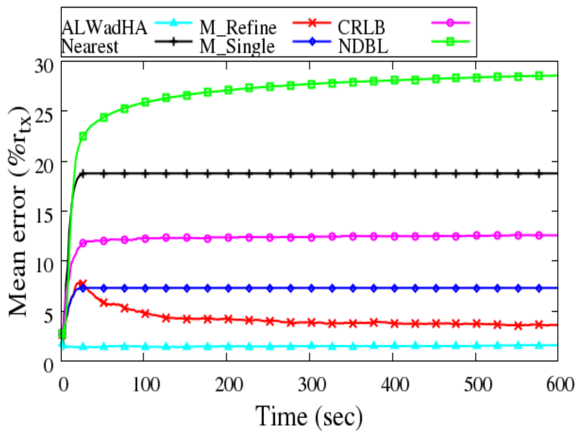
As expected, Figure 3.20, 3.21 and 3.22 show that increasing the beacons' density enhances the accuracy of all the localisation algorithms. However, the NDBL algorithm enhances its accuracy only slightly, compared with the other algorithms. When using a different beacon density, the ALWadHA algorithm achieves the best accuracy of all the algorithms. An interesting observation about ALWadHA is that the average number of references used is reduced by increasing the beacon density. This is because from the smart reference-selection method perspective, more beacons mean more references with a high probability of accuracy, which enables the nodes to select a smaller subset of references fulfilling the condition $P_{acc} \geqslant P_{min}$.

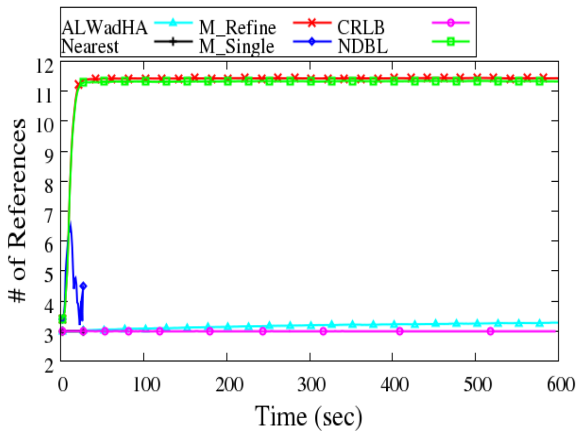




a. Mean error as a ratio of transmission range b. Average # of references

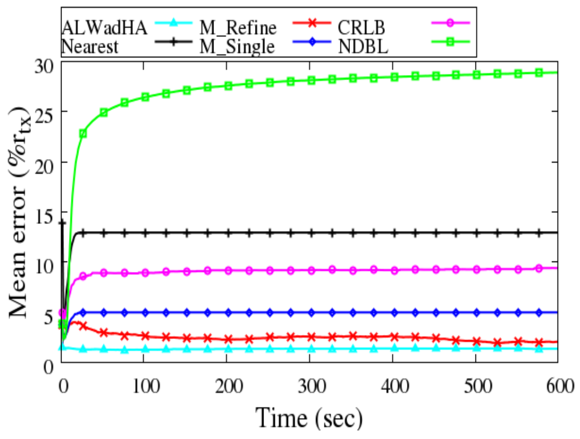
Figure 3.20. Beacons' density: 0.75 beacons per r_{tx}^2

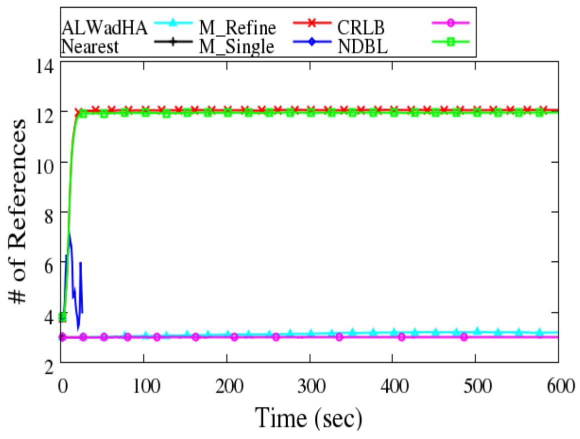




a. Mean error as a ratio of transmission range b. Average # of references

Figure 3.21. Beacons' density: 1 beacon per r_{tx}^2





a. Mean error as a ratio of transmission range b. Average # of references

Figure 3.22. Beacons' density: 1.25 beacons per r_{tx}^2

A performance comparison of the localisation algorithms using different unknowns and beacon density is shown in Table 3.5, where *B* and *U* refer to beacons and unknowns respectively.

Table 3.5. Performance comparison of different node densities

Density **# B**	**12**		**12**		**12**		**16**		**20**	
Density **# U**	**64**		**96**		**160**		**64**		**64**	
Algorithms	Mean error	Avg. # Ref	Mean error	Avg. # Ref	Mean error	Avg. # Ref	Mean error	Avg. # Ref	Mean error	Avg. # Ref
ALWadHA	1.75	3.24	1.74	3.31	2.09	3.37	1.58	3.17	1.37	3.14
M_Single	12.53	4.38	14.33	6.43	14.80	11.47	7.31	4.67	5	5.02
M_Refine	5.77	10.67	3.01	15.48	3.05	25.1	3.64	11.29	2.07	11.93
CRLB	19.99	3	22.36	3	27.69	3	12.58	3	9.41	3
NDBL	30.77	10.58	29.75	15.35	30.18	24.9	28.57	11.2	28.89	11.83
Nearest	31.86	3	32.7	3	43.27	3	18.77	3	12.95	3

3.3.4.3 Network size

To check the impact of changing the network size on the performance of localisation algorithms, three experiments were performed using different network sizes (small, normal and large), where nodes were distributed randomly using 0.75 beacon per r_{tx}^2 and four unknowns per r_{tx}^2 The size of networks and the number of nodes are specified as follows:

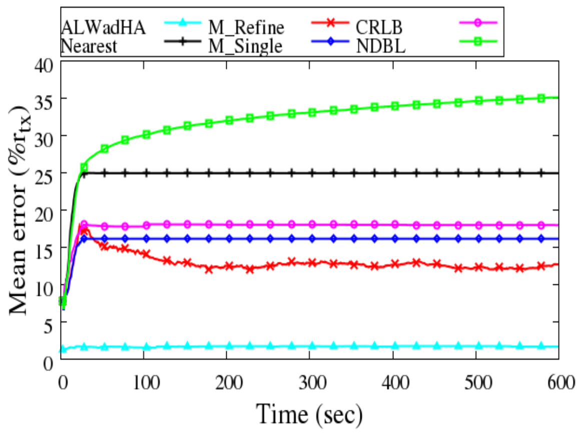
- Small: $100\,m \times 100\,m$ field with three beacons and 16 unknowns
- Normal: $200\,m \times 200\,m$ field with 12 beacons and 64 unknowns
- Large: $600\,m \times 600\,m$ field with 108 beacons and 576 unknowns.

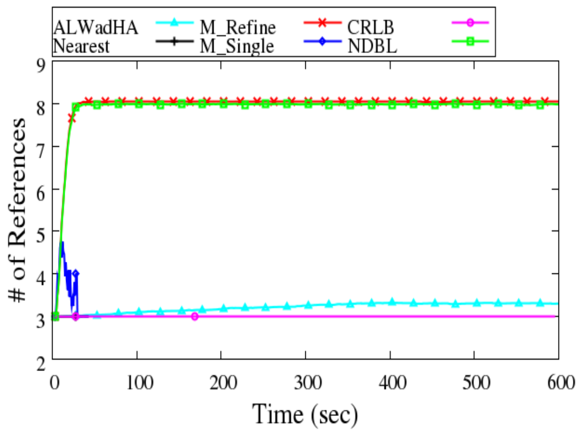
The large network follows the same characteristics as those mentioned earlier for random deployment, except that the network is divided into 36 sub-areas, while the small network consists of only one sub-area. Figure 3.23 shows that the CRLB and Nearest algorithms perform better in small networks; increasing the size of the network reduces the accuracy of these algorithms. In contrast, the accuracy of the M_Refine and NDBL algorithms is enhanced by increasing the network size, because in a small network (Figure 3.23.b) the number of references used by these two algorithms is lower than in normal and large networks. This small number of references is not enough for these two algorithms to enhance the accuracy of position estimation. The accuracy of the ALWadHA algorithm is almost the same in the three experiments.

Table 3.6 shows the mean error and average number of references used by the localisation algorithms for the three experiments done in this section.

Table 3.6. Performance comparison of different network sizes

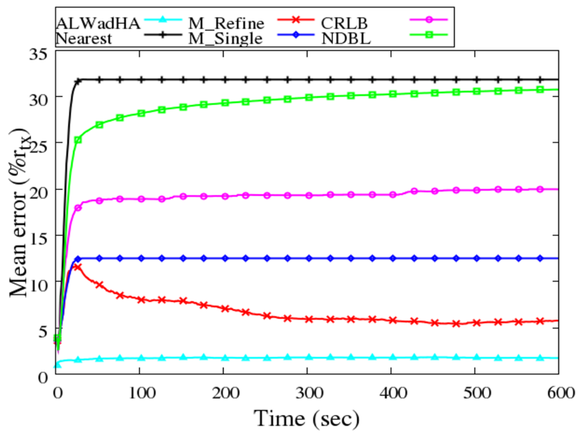
Network size	**Small**		**Normal**		**Large**	
Algorithms	**Mean error**	**Average # Ref**	**Mean error**	**Average # Ref**	**Mean error**	**Average # Ref**
ALWadHA	1.7	3.22	1.75	3.24	1.70	3.25
M_Single	16.15	3.77	12.53	4.38	16.75	4.86
M_Refine	12.71	7.95	5.77	10.67	4.42	12.55
CRLB	17.98	3	19.99	3	23.67	3
NDBL	35.11	7.89	30.77	10.58	26.96	12.43
Nearest	24.94	3	31.86	3	40.95	3

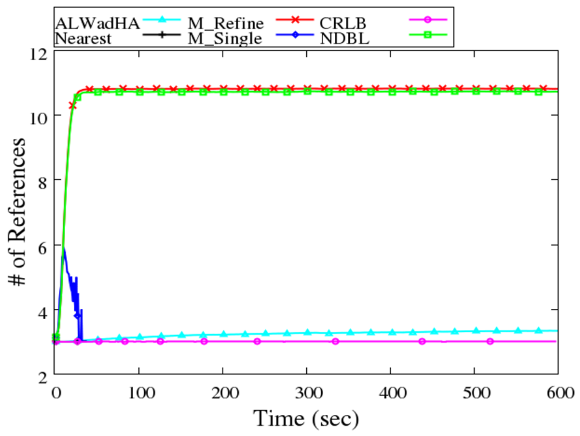




a. Mean error as a ratio of transmission range b. Average # of references

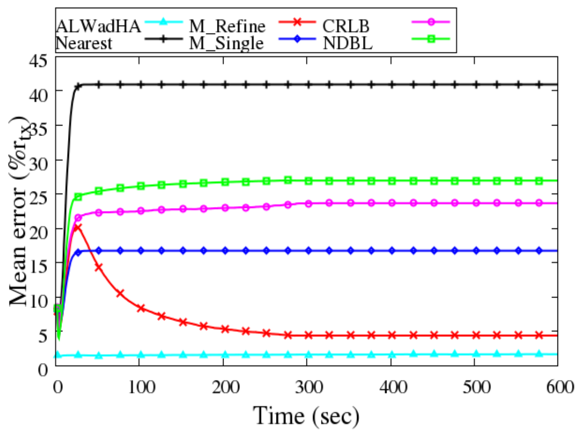
Figure 3.23. Small network

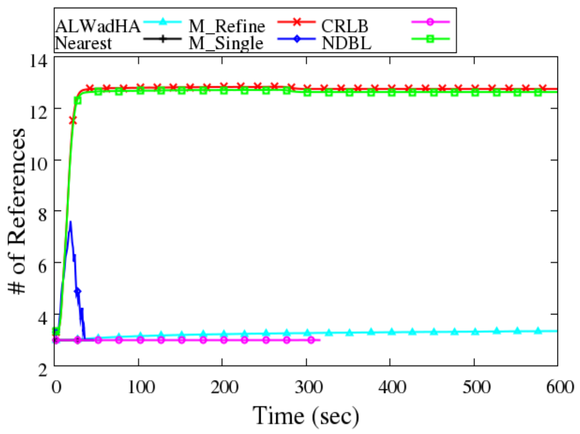




a. Mean error as a ratio of transmission range b. Average # of references

Figure 3.24. Normal network



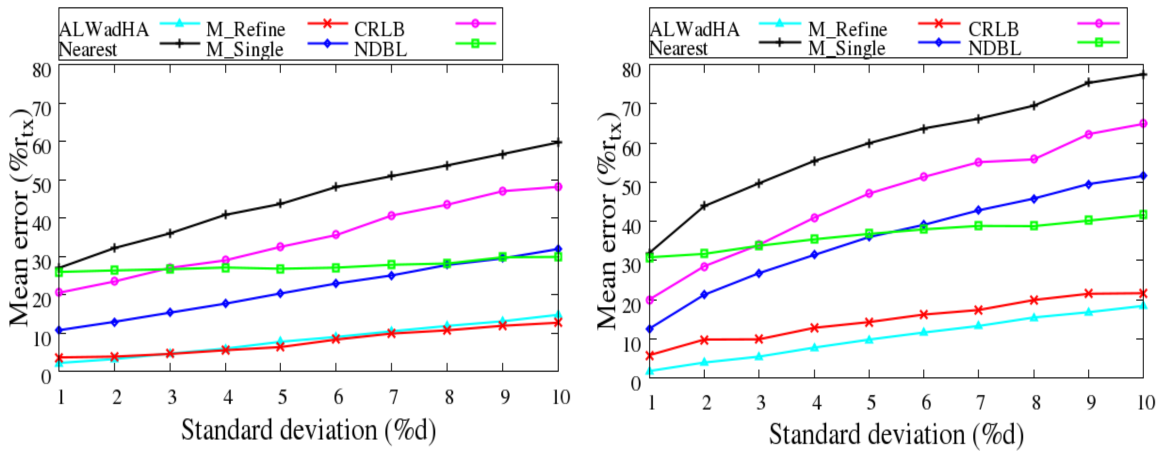


a. Mean error as a ratio of transmission range b. Average # of references

Figure 3.25. Large network

3.3.4.4 Distance-measurement error

The first experiment (Figure 3.14) was performed without adding any error to the measured distance, while in the rest of the experiments an error was added with a standard deviation equal to 1% of the measured distance. In this section, the value of the standard deviation is changed from 1% to 10% of the measured distance, in order to check the robustness of the localisation algorithms with the existence of error. Two networks were simulated: grid and random deployment in a field of $200\,m \times 200\,m$ with the same setup characteristics mentioned earlier. For each network, 10 experiments were performed with different standard deviation, then the mean error was recorded at the end of the run time. Figure 3.26 shows the mean error vs the standard deviation in the grid and random deployment. All the localisation algorithms performed better in grid deployment, as explained earlier. The mean error of the single-estimation algorithms (i.e. CRLB, M_Single and Nearest algorithms) increased dramatically with the increased distance-measurement error, while the successive-refinement algorithms (ALWadHA, M_Refine and NDBL algorithms) were more robust than the single-estimation algorithms. This result shows the main advantage of using the successive refinement approach in localisation algorithms.

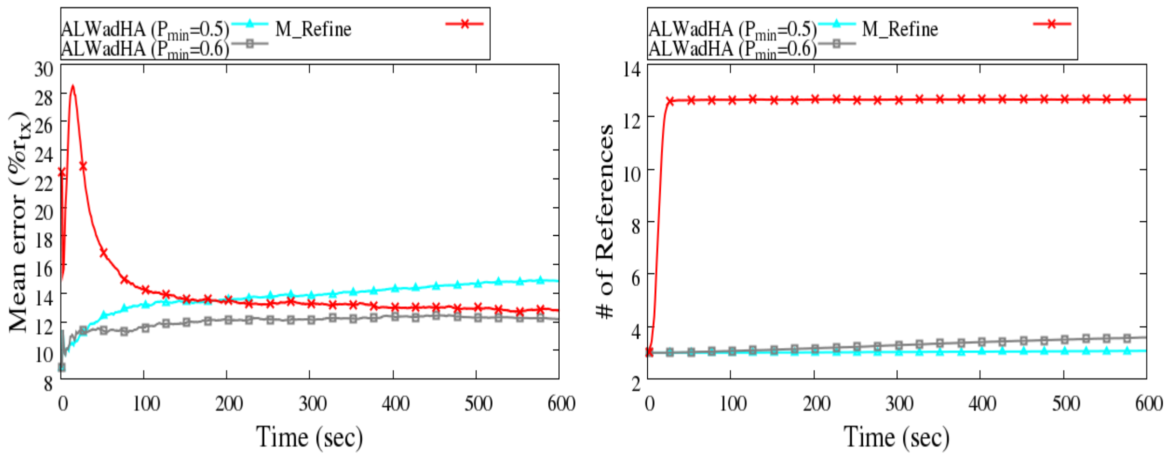


a. Grid deployment b. Random deployment

Figure 3.26. Mean error vs standard deviation

As shown in Figure 3.26, in the random deployment the ALWadHA algorithm has the lowest mean error. However, in grid deployment the M_Refine algorithm started to outperform the ALWadHA algorithm slightly when the distance-measurement error was increased. In the previous experiments, the ALWadHA algorithm was run with P_{min} equal to

0.5. To show the impact of changing the value of P_{min} and to show how the ALWadHA algorithm could outperform the M_Refine algorithm, an experiment was performed using the same grid network used in Figure 3.26.a with a standard deviation equal to 10% of the measured distance. Figure 3.27 shows that M_Refine uses an average number of references equal to 12.46 to outperform the ALWadHA algorithm with $P_{min} = 0.5$. Rerunning ALWadHA with $P_{min} = 0.6$ makes it outperform the M_Refine algorithm. Increasing the value of P_{min} allows the nodes to select a slightly higher number of references, as shown in Figure 3.27.b, which enhances the accuracy of the ALWadHA algorithm.



a. Mean error as a ratio of transmission range b. Average # of references

Figure 3.27. The impact of changing P_{min}

3.4 CHAPTER CONCLUSIONS

This chapter proved that using a proper selection method that is able to select the best set of references could relieve the burden of using all of the available references to enhance the accuracy of location estimation. Several experiments were performed by changing node deployment, node density, network size and distance-measurement error. The results of these experiments show that the ALWadHA algorithm has excellent accuracy compared with other localisation algorithms, in spite of its using a low number of references -- almost equal to the minimum possible number. This reduction in the number of references greatly improves the computation, communication, energy consumption, security, robustness and accuracy of a localisation algorithm.

The results of the experiments done in this chapter showed that the ALWadHA algorithm satisfies the following design objectives: accuracy, self-organising properties, simplicity and robustness. The ALWadHA algorithm achieved excellent estimation accuracy under different scenarios. The effective performance of ALWadHA in the two types of deployment (grid and random deployment) proved the self-organising properties of this algorithm, which also does not require any specific beacon placement. The simple computations required by the ALWadHA algorithm and the low number of references used can be considered as good evidence of the simplicity of this algorithm. Simulation results also show the robustness of ALWadHA, even when the distance-measurement error is increased. The next two chapters will discuss more evidence confirming the achievement of these design objectives and others.

Chapter 4

INFORMATION FUSION PROPERTIES OF THE LOCALISATION SYSTEM

Information fusion in WSNs is crucial for location discovery, and several location-discovery algorithms have used information fusion techniques to enhance and simplify the position computation process. This chapter provides an overview of such techniques; it distinguishes between distributed and localised algorithms and specifies three conditions to be complied with by localised algorithms. Several approaches can be used by localised information-fusion algorithms to satisfy these three conditions. The chapter will show how ALWadHA, defined in the previous chapter, uses several techniques to implement information fusion. Simulation results confirm that these techniques lead ALWadHA to achieve the most important objectives of information fusion: improving accuracy, reducing communication overheads and saving energy.

4.1 INFORMATION FUSION

Sensor nodes may be deployed to gather relevant data, either to provide a better understanding of the behaviour of the monitored entity, or to detect the occurrence of possible events. Information fusion is concerned with the way in which the data, once gathered, can be processed to enhance its relevance. The basic idea of information fusion is the combining of disparate data (either raw data or processed estimates) to improve accuracy and achieve more specific estimates than individual estimates could deliver.

Dasarathy [66] defines information fusion as "encompassing the theory, techniques, and tools conceived and employed for exploiting the synergy in the information acquired from multiple sources (sensor, databases, information gathered by human[s], etc.) such that the resulting decision or action is in some sense better (qualitatively or quantitatively, in terms of accuracy, robustness, etc.) than would be possible, if these sources were used individually without such synergy exploitation". This definition is depicted in Figure 4.1.

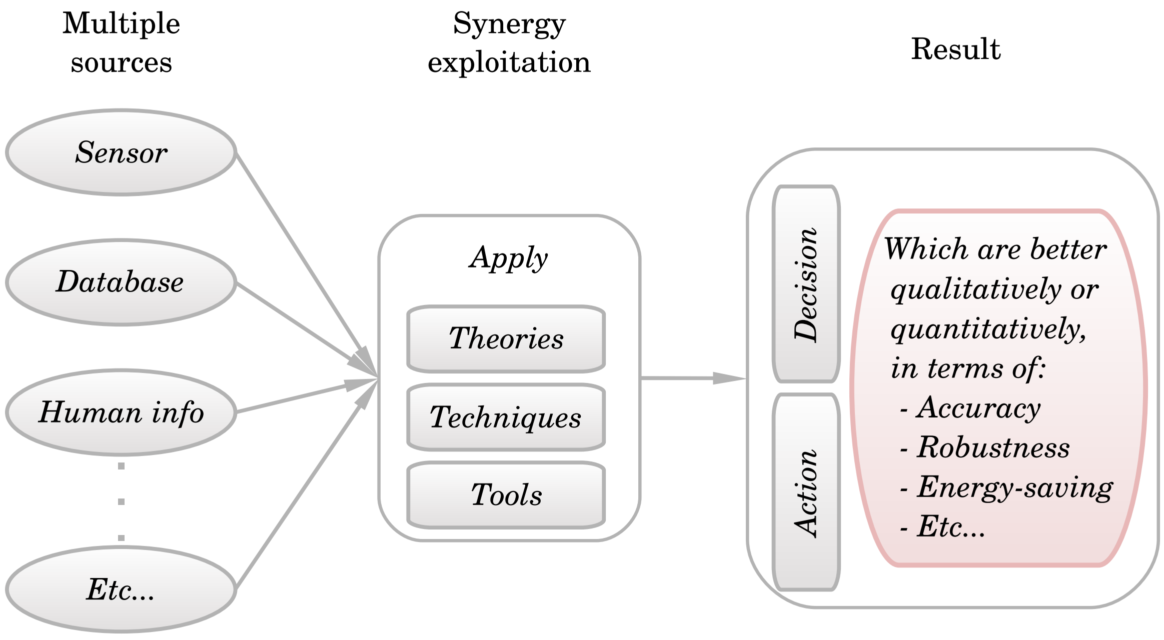


Figure 4.1. The definition of information-fusion

There are many objectives of information fusion. It can be used to compose a comprehensive view from the partial views provided by different nodes; fuse overlapping measurements to get more accurate information; and combine complementary data to allow inferences (e.g. a node can fuse the location of, and the estimated distance to, the neighbour references to obtain its position). Information fusion can reduce the overall communication load in the network and thus conserve energy and prolong the lifetime of the entire network. The minimum requirements for WSNs would be the improvement of accuracy and energy saving [67].

Two approaches to location fusion can be adopted: *centralised fusion* and *decentralised (distributed) fusion*. Theoretically, a centralised fusion system should outperform a distributed one, because the central unit has global knowledge in the sense that all measured data is available. However, this system also requires that all the raw data (or processed estimate) be transmitted from the nodes to the central unit. Such a high volume of communications might not be practical and might consume too many system resources.

In a decentralised or distributed fusion system, each node has its own processing facility, which cuts out the requirement for any central fusion or central communication facility. In this approach, each node estimates its position and the fusion occurs locally at each node, based on local observation and the information received from neighbouring nodes. The main advantages of a decentralised fusion system are that it reduces the communications overheads and thus overcomes the problem of limited communication bandwidth. It avoids the effect of centralised computational bottlenecks, which makes this approach scalable. It is also adaptable to dynamic changes in the network structure and to the addition or loss of sensing nodes. In view of the nature of WSNs, with their limited resources and bandwidth, these advantages make the distributed algorithms preferable to centralised algorithms.

Localised algorithms are a special type of distributed algorithm in which only a subset of nodes in the WSN is invoked for a specific task (e.g. sensing, tracking, reasoning, communication and computation) [17]. Localised algorithms dramatically reduce redundant processing and communication, and thus save power and prolong the lifetime of the network, which could make them the best solution for WSNs. However, choosing the proper subset of nodes to participate efficiently in a specific task is not a minor problem. Their scalability, robustness and energy-effectiveness have attracted several researchers to use localised algorithms to develop various protocols for WSNs, such as directed diffusion [68], Sensor Protocols for Information via Negotiation [69] and Collaborative Signal and Information Processing [70]. Meguerdichian *et al.* [17] developed a generic localised algorithm for location discovery in WSNs.

4.1.1 Information fusion and localisation systems

As explained in Section 2.2.1, localisation systems consist of three major components: distance/angle estimation, position computation and a localisation algorithm [18]. In order to explain the general role of information fusion within localisation, two supplementary components were added: a localised algorithm and an information-fusion technique, as shown in Figure 4.2.

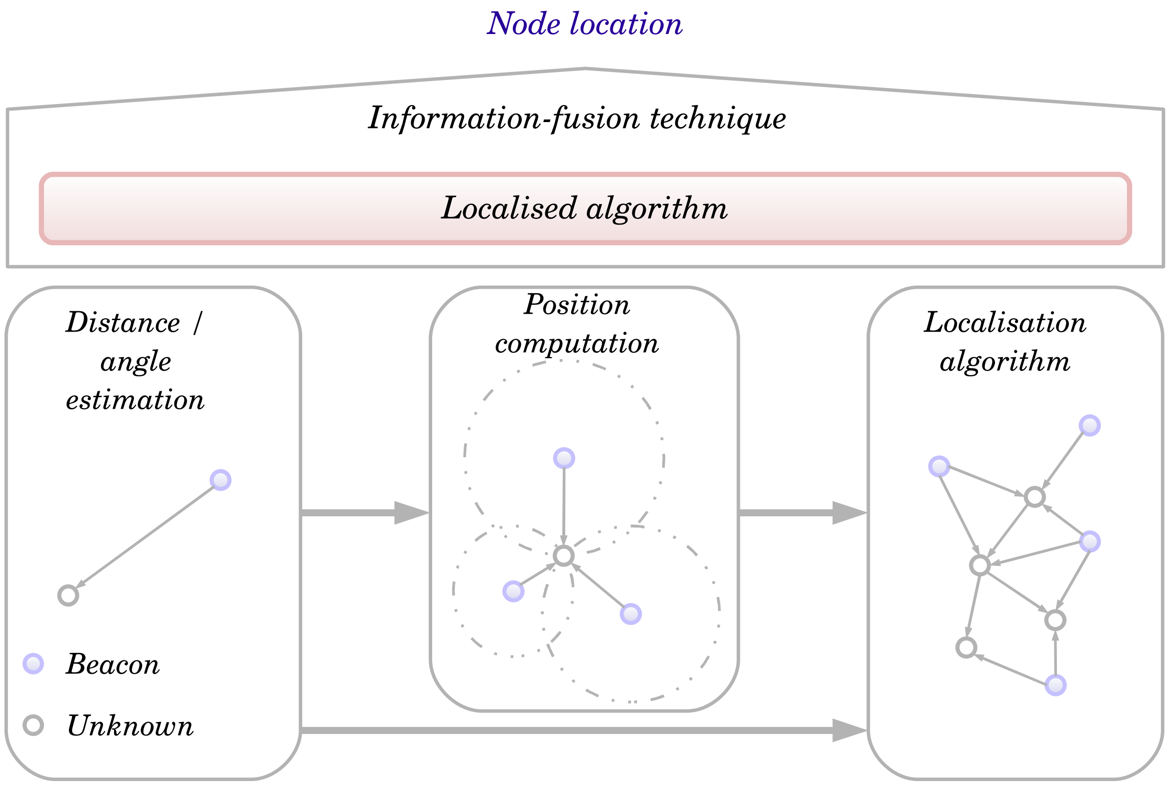


Figure 4.2. Components of localisation systems

Several information-fusion techniques have been used by various localisation algorithms to attain certain objectives, such as enhancing the accuracy of position estimation, reducing the required communication and computational requirements, and saving energy.

4.2 INFORMATION-FUSION TECHNIQUES FOR LOCATION DISCOVERY

Certain information-fusion techniques (such as Bayesian inference, maximum likelihood, least squares (LS), moving average filter, Kalman Filter, particle filter and occupancy grid) have been used by localisation algorithms to enhance the performance of the position discovery process. This section will explain these techniques briefly and mention some localisation algorithms using them. More details about these techniques can be found in [67].

- **Bayesian inference:** Bayesian inference offers a formal way to combine evidence according to the rules of probability theory. The uncertainty of systems can be represented in terms of conditional probability, which estimates the degree of belief in the [0, 1] interval, where 1 represents absolute belief while 0 represents absolute disbelief. The authors of [71] propose a localisation algorithm using Bayesian inference to process information from one mobile beacon. The unknown node uses the beacon's position and the RSS measurement to construct a constraint on its position estimate, and then it applies Bayesian inference to compute its new position estimate.

- **Maximum likelihood:** When the state being estimated is not the outcome of a random variable, then the MLE technique can be used. The likelihood function can be defined as the probability density function of the observation sequence given the true value of a certain state. Several localisation algorithms [29, 36, 41, 57, 58, 72] use MLE to estimate the position of unknown nodes by minimising the difference between the measured distances and the estimated distances, assuming that the unknown node has an adequate number of beacons. The MMSE from a set of distance measurements can be used to find the MLE of the unknown node's position.

- **Least squares:** The LS technique is a mathematical optimisation technique that searches for a function that best fits a set of input measurements. This is achieved by minimising the sum of the square error between points generated by the function and the input measurements [67]. Li *et al.* [73] propose the use of least median squares (LMS) as an improvement over LS for achieving robustness to attacks. However, LS outperforms LMS in the absence of attacks. In view of the measurement noise and error propagation that are introduced by the iterative techniques, Liu *et al.* [24] propose a robust least square (RLS) technique for localisation, which considers the error at each iteration. The authors show that RLS is more stable than LS in the presence of measurement noise.

- **Kalman filter:** The Kalman filter was proposed by Kalman [74] in 1960. Since then it has been the subject of extensive research and applications. The Kalman filter is a set of mathematical equations that provides an efficient computational solution of the least-squares method. The filter is a popular information-fusion method, because it supports estimations of desired states even when the precise nature of the modelled system is unknown. The Kalman filter technique has been applied in several distance estimation and location-discovery algorithms [75-77].

- **Particle filter:** The particle filter is recursive implementations of statistical signal processing known as sequential Monte Carlo methods [78]. The Kalman filter provides an effective solution to the linear Gaussian filtering problem. However, for a non-linear model, or non-Gaussian noise, the particle filter should be used. The authors of [79] show that particle filters allow great flexibility when addressing the problem of positioning, and they can be used in non-linear and non-Gaussian applications. Several algorithms [80-83] use the particle filter for refinement of node position estimates or for obtaining node location.

- **Moving average filter:** The moving average filter [84] is a very simple filter to use and to understand, making it the most commonly used in digital signal processing. This filter is also optimal for reducing random noise while retaining a sharp step response. Blumenthal *et al.* [54] propose a new distance estimation method using an exponentially weighted moving average filter to flatten the resulting sequence of distance estimates and to filter outliers. The filter uses multiplying factors to give different weights, which change exponentially, to different beacon locations based on the estimated distances.

- **Occupancy grid:** The occupancy grid is a multidimensional random field that maintains stochastic estimates of the occupancy state of the cells in a spatial lattice [85]. The basic idea of the occupancy grid is to represent a multidimensional map of the environment as evenly square or cubic cells which have random variables indicating the presence of an obstacle at that cell. The cell's probability of being occupied can be computed using Bayesian theory or fuzzy set theory [86], based on information provided by several sources. Wongngamnit and Angluin [87] propose a new robot localisation algorithm using the occupancy grid concept. Readers can find more information about using the occupancy grid for positioning estimation in [88].

Information fusion can play two roles in the localisation algorithms: a supporting role and a leading role. In the supporting role, information fusion acts as a tool to assist the localisation algorithms, by using one of the information fusion techniques to assist in the location discovery. In the leading role, the localisation algorithms are designed to support an information-fusion application. One or multiple information-fusion techniques may be executed to accomplish the application's objectives. Rather, these techniques are responsible for guiding the location discovery process and the fusion process simultaneously. This means the localisation algorithm should be designed with two objectives: location discovery and achieving information fusion.

4.3 LOCALISED INFORMATION-FUSION ALGORITHMS

The basic idea of using localised algorithms is not only to request and process information to estimate position locally, but also to use data only from nodes that are likely to contribute to rapid and accurate formation of the final position estimate [17]. In other words, the localised algorithms used for location discovery should consider the following three conditions:

- **Firstly**, request and process information with regard to the localisation algorithm only locally; i.e. report the location of the unknown but do not send the raw data to a centralised entity for processing.
- **Secondly**, only a subset of nodes takes part in the position estimation process.
- **Thirdly**, only the references that are most likely to contribute to accurate position computation of an unknown are selected.

Developing practical localised information-fusion algorithms for WSNs is very important and a challenging task for a number of reasons [17]: The communication delay in WSNs is significantly greater than that of other traditional networks, since communication consumes more energy than sensing and computation; dynamic changes occur in WSNs due to nodes dying or extra deployment; and nodes are not always able to participate in every task (e.g. due to lack of energy, obstacles, etc.). A WSN is resource constrained in terms of resources such as processing, communication and energy. Finally, security issues might require that only a subset of nodes take part in a task, thereby simplifying the task of ensuring that all nodes participating in the process are authentic.

Localised location discovery requires only the position estimates from a subset of the unknown's neighbours, and need not involve all other nodes in the network. This makes the system-wide location-discovery task a good candidate for distributed algorithms.

However, not all distributed localisation algorithms proposed in the literature can be considered as localised algorithms. In fact, most of these algorithms satisfy only the first two of the three conditions mentioned above. To satisfy the third condition, localised information-fusion algorithms should select only references that are likely to contribute to accurate position estimates (i.e. use only a subset of available references to compute the position instead of using all of them). In this type of algorithm, information fusion plays a leading role, since it not only assists in the position estimation process, but also accomplishes fusion’s objectives (such as energy saving and accuracy improvement).

Different approaches have been used to select a subset of references. Selecting a subset of references with low error is the approach followed by [38, 39]. Cheng *et al.* [37] simply choose the nearest three references to estimate nodes' position. In other approaches [41, 56], the node selects a subset of references based on a malicious-node removal approach, where the node tries to detect and prevent the malicious nodes from participating in its location estimate. Albowicz *et al.* [40] propose a localisation algorithm for choosing a reliable subset of references based on a reference consistency approach. Lieckfeldt *et al.* [13] consider the impact of geometry to select a subset of references. Costa *et al.* [42] propose a localisation algorithm that selects a subset of references based on a noisy distance measurement in order to avoid the biasing effect of a noisy environment. Section 2.3 discussed these approaches in more detail, while Section 2.4 compared them and highlighted the advantages and disadvantage of these approaches.

A researcher developing practical localisation algorithms for WSNs may take into account several design objectives, including self-organising properties, robustness, energy efficiency, localised information fusion and security. However, selecting the most accurate subset of references is not enough to fulfil all of these design objectives; in addition this subset could be the same as, or a little smaller than, an all-references set. The ALWadHA algorithm, as mentioned above, uses a method that selects almost the minimum possible number of references to achieve an accurate position estimation. This selection method is based on the low-error approach. The rest of this chapter extends the investigation of the ALWadHA algorithm. It shows the impact of using a proper localised selection method on achieving several design objectives, enhancing the performance of localisation systems and making information fusion play a leading role.

4.4 THE THREE FILTERS OF ALWADHA

The ALWadHA algorithm uses three types of filters to satisfy the three conditions required for it to be considered as a localised algorithm. These filters also help to achieve other design objectives, such as energy efficiency (by reducing the computation and communication overheads), accuracy, robustness and security. These three filters also make information fusion play a leading role, not only a supporting role. Figure 4.3 illustrates these three filters. The first filter is used by the known nodes to decide if they will respond to the “location request” packets -- in other words, act as references -- while the other two filters are used by the node itself to select the proper subset of references.

4.4.1 Filter one

This filter is used by a known node, which has received a “location request” packet, to decide if it will act as a reference node and send a “location response” packet. A known node should satisfy two conditions in order to act as reference node. Firstly, its probability of accuracy should be more than a specific value (P_{res}). Secondly, its probability of accuracy should be more than the required accuracy level sent by the requesting node (L_{acc}). Applying this filter at known level rather than at node level will reduce the cost of communication, since instead of eliminating those references with a low probability of accuracy at node level, those knowns will not send their responses and this will reduce the messages propagated through the network. This filter does not require any interaction between neighbouring knowns; rather it works on the available information. The requesting node will receive “location responses” packets only from R set of references where $R \subseteq (K \cup B)$.

4.4.2 Filter two

The node applies this filter to select a subset of references S, where $S \subseteq R$ based on the references' location error. The probability of accuracy of the subset S should be greater than a certain value (P_{min}). As shown in Figure 4.3, the node applies this filter twice firstly to select a subset S^0, where $S^0 \subseteq R$ in order to estimate an initial position $(\hat{z}^{\cdot})$ and secondly, to select a subset S, where $S \subseteq \bar{R}$, and $\bar{R}$ is the output set of filter three in order to estimate the refined position $(\hat{z})$.

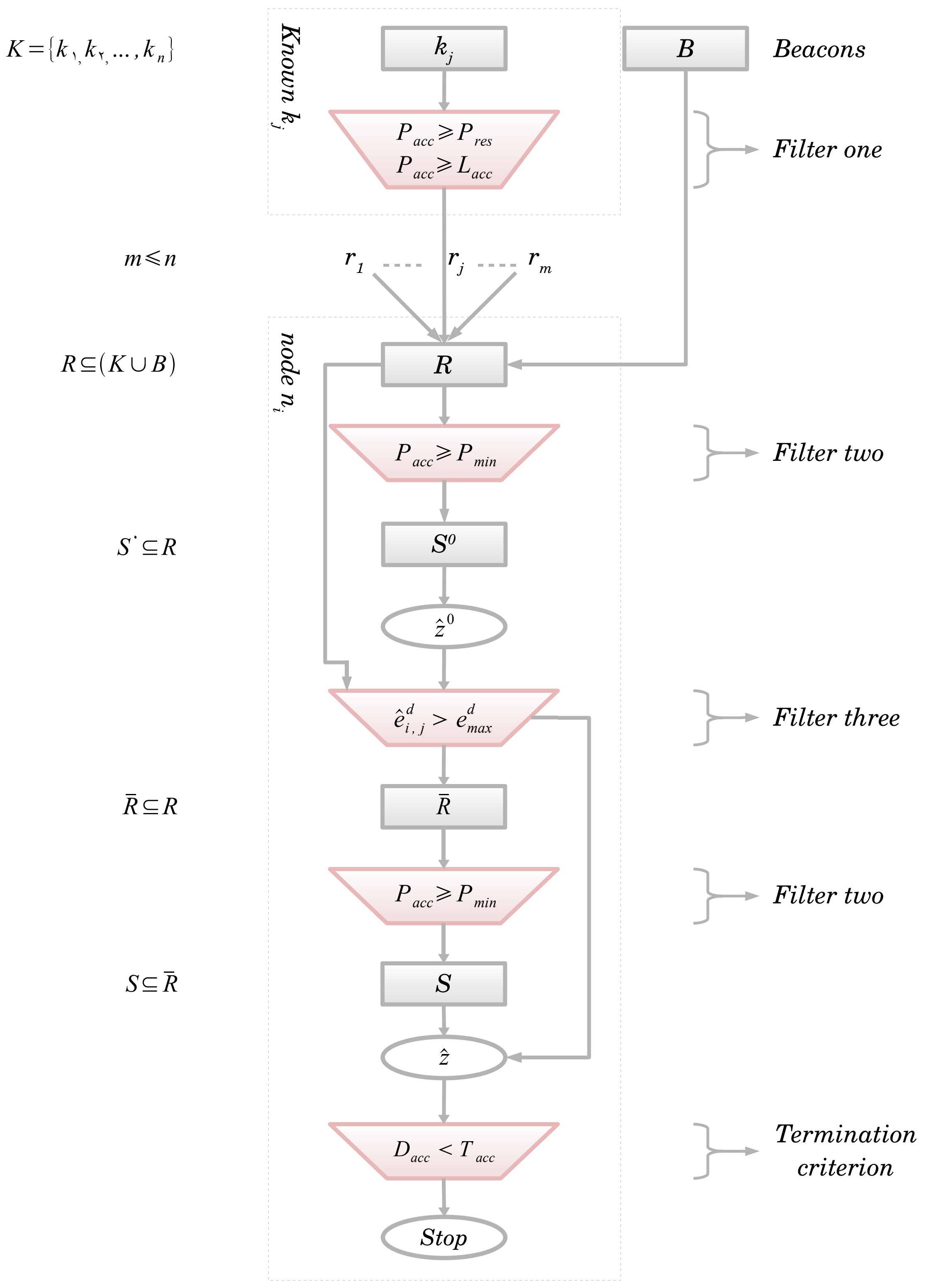


Figure 4.3. The three filters used by ALWadHA

4.4.3 Filter three

The node uses the initial position $(\hat{z}^0)$ to eliminate those references with high distance-measurement error; the result of this filter is $\bar{R}$, where $\bar{R} \subseteq R$. This filter is only applied if at least one of the references in the subset S^0 has an estimated distance error greater than a certain error value (e^d_{max}) .

4.5 INFORMATION FUSION IN A LEADING ROLE

As mentioned earlier in Section 4.2, information fusion can play two roles in the localisation algorithms: a supporting role and a leading role. In order to distinguish between these two roles in this discussion, the information fusion used by localisation algorithms will be classified into three levels, based on the objectives that can be achieved. In level one, localisation algorithms use one of the information-fusion techniques, explained in Section 4.2, in the position computation. The objective of information fusion on this level is to combine complementary data to allow inferences: the node fuses the location of, and the measured distance to, at least three references to determine its position. On this level, information fusion plays only a supporting role to assist in the location discovery. On level two, information fusion starts to play a leading role by achieving some of the information-fusion objectives: accuracy, robustness, fewer computations and security. However, reducing the communication overhead is not targeted by this level of information fusion. On level three, information fusion also plays a leading role by achieving several objectives. The main objective of information fusion on this level is to reduce the communication overhead in order to enhance the energy efficiency of the localisation algorithm. Developing an efficient localisation algorithm that employs the three levels of information fusion will enhance the performance of this algorithm and assist in achieving several design objectives.

Therefore, as shown in Figure 4.4, the ALWadHA algorithm employs the three levels of information fusion. On level one, the ALWadHA algorithm uses the MLE technique, where the MMSE is used to determine the MLE of the unknown nodes' position. On level two of information fusion, the ALWadHA algorithm uses filter two and filter three. These two filters assist in achieving several objectives, including accuracy, simplicity, robustness, reduced computation cost, localised algorithm and security. In order to accomplish level three of information fusion, the ALWadHA algorithm follows two approaches. Firstly, the sensor node uses the termination criterion to halt the process of localisation when it has determined its position with good accuracy. This termination criterion reduces the communication overhead by cutting out unnecessary "location request" packets. Secondly, known nodes use filter one to decide if they will act as references based on their accuracy. This filter reduces the communication overhead due to unnecessary "location response" packets. In addition to other objectives, these two approaches enhance the energy efficiency of the ALWadHA algorithm.

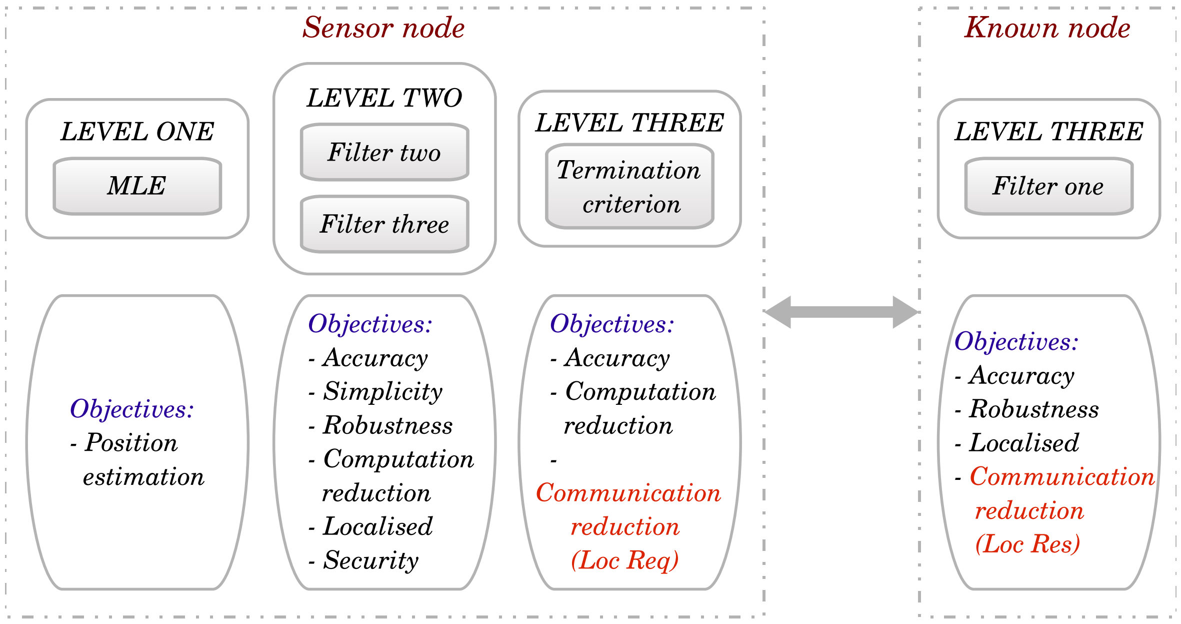


Figure 4.4. The three levels of information-fusion used by ALWadHA algorithm

4.6 SIMULATION

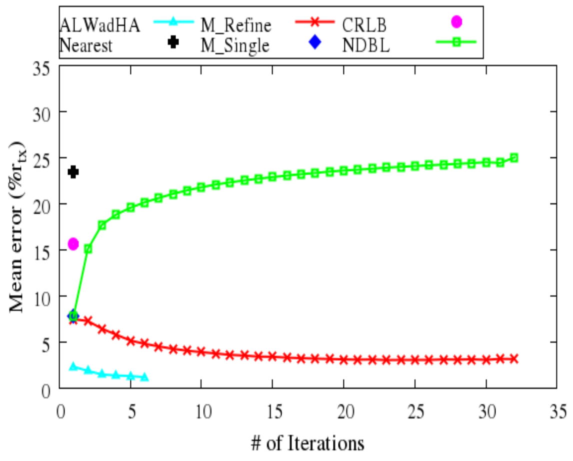
This section will evaluate the ALWadHA algorithm using the same grid and random deployment as described in Section 3.3.3. Several experiments were performed to evaluate the performance of the ALWadHA algorithm compared with other localisation algorithms. The performance of each algorithm was evaluated based on the following metrics: The localisation error at each iteration, number of "location request" packets, number of "location response" packets and the energy consumed.

4.6.1 Localisation error vs number of iterations

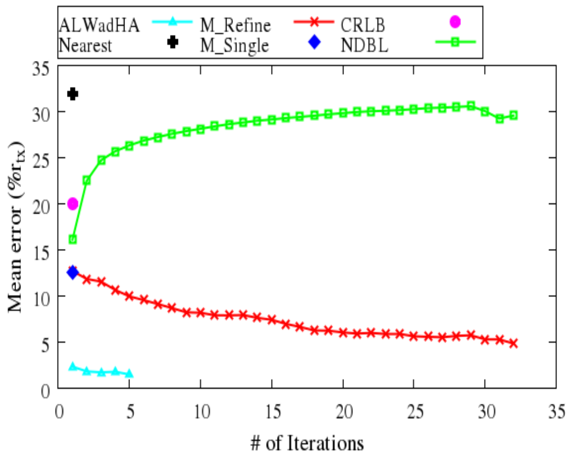
Refinement algorithms, such as M_Refine, NDBL and ALWadHA algorithms, perform several iterations of position estimation. In this section, the mean error is shown as a ratio of transmission range at each iteration. CRLB, M_Single and the Nearest algorithms are based on a single-estimation approach, in which the node estimates its position only once. The mean error at a specific iteration *I* is equal to the summation of location error of knowns that estimate their location at iteration *I*, divided by the number of these knowns, and then it is divided by the transmission range as follows:

$$Mean\ error_I = \left(\frac{1}{n} \sum_{i=1}^{n} \| \hat{z}_i - z_i \| \right) \frac{1}{r_{tx}} \ 100\% \tag{4.1}$$

where *n* is the total number of knowns that estimate their position at iteration *I*. Figure 4.5 shows that in the ALWadHA algorithm the nodes perform at the most six iterations before they establish their final position. This shows that the ALWadHA algorithm not only uses a low number of references, but requires a low number of iterations to get an accurate position; which also reduces the computation cost dramatically. In the grid deployment (Figure 4.5.a), the mean error using the ALWadHA algorithm at the sixth iteration is equal to 1.18%, using an average number of references equal to 3.42, while after 32 iterations the mean error of the M_Refine algorithm is equal to 3.14%, using an average of 12.46 references. Figure 4.5.b shows that none of the other algorithms could achieve the same level of accuracy that the ALWadHA algorithm achieves within only five iterations.



a. Grid deployment

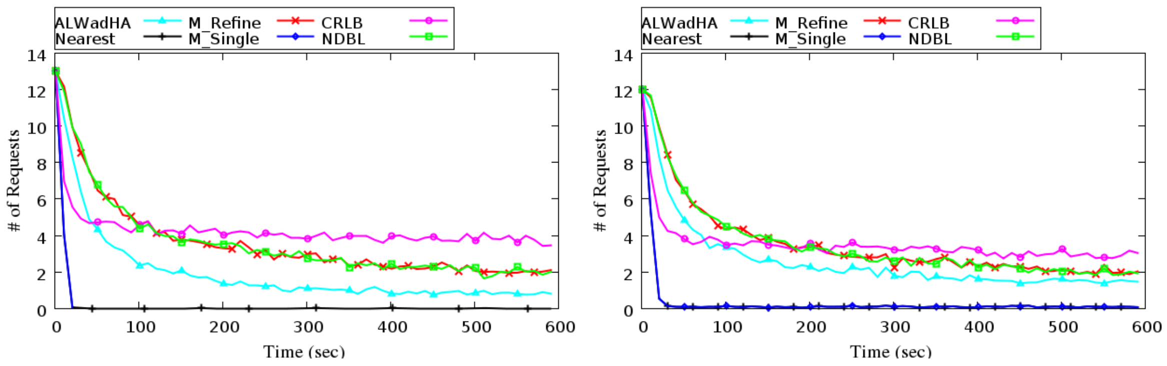


b. Random deployment

Figure 4.5. The mean error at each iteration

4.6.2 Number of “location request” packets

The node starts the localisation process by broadcasting a “location request” packet; if it cannot estimate its position it will again broadcast another “location request” packet after a specific time (for instance 5 seconds). In the refinement algorithms the node increases this time after each estimation. Figure 4.6 shows that the ALWadHA algorithm broadcasts a lower number of “location request” packets compared with CRLB, M_Refine and NDBL algorithms because of the termination criterion that has been followed by the ALWadHA algorithm. The M_Single and the Nearest algorithms require a low number of “location request” packets because they are based on the single-estimation approach, in which the node estimates its position only once.

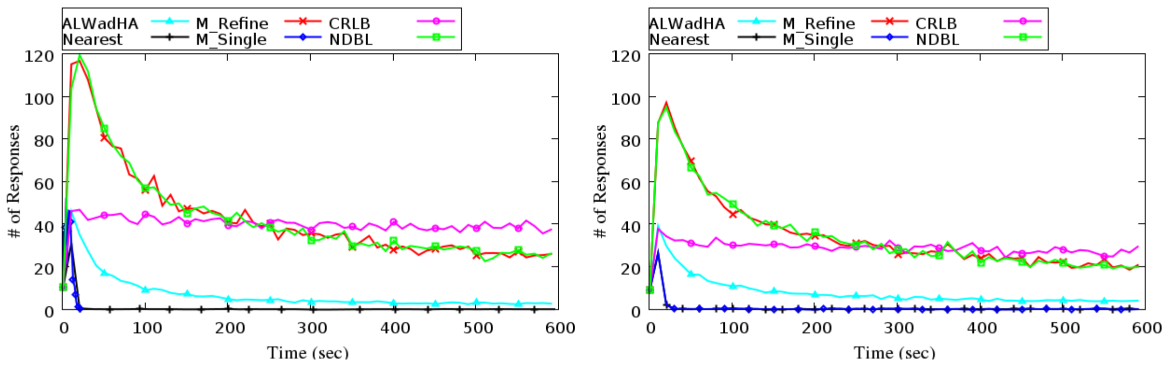


a. Grid deployment b. Random deployment

Figure 4.6. Number of “location request” packets

4.6.3 Number of “location response” packets

Figure 4.7 shows the number of “location response” packets vs time. This figure shows that the ALWadHA algorithm requires a lower number of “location response” packets than CRLB, M_Refine and NDBL algorithms, because of the “filter one” that the knowns use to decide if they will send their response or not, as explained in Section 4.4.1.

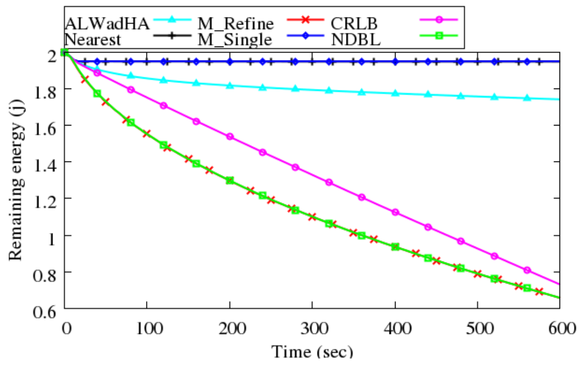


a. Grid deployment b. Random deployment

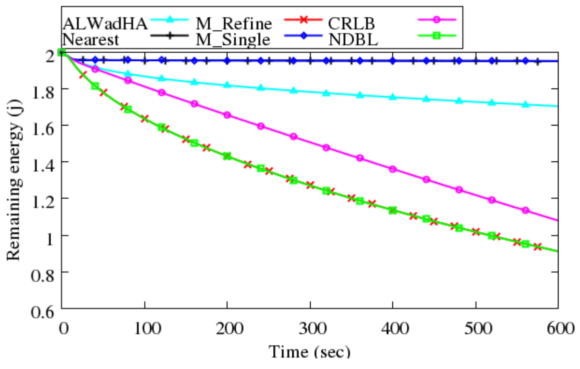
Figure 4.7. Number of "location response" packets

4.6.4 Remaining energy

At the beginning of the simulation each node has 2.0 joule. Figure 4.8 shows the average remaining energy vs time, considering only energy consumption due to communication. Since the ALWadHA algorithm requires a lower number of “location request” and “location response” packets, as shown in Figure 4.6 and 4.7, it consumes less energy compared with the CRLB, M_Refine and NDBL algorithms. The difference is expected to be higher if the energy consumption due to computation is also included, for two reasons. Firstly, the ALWadHA algorithm uses a lower number of references. Secondly, the ALWadHA algorithm requires a lower number of iterations to get an accurate position estimation (Figure 4.5). These two characteristics could reduce the computation cost dramatically, especially for WSNs. Unlike other algorithms, ALWadHA consumes less energy in the grid deployment, because the higher number of references around each node enables it to reach its final position faster than if the deployment was random, and thus it stops sending the "location request" packets.



a. Grid deployment



b. Random deployment

Figure 4.8. Remaining energy

4.6.5 Performance comparison

Table 4.1 shows a performance comparison between the implemented localisation algorithms in both grid and random deployment. The mean error is recorded at the last iteration shown in Figure 4.5. The average number of requests and the average number of responses represent their average during the run time. Finally, the remaining energy at the end of the run time is also listed. The results shown in this table confirm the objectives of employing the third level of information fusion in localisation algorithms. The termination criterion made ALWadHA broadcast a lower number of "location request" packets than other refinement algorithms. At the same time, filter one reduced the number of "location response" packets dramatically. For instance, in the random deployment, using ALWadHA, the average number of "location response" packets is only three packets for each "location request" packet, while in CRLB, M_Refine and NDBL algorithms there are ten "location response” packets for each “location request” packet.

Table 4.1. Performance comparison

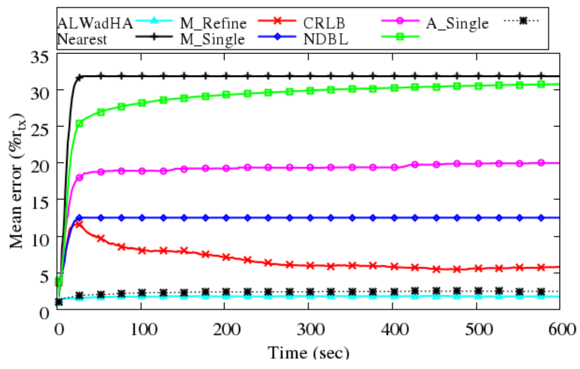
Metrics	**Mean error**		**# of Request**		**# of Response**		**Energy**	
Algorithm	**Grid**	**Rand**	**Grid**	**Rand**	**Grid**	**Rand**	**Grid**	**Rand**
ALWadHA	1.18	1.47	1.92	2.63	7.06	8.1	1.743	1.705
M_Single	7.77	12.53	0.21	0.3	0.77	0.8	1.950	1.952
M_Refine	3.14	4.83	3.61	3.47	42.96	34.49	0.657	0.912
CRLB	15.65	19.99	4.23	3.46	40.07	28.73	0.732	1.081
NDBL	25.00	29.54	3.62	3.47	42.92	34.49	0.658	0.912
Nearest	23.38	31.86	0.22	0.66	0.81	0.84	1.949	1.951

4.7 TOWARDS MORE ENERGY EFFICIENCY

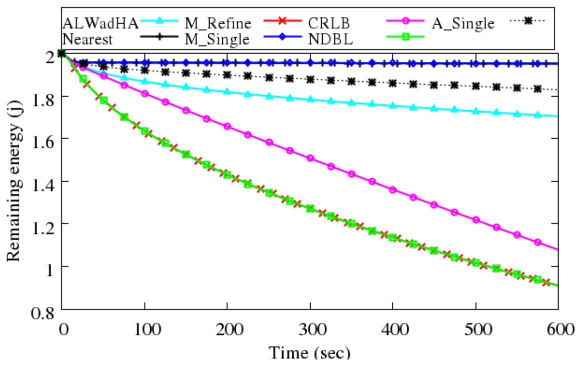
This section will investigate three methods that could be used for making ALWadHA more energy efficient. All the experiments done in this section employed random deployment.

4.7.1 Single-estimation approach

In this technique ALWadHA will consider the single-estimation approach instead of using the successive-refinement approach. In this approach the node estimates its position only once and it does not repeat the estimation to enhance the accuracy of positioning. Figure 4.9 shows that this technique (A_Single) could enhance the energy efficiency of the original algorithm (ALWadHA). On the other hand, the mean error is slightly higher than the original algorithm. However, Figure 3.26 showed that the mean error of the single-estimation algorithms increases dramatically by increasing the distance-measurement error, while the successive-refinement algorithms are more robust than single-estimation algorithms. Therefore, in a noisy environment it is not advisable to use the single-estimation approach, especially if accuracy is a critical issue.



a. Mean error as a ratio of transmission range



b. Remaining energy

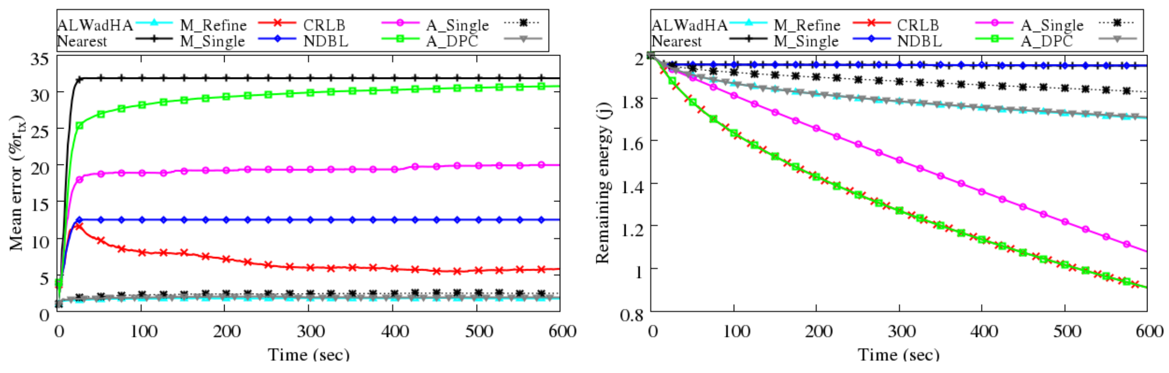
Figure 4.9. ALWadHA based on single-estimation approach

4.7.2 Dynamic power control

The dynamic power control technique can be used to adaptively change the level of transmission power based on the distance between the transmitting and receiving nodes. Instead of continuing to send the packets using the maximum transmission power to cover the entire range, the node dynamically specifies the required transmission power to enable the packet to reach the destination node. The dynamic power control technique is adopted in the proposed localisation system as follows: the node uses the maximum transmission power to broadcast the "location request" packets, to make sure all the references within its transmission range will receive this packet. The references which receive this request estimate the distance to the requesting node using the RSS and then estimate the required transmission power to enable the "location response" packets to reach the requesting node.

Figure 4.10 Shows that the use of this technique by ALWadHA (A_DPC) does not enhance the energy efficiency very much for several reasons. One reason is that this technique is used only to change the power level for sending the "location response" packets, while the nodes use the maximum transmission power to send the "location request" packets. The nodes' density is low, only 4.75 nodes per r_{tx}^2 , which means the nodes are not close to one another and therefore the transmission power is little lower than the maximum transmission power. As shown in Table 4.1, ALWadHA reduced the average number of "location response" packets dramatically compared with other algorithms (due to filter one), therefore this technique does not greatly affect the energy efficiency of the ALWadHA algorithm. However, this technique could enhance the energy efficiency of the

other localisation algorithms and also reduce the interference.



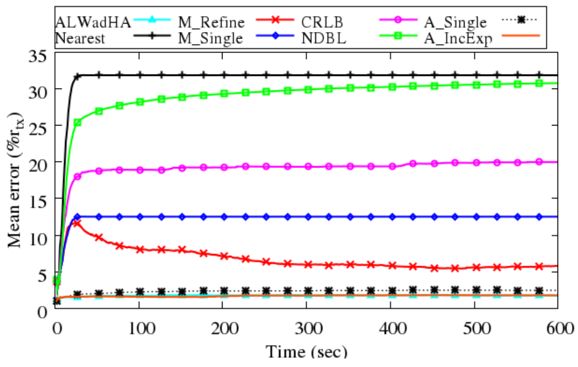
a. Mean error as a ratio of transmission range b. Remaining energy

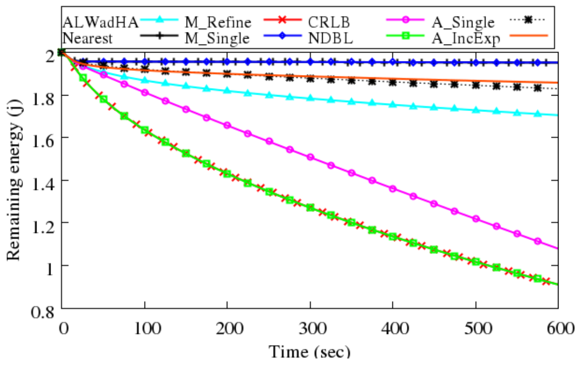
Figure 4.10. ALWadHA with dynamic power control

4.7.3 Incremental and exponential requesting rate

One target of a localisation system is to minimise the number of beacons used because of the cost or the difficulty of installing these nodes, which means only some of the nodes will be neighbours of these beacons. Initially, the nearby unknown nodes will be able to determine their position and they could act as references for other unknowns. Therefore, the unknown nodes far from the beacons need more time to determine their position compared with the ones close to them. Using a fixed requesting rate will lead to wasting the energy of sensor nodes, mainly those that are far from the beacons, because at first these will keep sending "location request" packets without getting enough "location response" packets back. Therefore, the requesting rate (Δt_{req}) is updated after each iteration, either incrementally or exponentially, as follows: if the node has determined its position at a specific iteration (or enhanced the accuracy of the current position), it will increase the requesting rate by one $(\Delta t_{req} = \Delta t_{req} + 1)$, otherwise it will multiply the requesting rate by two $(\Delta t_{req} = 2 * \Delta t_{req})$.

As shown in Figure 4.11 ALWadHA, when using the incremental and exponential requesting rate technique (A_IncExp), consumes less energy than the original ALWadHA and A_Single; moreover, the accuracy is slightly better. This technique allows the nodes close to the beacons to reach their final estimation faster, which also enhances the position accuracy of other nodes. At the same time it prevents the nodes farther away from continuing to send useless “location request” packets and thus enhances the energy efficiency of the algorithm.





a. Mean error as a ratio of transmission range b. Remaining energy

Figure 4.11. ALWadHA based on the incremental and exponential requesting rate

4.7.4 Performance comparison

Table 4.2 shows a performance comparison between the original ALWadHA algorithm and the new techniques. The mean error and the remaining energy are recorded at the end of the run time. ALWadHA using the incremental and exponential technique (A_IncExp) achieved the best result. A_IncExp still follows the successive-refinement approach, thus it is still robust even in a noisy environment.

Table 4.2. Performance comparison of the new techniques

Algorithm	Mean error	Remaining energy
ALWadHA	1.797	1.705
A_Single	2.462	1.829
A_DPC	1.900	1.710
A_IncExp	1.773	1.857

4.8 CHAPTER CONCLUSION

WSNs are regarded as resource-constrained networks, therefore this chapter emphasised that localised information-fusion algorithms should be used to provide an accurate position estimate at reduced cost. The localised algorithms should satisfy three conditions, which are that: information is requested and processed only locally; only a subset of nodes takes part in the position-estimation process; and only the references that are most likely to contribute to accurate position computation of an unknown are selected. In fact, the focus is on the impact and significance of the third condition, which enhances the accuracy of localisation systems.

Several approaches can be used by localised algorithms. A number of design parameters and certain requirements play a vital role in selecting the proper approach. However, selecting a certain approach may not always give the best position estimate (for instance because of its simplicity), and so selecting another approach with a more sophisticated technique for selecting a subset of references is required. The accuracy of position estimation is one of the most important design objectives, thus the ALWadHA algorithm follows the low-error references approach. However, ALWadHA uses several techniques to overcome the drawbacks and limitations of this approach.

The ALWadHA algorithm carefully selects a sufficient number of the best references in order to enhance accuracy at reduced cost. In order to select a proper subset of references it uses three types of filters. The first filter is used by known nodes to reduce the communication load of “location response” packets, while the next two filters are used by the node itself to select the proper subset of references. The second filter is used to handle the location error, while the third filter is used to deal with distance-measurement error. Employing these three filters enables the ALWadHA algorithm to achieve high estimation accuracy using only a low number of references, almost equal to the minimum possible number of references.

The information fusion used in localisation systems was classified into three levels, based on the objectives achieved. On the first level information fusion assists in location discovery. On the second level it achieves several objectives such as accuracy, robustness and security. However, reducing the communication overhead is not targeted on this level, which is the main objective of the third level. The ALWadHA algorithm made use of these three levels. It was shown that using the three levels of information fusion enhances the performance of the ALWadHA algorithm and assists in achieving several design objectives. Selecting the best subset of references will enhance the accuracy and simplicity of the algorithm. Filtering out those references with high location error or distance-measurement error will increase the robustness of the algorithm. These levels also reduce the computation and communication overhead and thus enhance the energy efficiency of the ALWadHA algorithm. The simulation results confirmed this achievement and showed that ALWadHA required a lower number of iterations to achieve better accuracy (due to level two and level three). It was also shown that ALWadHA required a lower number of “location request” and “location response” packets (due to level three), which reduced communication overheads and therefore enhanced energy efficiency. Finally, three techniques that could be used to enhance the energy efficiency of the ALWadHA algorithm were investigated.

Chapter 5

SECURE LOCALISATION SYSTEMS

Most of the proposed localisation algorithms studied the problem of location discovery in a non-adversarial environment. Although these types of algorithm are vulnerable to several types of security attack, less work has been done to implement secure localisation algorithms that are able to work in a hostile environment. The very nature of WSNs enables the attackers to abuse the localisation systems. Malicious nodes could claim fake locations that are not where they are physically located. An attacker could compromise, or masquerade as, a beacon node and send incorrect location information. Localisation in a hostile environment is a critical problem in WSNs because compromising the localisation system could disturb and subvert the entire functioning of the WSN.

Localisation systems consist of three major components: distance estimation, position computation and localisation algorithm. These components are strongly connected. Compromising one of them could affect the entire localisation system. However, securing the first two components greatly enhances the security of the third component and the entire localisation system.

Several techniques can be used to implement secure distance-estimation components, such as directional antennas, RF fingerprinting, centralised systems and distance bounding. A distance-bounding approach has several characteristics that make it very suitable to be used in WSNs. Distance bounding can function within an ad-hoc, mobile environment with any number of nodes and different types of network topology. Distance bounding does not consume valuable resources and it can be executed in minimal time.

Distance bounding involves only two nodes, to estimate the distance between them, which makes it a good choice for the ALWadHA algorithm. ALWadHA, as has been shown, uses a subset of a few references compared with other secure localisation algorithms. Therefore, ALWadHA can use the distance-bounding method to estimate the distance to the references within the selected subset without involving other references. This can also reduce the overhead that could be caused by estimating the distance to all available references. Using a distance-bounding approach will assist ALWadHA to accomplish several design objectives, such as self-organising properties, simplicity, energy efficiency, localisation and security.

Distance bounding can be used to implement secure distance-estimation components. However, this is not enough to secure the entire localisation system, and more techniques should be used to secure the other components. Therefore, ALWadHA relies not only on using a secure distance-bounding protocol, but also uses a robust position-computation component that tolerates the existence of malicious nodes.

5.1 THE SECURITY OF LOCALISATION SYSTEMS

Localisation systems play a key role in WSNs, because in addition to locating events they could be used as the base for routing, density control, tracking and other services and protocols. This role makes the localisation system a target of attackers. An attacker that compromises the localisation system could disturb the entire functioning of WSN and lead to incorrect plans and decisions. Therefore, as stated above, WSNs require a secure localisation algorithm, which not only solves the problem of localisation, but should also be resistant to the presence of fraudulent, malicious and/or compromised nodes.

5.1.1 Attacks on localisation systems

As mentioned in Section 2.2.1, localisation systems consist of three main components: distance/angle estimation, position estimation and localisation algorithm. These components are strongly dependent on one another. Malfunctioning in any of these components could affect the entire localisation system, and lead to incorrect estimation. Because of the strong relationship between them, any of these components can be targeted by an attack on a localisation system, making these systems very fragile and hard to secure [18].

5.1.1.1 Attacks on distance/angle estimation

Different approaches can be used for distance/angle estimation, such as directional antennas, RF fingerprinting (communication neighbour authentication), connectivity (in range) and distance bounding. Practically, these approaches use several techniques, including RSS, ToA, TDoA, AoA, RTT and hop-count based ones.

RSS is not a secure method, since attackers can easily alter transmission power by either amplifying or attenuating a signal. Changing the transmission power makes the neighbour nodes think the compromised node is nearer or farther away than it really is. In the ToA and TDoA techniques, an attacker can delay the transmission time to pretend that it is further away. On the other hand, there are no measures in place to prevent the compromised node from lying about the sending time and appearing closer to the neighbour nodes as a result. Secure AoA technique requires that the nodes must have synchronised clocks to ensure that a transmission was made to multiple references at the same time. However, time synchronisation to the accuracy required for distance estimation is a challenge in wireless networks. In the hop-count-based technique, the node first estimates the average distance to neighbouring nodes, then it uses this average distance to estimate the distance for other nodes by multiplying this average distance by the number of hops to these nodes. A compromised node can deliberately change the computed hop count to mislead other nodes about its real distance.

In a compromised environment, both RSS and ToA techniques can be targeted by changing the physical medium, for example by introducing noise, obstacles or smoke. Also, the AoA technique can be compromised by deploying magnets in the sensor field [18].

The next two components of localisation systems rely on accurate distance estimates, so if the system is to be secured, the designer must consider the possible vulnerabilities of these techniques that could be exploited by an attacker. Sections 5.2 to 5.6 investigate the security issues of these different approaches and techniques and show that distance-bounding approaches that are based on RTT and follow the four principles proposed by Clulow *et al.* [89], could achieve secure distance estimation between two nodes.

5.1.1.2 Attack on position computation

In addition to the distance estimations of at least three references, the node also needs to know the location of these references in order to estimate its position. An attack on the distance estimation can indirectly affect the position computation. However, some attacks can affect this component directly by advertising incorrect information about the compromised node's location. As illustrated in Figure 5.1, an attacker may provide incorrect location information by either pretending to be an honest reference node (Figure 5.1.a) or compromising beacon nodes (Figure 5.1.b). In both types of attack, unknown nodes which received the incorrect location information will determine their locations incorrectly.

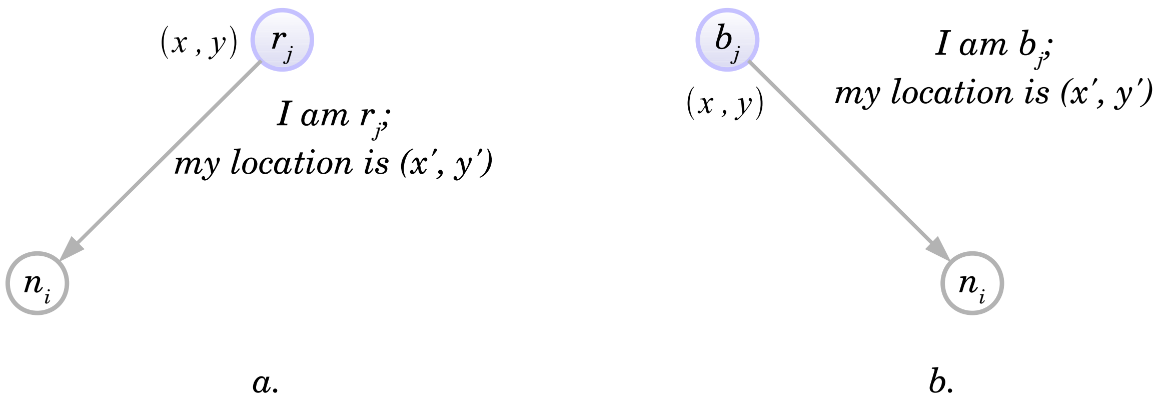


Figure 5.1. Attacks against position computation. a. Dishonest reference node, b. Compromised beacon node

The second type of attack is more difficult to achieve than the first one, since an attacker would have to be able to compromise a beacon node first to perform the second attack, while in the first type an attacker can directly advertise the incorrect location information, pretending to be an honest reference node. Several localisation algorithms, for example Nearest algorithm, do not distinguish between the reference and the beacon nodes. Therefore, both of these attacks are equivalent for them, and so an attacker can simply perform the first attack. On the other hand, other localisation algorithms distinguish between these two types of node. For example, the ALWadHA algorithm assigns the highest probability of accuracy to beacon nodes to enable the unknown nodes to select them first. The NDBL algorithm uses a different weight for beacon nodes. In addition to the other benefits of this distinction, such as achieving better accuracy, an attacker has to perform the second type of attack in order to increase the erroneousness of position computation.

5.1.2 Secure localisation algorithms

Secure localisation algorithms use several techniques to deal with the security problems of location discovery in WSNs. The main techniques used are cryptography, detecting and blocking compromised nodes, making statistical decisions or filtering the positions used in computations [18].

5.1.2.1 Cryptography

Cryptography can be used to achieve several security requirements of localisation systems, including firstly *authentication*, where nodes accept information only from authorised references, secondly *message integrity*, thirdly the *availability of the required information* to estimate a node's position, fourthly *non-repudiation*, which can be used to make sure that the references are unable to deny the location information sent by them, and fifthly, *confidentiality*, to prevent malicious nodes from collecting nodes' location information and from claiming a different legitimate location in the network.

Cryptography does not solve all the security problems of a localisation system. An attacker may disclose the key of the compromised node and forge a “location response” packet. An attacker could also replay “location response” packets intercepted in a different location. Moreover, cryptography could require extensive resources that may not be available in the resource-constrained WSNs. Therefore, additional security techniques should be used to protect localisation systems. Several localisation algorithms use non-cryptographic security techniques and rely on cryptography as a second line of defence, such as the algorithms proposed in [56, 90-94].

5.1.2.2 Detection of malicious nodes

The second technique for securing localisation systems is to detect and block malicious nodes. In this technique the node tries to detect malicious nodes and then does not use their location information to compute its position. Several methods are used to detect malicious nodes. For example, Liu *et al.* [56] compare the measured distance (using RSS, ToA, AoA, etc.) and the distance estimated by using the advertised location information. A great difference between these two distances indicates that the location information could be sent by malicious nodes. Another method is to report the location information to a central authority, which manipulates this information and filters out malicious nodes accordingly [56]. A distributed method, proposed by Srinivasan *et al.* [95, 96], enables beacons to monitor their neighbour nodes and exchange information to allow other nodes to choose trustworthy beacons based on a quorum voting approach.

Zhong *et al.* [97] propose two localisation algorithms and prove that when the number of malicious beacons is less than a specific threshold ((n - 3) / 2), where n is the total number of nodes, the localisation error is bounded, assuming that the measurement error is ideally small. In [98] Zeng *et al.* introduce a powerful attack, called “Pollution attack”, that can succeed and seriously distort the location estimation even when the number of malicious beacons is less than the lower bound and the measurement error is practically small.

5.1.2.3 Robust position computation

Implementing a robust position computation component is another technique that can be used to deal with malicious nodes. This technique enables the nodes to estimate their position with acceptable accuracy even in the presence of malicious nodes. This technique is based on using statistical and outlier filtering methods to deal with malicious nodes. Li *et al.* [73] propose the use of LMS as an improvement over the LS method for achieving robustness to attacks. LMS tolerates up to 50% outliers and still provides correct estimation. However, the original LS method outperforms LMS in the absence of attacks.

The authors of [41] investigate two types of attack-resistant location estimation techniques to tolerate the malicious attacks against range-based location discovery in WSNs. In the first technique the unknown nodes defeat malicious attacks by checking the consistency of references and then removing the inconsistent malicious references. This technique starts by using the entire set of references and then it gradually removes the most suspicious references till it gains a certain level of consistency, which depends on the measurement error of an estimated location. The authors develop an incremental MMSE approach to reduce the computation cost, but it increases the size of the required memory. The second technique is called voting-based location estimation, which quantises the deployment field into a grid of cells, and then the unknown node determines how likely it is in each cell based on each reference. After the unknown node has processed all references, it chooses the cell(s) with the highest vote and uses its (their) geometric centroid as the estimated location of the sensor node. However, specifying the voting by each reference at each cell of the grid requires a high computation cost.

Misra *et al.* [99] propose a convex optimisation-based localisation scheme that can accurately localise a node, as long as the number of neighbouring malicious beacons is lower than a critical threshold value. This scheme computes the geometric centre of the intersection of circles corresponding to location references. Since it uses a distance-bounding protocol, it assumes that the attackers can only enlarge the distance measurements. This scheme is also able to identify a significant number of malicious nodes.

5.1.2.4 Location verification

In fact, this technique can be used only to verify the location estimation results from the overall localisation system, thus it can enhance the security level of the three localisation system components. Du *et al.* [100] propose several ways to verify the estimated location. Their techniques are independent of the localisation system and they can be performed after estimating the location. The authors use deployment knowledge, with a group-based deployment model, to compare the actual observation of the deployment knowledge and the observation obtained from the estimated location. If the two observations deviate substantially from each other, the node knows that the estimated position is inconsistent with the real location and it can be rejected.

In [101] Hwang *et al.* propose a secure localisation algorithm that allows all nodes to play the role of verifier to detect the existence of malicious nodes. Each node generates a local map, checks whether its measured ranges are consistent with its ranges in the map and then finds the largest consistent subset, which could contain all the honest nodes.

Wei *et al.* [102] propose two location verification algorithms; greedy filtering by matrix

(GFM) and trustability indicator (TI). In GFM, the verification centre identifies inconsistent locations using several matrices based on neighbourhood observations and sensors' location estimations. In TI, the verification centre calculates indicators for the sensors. The sensor's location will be accepted if its final indicator is greater than a threshold.

In [103] Ekici *et al.* propose a Probabilistic Location Verification algorithm to verify the nodes' location with a small number of trusted verifiers. The verifiers use the number of hops and Euclidean distance to other nodes to determine the plausibility of the claimed location and then create an arbitrary number of trust levels in the location claimed. Finally, a central node collects these two values (i.e. the plausibility and the trust levels) from all the verifiers to decide whether it will accept or reject the claimed location.

5.2 SECURE DISTANCE ESTIMATION

Distance estimation is the first component of localisation systems, and is responsible for determining the physical distance between nodes. In hostile environments this distance estimation process must be executed using secure protocols to maintain the integrity of services that rely on this ‘next-hop’ information. There are a number of possible approaches to implementing secure distance estimation: directional antennas, RF fingerprinting, centralised systems, location-based and distance-bounding approaches [104].

Directional antennas triangulate position using AoA, which requires static references with antenna arrays providing fixed reference directions, e.g. [44]. To be secure, these nodes must also have synchronised clocks to ensure that a transmission was made to multiple references at the same time, i.e. from the same location. But time synchronisation to the accuracy required for distance estimation is a challenge in wireless networks. To determine the distance to another node would also require that node to be covered by at least two references. This limits the topology and the connection structures to a network ‘cloud’, where all nodes are covered by multiple references. For example, AoA does not provide secure distance estimation in point-to-point connections, where only two nodes are communicating with each other.

Centralised approaches work on the assumption that there are many nodes that can collaborate and aggregate data to a central system controller, which can check for suspicious node activity (for example, if two nodes receive the same transmission and it is impossible for that transmitting node to be within the communication range of both, e.g. [46, 73]). The effectiveness of these approaches relies on the network consisting of a large number of nodes capable of providing simple ancillary measurements. However, this approach requires that all data should be transmitted to a centralised authority that would be able to search connectivity graphs for wormholes or orchestrate network-wide node localisation. Such a high volume of communications might not be practical for a resource-constrained WSN.

RF fingerprinting characterises the physical communication channel aspects of a node to generate a unique identifier that is difficult to forge, e.g. [45]. This is a promising approach to identifying nodes uniquely. However, each device needs to be characterised, and if nodes are mobile and subject to varying multi-path and path-loss effects, the fingerprint might need to be revised often. This reduces the practicality of using fingerprinting in WSNs if the topology is bound to change quickly and if ad-hoc connections are often made to new nodes entering the network's area of coverage.

Location-based methods require the physical location of each node to be known at node level; for example, each node is equipped with a GPS unit, or at system level using a localisation scheme. Examples of such systems are described in [18]. Determining the location at node level requires additional network infrastructure and resources, especially indoors, where GPS is not as effective, and a system-wide localisation scheme has the same disadvantages as previously discussed for centralised approaches. Localisation schemes still rely on accurate neighbour detection, and several secure localisation proposals use distance-bounding protocols for the underlying distance estimation between nodes [29].

Distance bounding is a secure neighbour detection method that determines an upper bound for the physical distance between two communicating parties based on the RTT of cryptographic challenge-response pairs. Brands and Chaum proposed the first true distance-bounding protocol [105], and several new protocols have been proposed since. Most distance-bounding protocols have been proposed for providing security services in radio frequency identification (RFID), e.g. [106, 107], and WSN environments, e.g. [47, 108]. In the RFID environment distance bounding can be used to prove the proximity of a RFID token cryptographically to a reader, while in WSNs its ability to verify the physical proximity of "neighbour nodes" makes it a key building block in secure routing and localisation methods.

5.2.1 Distance bounding as a possible solution for ALWadHA

Components of a sensor network should ideally be designed to be as autonomous as possible, which means that a secure distance-estimation method that can be performed locally by any node, with minimal collaboration with other network entities, is preferable. Distance bounding requires a node that can make round-trip time measurements, that is, it requires a suitable RF channel and an accurate, albeit unsynchronised, clock, but can function within an ad-hoc, mobile environment with any number of nodes and network topology. Moreover, distance-bounding protocols can be executed in minimal time, while not consuming valuable resources that could be allocated to the main network services.

These characteristics make distance bounding a good approach for secure distance estimation. Using a distance-bounding method will assist ALWadHA to accomplish several design objectives such as self-organising properties, simplicity, energy efficiency, localised algorithm and security. Unlike other localisation algorithms, ALWadHA does not select a subset of references based on the estimated distance to the available references; rather it initially selects this subset of references based on their probability of accuracy. Each reference node is able to determine its probability of accuracy without relying on collaborating information from a centralised controller or other neighbours. After selecting a subset of references, the node estimates the distance only to the selected subset of references. Therefore, the use of a distance-bounding approach by ALWadHA will not add an overhead that could be caused by estimating the distance to all available references. Estimating the distance between two nodes using distance bounding involves only these two nodes. Therefore, ALWadHA can use a distance-bounding method to estimate the distance to the references within the selected subset without involving other references.

5.3 DISTANCE-BOUNDING PROTOCOLS

Distance bounding involves only two parties: a *prover* and a *verifier*, and provides the verifier with cryptographic proof of the maximum physical distance to the prover. The verifier relies exclusively on information gained from executing the protocol with the prover. As the verifier requires a reliable and secure estimate of the distance to the prover, distance-bounding protocols should be integrated into the underlying communication channel. The security of the protocol therefore depends not only on the cryptographic mechanisms, but also on how the physical attributes of the communication channel are used to measure proximity.

Time-of-flight (ToF) and RSS are often used for distance estimation between two nodes. RSS is, however, not a secure method since attackers can easily alter RSS, by either amplifying or attenuating a signal. ToF measures elapsed time for a message exchange and then estimates the distance based on the communication medium's propagation speed. Both RF and ultrasound channels (US) have been used in ToF systems. The propagation speed of sound is much slower than that of radio waves. As a result, an attacker can intercept the US communication and forward it over a faster communication medium to an accomplice closer to the verifier, or prover, thus reducing the time measurement and decreasing the distance estimate. RF channels resist this simple relay attack and are often proposed as the channel of choice for implementing distance-bounding systems. WSNs are potentially suited to implement RF ToF distance bounding. High bandwidth communication channels proposed for use in WSNs, such as ultra-wideband (UWB), are also suitable for fine resolution time-of-flight distance estimation.

There are two well-known examples of how ToF is generally used to determine distance: ToA and RTT. In ToA systems the estimate is made purely on data transmitted in one direction, from the prover to the verifier.

The distance between two devices can be calculated as

$$d = c.t_p \tag{5.1}$$

where c is the propagation speed and t_p is the one-way propagation time between the transmitter and the receiver. A distance-bounding protocol using this approach could possibly be implemented as described below if both the verifier and the prover share a common, high-precision time base, such as secure GPS receivers. The verifier sends out a challenge α at time t_0, which the prover receives at time $t_0 + t_p$. The prover then replies with a message-authenticated data packet $(t_0 + t_p, \alpha)$, where α is the challenge he received. The verifier checks the message-authentication code of this packet with the shared key and also whether $t_p \leqslant d/c$, where d is the upper bound for the distance and c is the speed of light. An attacker relaying the challenge would introduce extra propagation delay, which would be reflected in t_p, and as a result the verifier would conclude that the prover is outside the allowable distance bound. There are, however, some problems with this implementation. An accurate time base shared between devices is difficult to achieve, so this method is not practically feasible, especially in wireless network nodes. This protocol also assumes that the prover is a trusted entity, thus making the prover responsible for taking the time measurement. There are no measures in place to prevent the prover from lying about $t_0 + t_p$ and appearing closer to the verifier as a result. From a security perspective this is not acceptable, as a distance-bounding protocol should provide the verifier with reliable proof that the prover is actually within a certain distance, even in cases where the prover is not trusted.

The problem of maintaining a shared time base in both the verifier and the prover could be addressed by making distance measurements based on RTT. In RTT systems, only the verifier is required to maintain an accurate time base and the prover is also no longer responsible for providing the security-critical time measurements. The distance between the prover and verifier can be calculated as

$$d = c \,.\, (t_m - t_d)/2 \tag{5.2}$$

$$t_m = 2 \,.\, t_p + t_d \tag{5.3}$$

where c is the propagation speed, t_p is the one-way propagation time, t_m is the measured total round-trip time and t_d is the prover's processing delay between receiving a challenge and sending a response.

A distance-bounding protocol using RTT could possibly be implemented as follows: The verifier again sends a challenge α at time t_0, which the prover receives at time $t_0 + t_p$. Once the challenge has been received the prover sends a response β in the opposite direction at time $t_1 = t_0 + t_p + t_d$, which the verifier receives at time $t_2 = t_0 + 2 \,.\, t_p + t_d$. The verifier determines $t_m = t_2 - t_0$, checks that the response is correct and finally confirms that $t_p \leqslant d/c$, where d is once again the upper bound for the distance. Although this protocol is an improvement on the first example, extra care must be taken to specify the nature of the response. If the prover simply echoed the challenge received α' it would provide limited security, as any attacker who controlled a proxy token close to the verifier would be able to do this and achieve an acceptable measurement $t_p \leqslant d/c$. Ideally, the response should depend on the challenge, i.e. $\beta = f(\alpha)$, as this is also an effective measure against fraudulent provers. If this was not the case, and β was merely a random nonce agreed on beforehand between the prover and verifier, the prover could send β before receiving α and effectively decrease the round-trip time. Specifying that β is a function of α forces the prover to wait until he receives the challenge and prevents him from pre-emptively transmitting his response.

5.3.1 Distance-bounding attacks

Distance-bounding protocols should be resistant to attacks from a fraudulent prover and a malicious third party. There are three types of attack that can be prevented by distance-bounding protocols, namely distance fraud, mafia fraud and terrorist fraud.

5.3.1.1 Distance fraud

A fraudulent prover wants to convince the verifier that its physical distance is closer to the verifier than is actually the case, as shown in Figure 5.2.a. Distance-bounding protocols prevent distance fraud if the prover is not in a position to transmit its answer pre-emptively. In other words, the response is made dependent on the challenge. A possible scenario of distance fraud in WSNs is if a node that is outside the allowable network coverage area tries to initiate a connection by pretending to be within the allowable area.

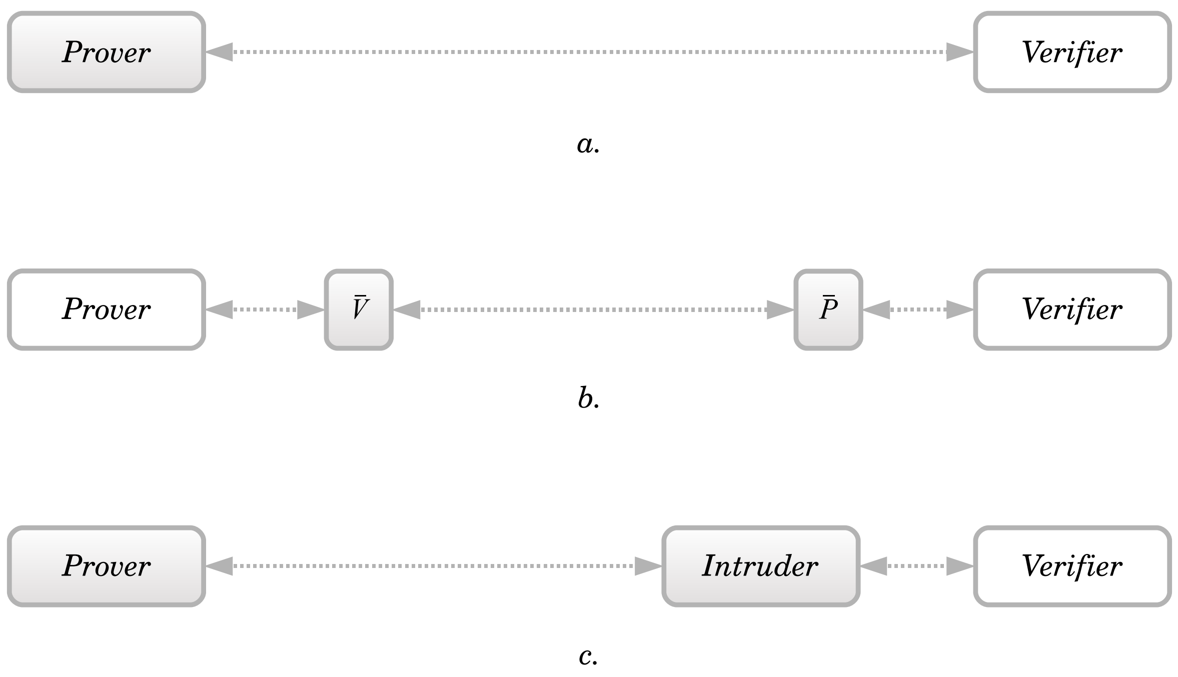


Figure 5.2. Main attacks that distance-bounding protocols aim to prevent. White boxes are the honest parties, while the grey ones are the dishonest parties. a. Distance fraud, b. Mafia fraud, c. Terrorist fraud

5.3.1.2 Mafia fraud

In this fraud, the prover (P) and the verifier (V) are honest, while a third party attacker attempts to carry out the attack. As depicted in Figure 5.2.b, the attacker consists of a dishonest prover and dishonest verifier $\{\bar{P}, \bar{V}\}$ interacting with the honest verifier and prover respectively. This fraud enables the attacker to convince V that P is in close proximity, and does not require the attacker to circumvent the application level security. The attacker does not need to know any key material; in fact he does not even need to know the content of the communication he relays. Mafia fraud was first described by Desmedt [109] but similar attacks in WSN and RFID environments have also been described as wormhole or relay attacks respectively. Mafia fraud is detected by distance-bounding protocols, as the attacker introduces additional delay into the RTT measurement, for example, even if theoretically the attacker's hardware introduces no delay, the time taken for the relayed data to propagate the extra distance will add to the RTT. The most likely scenario of a mafia-type fraud in WSNs is a wormhole attack where two nodes forward data and routing packets between remote areas of the network. This artificially short routing path changes the true topology of the network, which potentially places the attacker in a position to carry out further attacks on the network and the data transmitted over this path.

5.3.1.3 Terrorist fraud

A third-party attacker and a dishonest prover collaborate to perform terrorist fraud, as shown in Figure 5.2.c. The fraudulent prover, who is far from the honest verifier, assists an attacker to convince the verifier that the prover is in close proximity. Terrorist fraud was first described in [109]. The prover ideally does not reveal his secret information to his accomplice, who will be able to impersonate the prover if he obtains this secret information. In order to prevent terrorist fraud, some distance-bounding protocols therefore ensure that if the response is revealed in advance, the accomplice will learn the secret key of the prover.

5.3.2 Types of distance-bounding protocol

Since Brands and Chaum first described distance-bounding protocols in 1993 [105] several protocols have been published that allow a verifier to determine an upper bound on the physical distance to a specific prover. Most distance-bounding protocols have been proposed for WSN and RFID environments. These proposals can be classified by how they implement different stages of the distance-bounding process. Most of these distance-bounding protocols consist of three basic stages: the *setup* stage, where the verifier and prover prepare for the *exchange* stage, the timed exchange of challenge and response data, and the *verification* stage that ensures that the exchange step has been executed faithfully. In the literature there are three different types of distance-bounding proposals [89].

Timed authentication protocols are the simplest form of ToF-based distance bounding, with the verifier timing normal, authenticated data exchanges. The basic idea is to execute a challenge-response authentication protocol under a very tight time-out constraint, which was a concept first proposed by Desmedt [109]. For example, a verifier V transmits a random n-bit nonce $N_V \in_R \{0, 1\}^n$ to the prover P, who replies with a message-authentication code $h(S, N_V)$, where h is a keyed pseudo-random function and S is a shared secret key. Numerous protocols have been proposed using different constructions for pseudo-random functions keyed with shared secrets, public-key mechanisms or trusted third parties. Examples of such protocols are the secure neighbour detection protocol proposed by Hu *et al.* [110], and the protocol proposed by Tang *et al.* [111]. This set of protocols generally time an exchange without considering variations in the processing time of the response or the format of the challenge and response. The possible variations in the processing delay t_d affect the time measurement, which in turn causes the distance bound to be unreliable. For example, a node that usually takes 100 ms to compute a public-key signature and has a 1% (1 ms) processing time variation could cause a 333 km error in the distance estimate. Furthermore, a fraudulent prover has scope to speed up the processing time and transmit his response earlier than expected.

To reduce, or accurately predict the processing delay t_d, some protocols suggest that the prover determines the possible responses before the exchange stage. This is usually accomplished by using *pre-commitment* or *pre-computation*. Now the prover only has to choose a response β based on the challenge α received from the verifier during the exchange stage, so t_d is significantly reduced. These protocols generally propose that the function $\alpha \rightarrow \beta$ be implemented using a simple XOR or table look-up operations to minimise variation in t_d. In protocols using pre-commitment, the prover prepares possible responses during the setup stage. For example, the verifier generates a random challenge bit string, $\alpha = (\alpha_{1,}\, \alpha_{2,} \cdots, \alpha_l)$, while the prover generates a response string, $R = (R_{1,}\, R_{2,} \cdots, R_l)$. The prover commits to R, for example by transmitting a collision-resistant message authentication code $h(S, R)$. The verifier then sends one α_i after another, which the prover receives as $\alpha_i{}'$. It then instantly replies with a bit $\beta_i = \alpha_i{}' \oplus R_i$, which is calculated by XORing each received challenge bit with the corresponding bit of R. Finally the prover reveals R and authenticates α', i.e. the prover sends $MAC(\alpha')_S$ to the verifier.

In protocols using pre-computation, the prover and the verifier calculate the possible response strings before the exchange stage starts. For example, the verifier and the prover first exchange nonces N_V and N_P. Both the prover and the verifier then use a pseudo-random function F and a shared key S in order to calculate two n-bit response strings R^0 and R^1:

$$(R_1^{0,} R_2^{0,} R_3^{0,} \cdots, R_l^{0,} R_1^{1,} R_2^{1,} R_3^{1,} \cdots, , R_l^1) := F_S(N_V, N_P) \quad . \tag{5.4}$$

Since the verifier also knows R^0 and R^1 at this stage, the prover is effectively committed to two response strings, without explicitly making a commitment during setup. As a result the prover does not have to open his commitment during the verification stage, thus decreasing the data that need to be transmitted.

Although most protocols using pre-commitment or pre-computation propose the exchange of single-bit challenges and responses, there are some protocols which use a single multi-bit exchange instead, for example Third Party Location Proofs by Waters and Felten [112], the multi-bit exchange variant of Brands and Chaum proposed in [29] and the protocol by Nikov and Vauclair [113]. Communication channels usually transmit multi-bit data packets rather than single bits, so exchanging a multi-bit bit stream is easier to implement with off-the-shelf components. The problem is that these channels usually add some extra formatting information, such as trailers, headers and error-correction measures that introduce latency. This could be as simple as adding parity, start and stop bits to the data sent. An attacker may not be restricted by regular implementations and could reduce latency introduced by the communication layers to commit distance fraud or hide mafia fraud [89]. For example, an attacker could ignore packet framing and start to calculate his response before receiving the packet trailer information, and as a result be in a position to respond earlier than expected. A verifier expecting the prover to adhere to the communication channel 'rules' would therefore measure a reduced round-trip time. Even if a nonce was exchanged without any formatting, or error correction, an attacker could still gain a timing advantage. As shown in [114], an attacker can gain a timing advantage from tolerances introduced in the modulation and encoding stages, while also having the option of decreasing the decoding process of a legitimate device if the data recovery clock is derived from the transmitted data. The exchange of multi-bit data packets over normal communication layers therefore does not achieve the secure timing measurement required. In contrast, a multiple single-bit exchange provides the highest time resolution, as it depends only on the propagation time, pulse width and processing delay of a single bit. Multiple-timed bit exchanges may appear inefficient, but multiple measurements increase accuracy and confidence. An added performance benefit of transmitting single bits is that it reduces the communication overhead. This decreases the protocol execution time, especially in resource-constrained environments, such as RFID, where the capacity of the communication channel is limited.

5.3.3 Principles of secure distance bounding

To protect against attacks at the physical and packet layer of the communication channel, Clulow *et al.* [89] propose that the designer of a distance-bounding protocol should optimise the choice of communication medium and transmission format according to the following principles:

- **Principle 1:** Use a communication medium with a propagation speed which approaches the physical limit for propagating information through space-time.
- **Principle 2:** Use a communication format in which the recipient can instantly react to the reception of each individual bit. This excludes most traditional byte or block-based communication formats, and in particular any form of redundancy such as error-correction and packet delimiters such as headers and trailers.
- **Principle 3:** Minimise the length of the symbol used to represent each single bit.
- **Principle 4:** The distance-bounding protocol should be designed to cope with bit errors during the rapid single-bit exchange, because the previous principle may limit the reliability of the communication channel.

Principle 4 is of particular importance in WSNs, as it ensures that the system is more resilient as a whole and that ad-hoc connections can quickly and reliably be formed even if some communication errors occur. A rapid bit exchange channel, with minimal latency needed for accurate distance bounding, has no error correction and might be implemented on resource-constrained devices that contain simple radio receivers more susceptible to noise, so that bit errors could occur. It is therefore an advantage if a distance-bounding protocol is resistant to bit errors during the exchange stage, rather than incurring the time cost of rerunning the entire protocol. Hancke and Kuhn (HK) [106] propose that protocols handle errors by defining a bit-error threshold, and that the protocol therefore successfully completes even if some responses are incorrect. Although this method is easier to apply to pre-computation protocols with no verification stage, these authors also point out that further protocols can tolerate bit errors during the exchange stage using a threshold method, as long as the challenge bits, $\alpha_i{}'$, received by the prover and the response bits, β_i, sent by the prover are transmitted over an error-corrected channel during the verification stage.

An alternative to pre-commitment protocols is to use error-correcting codes (ECC), as proposed by Singelée and Preneel (SP) [115]. $ECC(l + k, l)$ is applied to the generated strings response string R and this results in a new string. The protocol now continues exactly as before except that there are now $l + k$ timed bit exchanges instead of l. After the bit exchanges, the verifier reconstructs R from the challenges it sent and the responses it received. Finally, the verifier uses the ECC to correct any bit errors in the reconstructed R. The verification stage then proceeds as normal. The disadvantage of using ECC for error resistance is that it increases the protocol execution time. This is a result of the node performing additional ECC calculations and also transmitting additional redundant bits, which are used to detect and correct bit errors in the challenge and response strings. Recently, Munilla and Peinado [116] proposed two effective attacks against the SP protocol. These two attacks increase the adversary's success probability. Moreover, they showed that when the number of allowed failures increases, the false acceptance probability of the SP protocol can be higher than that of the HK protocol.

5.4 COMPARISON FRAMEWORK

The implementation of distance-bounding protocols can differ in a number of ways. For example, not all protocols are designed to be resistant to distance, mafia and terrorist fraud, while some protocols might be optimised to reduce communication or intended for use in systems with resource-constrained nodes. As a result, the security, resource requirements and execution time vary for each protocol. Distance-bounding protocols should not adversely affect the quality of service and should execute in minimal time without consuming resources that could be allocated to main network applications. This section defines the metrics of a framework which will be used to compare selected protocols. The framework consists of the following metrics:

- **Attacks prevented:** Not all protocols that estimate the distance between nodes are necessarily designed to be secure, e.g. [117]. This investigation considers only distance-bounding protocols that prevent one or more of the attacks described in Section 5.3.1, and it will show which attack(s) they prevent.

- **Attack success probability:** Security can be quantified by the probability that an attacker may succeed in executing a chosen attack, which is obviously considered to be a critical parameter for choosing a distance-bounding protocol. For each protocol, the probable success of the attacker in committing the attack(s) that these protocols are meant to prevent will be shown.

- **Cryptographic primitives required:** In order to run a distance-bounding protocol, a selection of cryptographic primitives are needed, such as random number generators, hash/pseudo-random functions, symmetric cryptography and/or asymmetric cryptography. Cryptographic primitives affect not only security but also the resources required to implement and execute the protocol in practice. Therefore, it is important to consider the types of primitives needed to implement a protocol and also how many times during each protocol run this primitive is used in a computation.

- **Memory:** This represents the amount of memory space required to store the functional variables required by each distance-bounding protocol during execution. For example, the protocol needs to store the challenges, responses, nonces, keys, and the like. In order to compare the memory overhead required by each protocol, the following length variables are assigned: K_{SYM} for the symmetric cryptography keys, K_{PUB} for asymmetric cryptography keys, F for a hash or pseudo-random function, MAC for MAC authentication (this indicates the cost of having an additional MAC key, security best practice, or storing the intermediate result from the symmetric encryption function) and N_{RAND} for the random number. For instance, if a protocol uses symmetric cryptography, for which it needs to store a shared symmetric key, and also requires one hash function and two random numbers, the memory needed is: $m = K_{SYM} + F + 2 * N_{RAND}$.

- **Error resilience:** Resilience against channel errors is important for distance-bounding protocols' robustness, especially for those using the rapid bit exchange phase, since they are typically sensitive to channel errors. Some of the distance-bounding protocols that will be discussed in this chapter are designed to be tolerant to faults occurring during transmission, and so they can be used in noisy environments. Others are able to handle channel errors if the protocol is modified.

- **Required computation:** Efficient computation should be taken into consideration in order to implement a distance-bounding protocol that executes within a specified time, especially if the protocol will be required to run using limited resources. The computational efficiency of each distance-bounding protocol is compared, based on the number of variables or values that need to be computed during each protocol run. The values considered are hash or pseudo-random function results, a symmetric encryption operation, MAC computation and random number generation. This is a slightly crude metric, as the exact times of executing these functions may vary, especially considering the different algorithm options for each computation. However, in the case of distance-bounding protocols where the number of exchanges is not considerably larger than the input block sizes of the primitives, it is feasible that execution times could be comparable, e.g. a symmetric encryption, a MAC or a hash of a short string could be equivalent.

- **Data transmission:** The execution time of the protocols is compared with regard to the amount of data that needs to be transmitted. The speed of the communication link could influence the overall execution time of the protocol, and for nodes that are required to facilitate quick route discovery and connection setup, the communication volume and protocol execution time need to be minimised. It is assumed that each protocol uses the same communication channel, with the total communication time T needed to execute the protocol being equal to the total number of exchanged bits multiplied by the transmission time period (P_B) of each bit, or line coding symbol representing a bit.

5.5 SELECTED DISTANCE-BOUNDING PROTOCOLS

It is not feasible to discuss and compare all existing distance-bounding and secure-ranging proposals in the literature. Therefore, the comparison is limited to protocols adhering to the 'principles of secure distance bounding' as defined in Section 5.3.3. Please note that Principle 2 implies that all these methods would require a special bit-exchange channel for maximum security. However, all these protocols can be converted to a single multi-bit exchange if a special channel is not available. In such a case the effects of the physical communication layer attacks demonstrated in [89, 114] should be taken into account. Examples of such a security analysis for a UWB and IEEE 802.15.4 radio channel can be found in [118, 119]. Please note that the investigated distance-bounding protocols were originally developed for different application scenarios. For example, protocols designed for RFID might place more emphasis on being suitable for devices with resource constraints, while protocols designed for nodes with more resources, which might be found in some WSNs, might rather concentrate on adding functionality or increased resistance to attacks. All protocols discussed here can be applied to WSNs for secure distance estimation.

5.5.1 Brands and Chaum's distance-bounding protocol

Brands and Chaum (BC) [105] presented the first distance-bounding protocol based on measuring the RTT for a single-bit challenge-response exchange. As shown in Figure 5.3, the prover commits to a new random *n*-bit string *R* (e.g. transmits hash value $h(S, R)$). The verifier generates a random *n*-bit string α . In the challenge-response exchange phase the verifier sends one challenge bit α_i to the prover, who replies with the bit $\beta_i = \alpha_i \oplus R_i$. The verifier measures the RTT between sending the bit α_i and receiving the bit β_i . In the end the prover sends *R* in addition to the signed bit strings to the verifier, who checks the correctness of $\beta_i = \alpha_i \oplus R_i$.

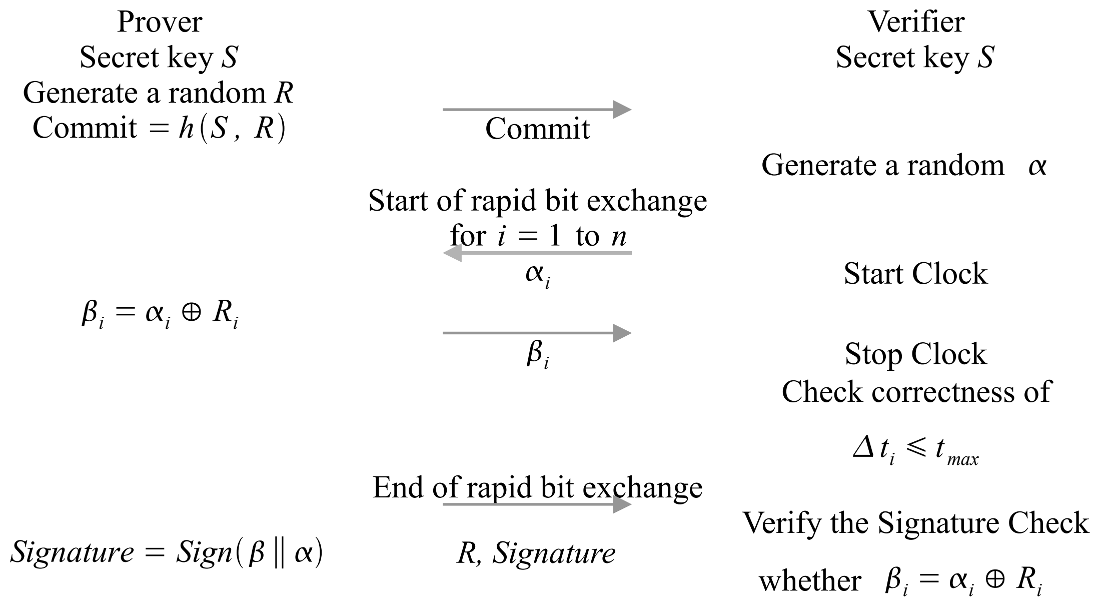

Prover
Secret key S
Generate a random R
Commit $= h(S, R)$
Verifier
Secret key S
Commit
Generate a random α
Start of rapid bit exchange
for $i = 1$ to n
α_i
Start Clock
$\beta_i = \alpha_i \oplus R_i$
β_i
Stop Clock
Check correctness of
$\Delta t_i \leqslant t_{max}$
End of rapid bit exchange
Verify the Signature Check
$Signature = Sign(\beta \| \alpha)$
R, $Signature$
whether $\beta_i = \alpha_i \oplus R_i$

Figure 5.3. Brands and Chaum's Protocol

5.5.2 Bussard and Bagga's distance-bounding protocol

Bussard and Bagga (BB) [120] proposed a protocol called Distance-Bounding Proof of Knowledge (DBPoK) protocol, which combined the original distance-bounding proposal [105] and zero-knowledge proof of knowledge protocols [121]. The DBPoK protocol relies on public key cryptography with a certificate generated by a trusted authority.

The first stage of the DBPoK protocol is called the bit commitment stage. The prover generates a random key K to encrypt its private key S, which results in $e = \varepsilon_K(S)$, and then the prover commits to both K and e. During the bit-exchange stage, the prover responds with either K_i or e_i, depending on the challenge bit received from the verifier. The prover opens the commitments, which are checked by the verifier, on the released bits of K and e in the commitment opening stage. During the proof of knowledge stage, the prover convinces the verifier in a zero-knowledge interaction that he performed the first three stages.

5.5.3 Čapkun et al.'s distance-bounding protocol

Čapkun *et al.* (CBH) [108] proposed a protocol for mutual authentication with distance bounding (MAD). The protocol basically modified the BC protocol to allow for mutual distance bounding. The verifier and the prover generate two random numbers each (r, r' and s, s' respectively) and exchange the commitment of these numbers. In the fast challenge-response phase the verifier starts by sending $\alpha_1 = r_1$, to which the prover replies with $\beta_1 = s_1 \oplus \alpha_1$. The exchange continues with the verifier sending $\alpha_i = r_i \oplus \beta_{i-1}$ and the prover replying with $\beta_i = s_i \oplus \alpha_i$. During the exchange stage both parties measure the time taken for the response to arrive after sending a challenge. During the authentication phase, the two parties both open commitments (r' and s') and transmit an MAC message containing the response-challenge strings, which both can then verify.

5.5.4 Hancke and Kuhn's distance-bounding protocol

Hancke and Kuhn (HK) [106] proposed a pre-computation protocol that did not require additional data to be transmitted during the verification stage. The protocol, as shown in Figure 5.4, requires the verifier V and prover P to share a common secret key S. During the setup stage V and P exchange random nonces N_V and N_P, and then they compute two n-bit sequences, R^0 and R^1, using a pseudo-random function h of the key and the concatenation of the nonces N_V and N_P. During the single-bit challenge-response exchange V generates and sends an unpredictable random challenge bit α_i , to which P responds instantly with a bit from either R^0 or R^1 based on the value of α_i . During the verification stage the verifier checks that the responses received match the possible responses it calculated during the setup stage. HK protocol is designed to tolerate transmission errors during the exchange stage and a verifier will accept a prover if only k out of n bits are correct.

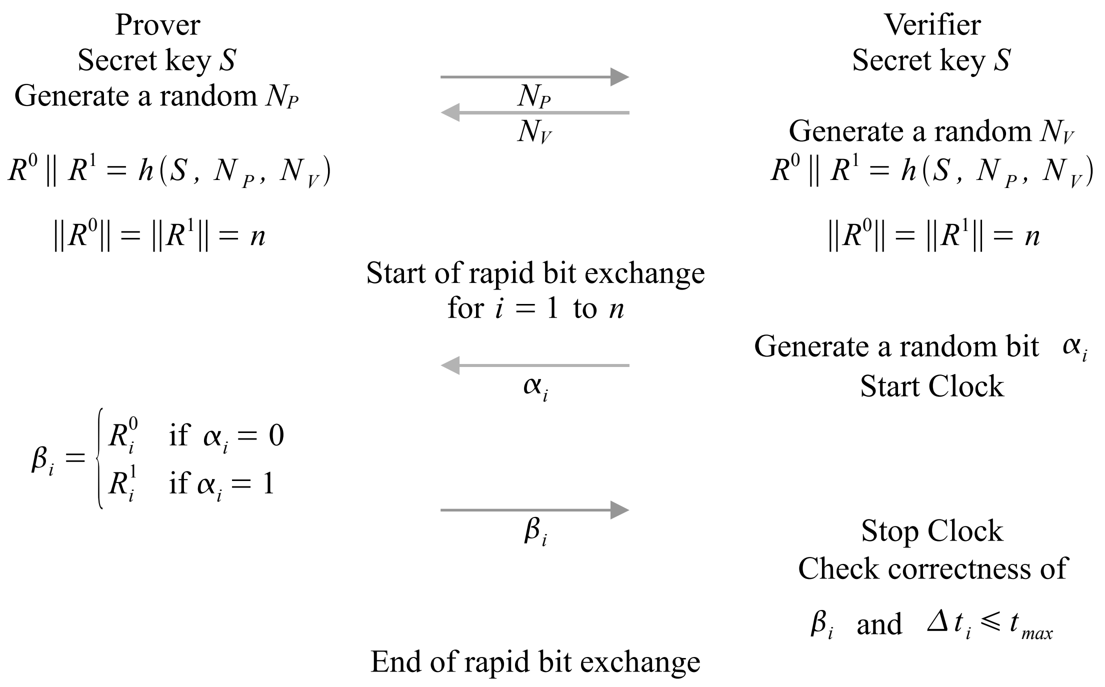

Prover
Secret key S
Generate a random N_P
$R^0 \| R^1 = h(S, N_P, N_V)$
$\|R^0\| = \|R^1\| = n$
N_P
N_V
Verifier
Secret key S
Generate a random N_V
$R^0 \| R^1 = h(S, N_P, N_V)$
$\|R^0\| = \|R^1\| = n$
Start of rapid bit exchange
for $i = 1$ to n
Generate a random bit α_i
Start Clock
α_i
$\beta_i = \begin{cases} R_i^0 & \text{if } \alpha_i = 0 \\ R_i^1 & \text{if } \alpha_i = 1 \end{cases}$
β_i
Stop Clock
Check correctness of
β_i and $\Delta t_i \leq t_{max}$
End of rapid bit exchange

Figure 5.4. Hancke and Kuhn's protocol

5.5.5 Reid et al.'s distance-bounding protocol

Reid *et al.* (RNTS) [122] modified HK protocol [106] to be resistant to terrorist fraud by applying the terrorist fraud deterrent method proposed by BB protocol [120]. A verifier and a prover exchange their identities and nonces and then they use a pseudo-random function h to derive a session key k, which is XORed to a shared key S to get the ciphertext c. In the fast challenge-response phase the prover responds to the verifier with either c_i or k_i, depending on the value of α_i . The prover cannot reveal c_i and k_i to an adversary, as the adversary can then compute S and then impersonate the prover in more than a single run of the protocol.

5.5.6 Tu and Piramuthu's distance-bounding protocol

Tu and Piramuthu's (TP) proposed protocol [123] uses the same principles as were used in [105, 106, 122]. Their main contribution is dividing the fast challenge-response phase into four loop iterations using different hash outputs; they reduce the success probability of an adversary to $(9/16)^n$. However, Kim *et al.* [124] showed that this protocol is not secure against an active adversary; a mafia fraud attacker which could change the challenge bits and relay multiple protocol runs would recover the secret shared key.

5.5.7 Munilla and Peinado's distance-bounding protocol

Munilla and Peinado (MP) [107] modified HK protocol in order to reduce the adversary's mafia fraud success probability. They used an additional void challenge in addition to the two existing bit values that can be transmitted. A void challenge is basically a period in which the verifier does not send any challenge bit. If a mafia fraud attacker tries to query the prover pre-emptively to read out possible response bits, he might send a challenge in a time slot in which the prover knows it should not expect a challenge, and therefore the attack will be detected. In order to use the void challenge, the protocol requires that the verifier and the prover have crude synchronisation, although it has been shown that the protocol can be modified to eliminate this requirement.

5.5.8 Kim and Avoine's distance-bounding protocol

Kim and Avoine (KA) [125] modified the previous protocol [107] by using mixed challenges. The challenges are divided into two categories: random challenges and predefined challenges. The first method uses random bits generated by a verifier and the second uses predefined bits known to the verifier and the prover in advance. After exchanging nonces, the verifier and the prover compute a $4n$-bit sequence $T \| D \| R^0 \| R^1$. The string T indicates which challenges a verifier sends; string D is the predefined challenges and strings R^0 and R^1 are the responses used by a prover. The random challenges, which are unknown to the prover, ensure that the protocol is still secure against distance fraud, as the prover would not be able to respond pre-emptively to these challenges. The predefined challenges, of which the prover knows the value beforehand, allows the prover to detect whether an attacker is transmitting the query, i.e. if the challenge received is different from what is expected, it did not originate from the verifier.

5.5.9 Kim et al.'s distance-bounding protocol

Kim *et al.* (KAKSP) [124] proposed a distance-bounding protocol based on the authentication protocols MAP1 [126] and MAP1.1 [127]. In order to achieve distance bounding, they added a timed challenge-response phase to the MAP1.1 protocol and then they modified it to ensure the privacy of the prover. The first two phases of this protocol are, however, very similar to the first two phases of the RNTS protocol [122]. In the verification stage the prover and the verifier authenticate each other by exchanging messages consisting of a keyed pseudo-random function of the exchanged bits. The prover also sends the challenges it received and responses sent in plaintext. This allows the verifier to incorporate error resistance into the protocol run using the threshold method.

The verifier must perform an exhaustive search over its database that grows linearly with the number of nodes deployed in order to find the prover's identity and key, which should be used to verify the prover's authentication message. Using this protocol in a system with many nodes may therefore not be efficient. Peris-Lopez *et al.* [128] have shown that an attacker who observes multiple protocol runs can recover the prover's secret key.

5.5.10 Meadows et al.'s distance-bounding protocol

The protocol proposed by Meadows *et al.* (MSC) [47] can be seen as a hybrid of a pre-computation and pre-commitment protocol. During the setup phase the verifier requests that the protocol must commence and sends a nonce to the prover. The prover calculates a response string *R* by hashing its ID and a random nonce N^P it generates. The verifier sends a random challenge, which the prover XORs to *R* and then transmits back. Finally, the prover sends an authentication message to the verifier, including the random nonce used to calculate *R*. In this protocol variation only the prover calculates the response string and there is no explicit commitment before the exchange stage. The prover is prevented from pre-emptively responding with $R' \oplus \alpha$, as it will be impossible to find a value for N^P so that $h(N^P, P) = R' \oplus \alpha$ before the verification stage. The advantage of this protocol is that it involves minimal data transmission and does not require the nodes to share a key before the verification stages, which means the verifier can make initial estimates of the distances to other nodes and determine its neighbours before keys are exchanged.

5.5.11 Avoine *et al.*'s technique

Avoine *et al.* [129] presented a generic technique called MULtiState Enhancement

(MUSE) that can be used by the existing distance-bounding protocols. MUSE extended the void challenges concept, which was introduced in MP protocol [107], to *p*-symbols, where $p \geqslant 2$. In other words, instead of using binary messages {0, 1}, MUSE uses *p*-state messages {0, 1, 2, …, *p*-1}. The use of *p*-state messages decreases the adversary's success probability without changing the number of exchange messages. Therefore, MUSE could improve the performance of distance-bounding protocols that use this technique. However, using the MUSE technique requires more memory, exchanges a higher number of bits, and in practice it could be difficult to implement the multistate messages required by this technique. In the comparison, the MUSE-4 HK will be considered, which is an improvement over the four-state message approach of HP protocol.

5.5.12 Avoine and Tchamkerten's distance-bounding protocol

Protocols belonging to the HK family do not distinguish authentication from proximity check. Avoine and Tchamkerten (AT) introduced a tree-based RFID distance-bounding protocol [130], which distinguishes authentication from proximity check. The tree-based RFID protocol performs the authentication before the fast phase, using the first *m* bits of the keyed-hash value. The prover and the verifier use the remaining 2^{n+1} - 2 bits of the keyed-hash value to label a full binary tree of depth *n*. The left and the right edges are labelled 0 and 1, respectively. Each node in the tree (except the root) is associated with a value of a particular bit from the 2^{n+1} - 2 bits. The edge and the node values represent the verifier's challenges and the prover's replies, respectively.

Compared with HK protocol, this protocol reduces the probability of a successful attack to $(1/2)^n(n/2 + 1)$, but it requires more memory space. However, this protocol can use multiple trees to balance the false-acceptance rate (FAR) and memory requirement. Using multiple trees requires storing $\alpha(2^{k+1} - 2)$ bits for the fast phase where α is the number of the trees, *k* is the depth of the trees and $n = \alpha k$. This memory is much less than the one required for using a single tree $2^{n+1} - 2$. With a multiple tree FAR is equal to $(2^{-k}(k/2 + 1))^{\alpha}$. The required memory shown in Table 5.1 and Figure 5.8 considers using eight trees with depth 8.

5.5.13 Trujillo-Rasua et al.'s distance-bounding protocol

The authors of [131] introduced a graph-based distance-bounding protocol (TMA). The graph consists of $2n$ nodes $\{q_0, q_1, \cdots, q_{2n-1}\}$ and $4n$ edges. Both the verifier and the prover start at node q_0. The next node will be specified based on the challenge bit and the corresponding edge. The prover and the verifier either move to the next node (q_{i+1}) or to the node after two steps (q_{i+2}). However, TMA protocol can be simply represented without involving any graph, as shown in Figure 5.5; also l can be eliminated, which is the complementary of s.

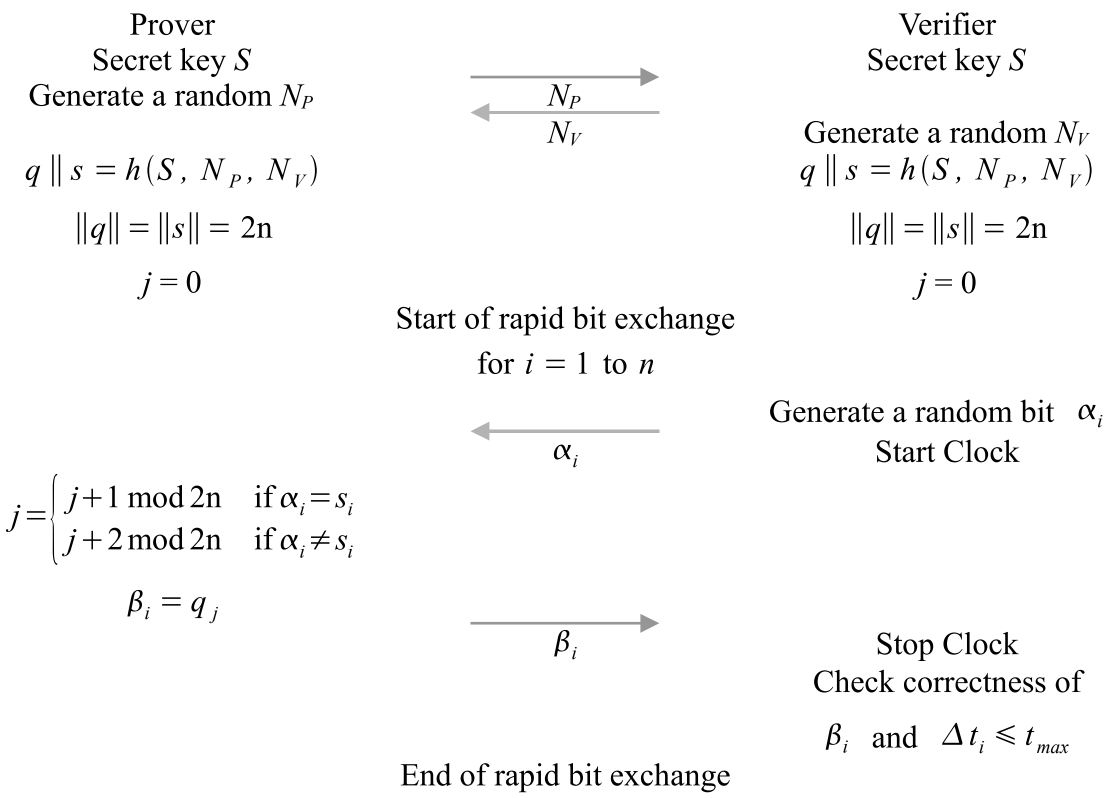

Prover
Secret key S
Generate a random N_P
N_P
N_V
Verifier
Secret key S
Generate a random N_V
$q \| s = h(S, N_P, N_V)$
$\|q\| = \|s\| = 2n$
$j = 0$
$q \| s = h(S, N_P, N_V)$
$\|q\| = \|s\| = 2n$
$j = 0$
Start of rapid bit exchange
for $i = 1$ to n
Generate a random bit α_i
Start Clock
α_i
$j = \begin{cases} j+1 \bmod 2n & \text{if } \alpha_i = s_i \\ j+2 \bmod 2n & \text{if } \alpha_i \neq s_i \end{cases}$
$\beta_i = q_j$
β_i
Stop Clock
Check correctness of
β_i and $\Delta t_i \leqslant t_{max}$
End of rapid bit exchange

Figure 5.5. Trujillo-Rasua *et al.*'s protocol

5.5.14 Peris-Lopez et al.'s distance-bounding protocol

Peris-Lopez *et al.* [128] presented a passive attack against KAKSP protocol, which is also exploitable against RNTS and TP protocols. This attack allows an adversary to discover the long-term secret key of the prover owing to the weak protection mechanism adopted against terrorist fraud attacks. Therefore, this attack wrecks all the security properties claimed by these protocols. Peris-Lopez *et al.* introduced design guidelines to propose a secure and efficient distance-bounding protocol. These guidelines were used to design a new protocol called Hitomi. Hitomi protocol was inspired by KAKSP protocol.

In KAKSP protocol, knowing Z^0 and Z^1 makes it easy to calculate the value of secret key x, since $x = Z^0 \oplus Z^1$. In contrast, in Hitomi protocol there is a non-linear relation between Z^0 and Z^1; moreover, no information on the secret key x is revealed through the responses bits. However, Hitomi protocol requires more memory space and bits transmission than does KAKSP protocol.

5.6 COMPARISON OF DISTANCE-BOUNDING PROTOCOLS

This section presents a comparison of the distance-bounding protocols discussed in Section 5.5. The comparison is based on the proposed framework defined in Section 5.4, with the following metrics: attacks prevented, success probability of an adversary to perform distance fraud, mafia fraud or terrorist fraud, the protocol error resilience, requirement of special channel, protocol efficiency based on the primitives used by the prover, the amount of memory space required by the prover, the computation required by the prover and the amount of data exchanged between the two parties. For the sake of comparing the different protocols, the resource variables were assigned the following values: K_{SYM} = 128 bits, K_{PUB} = 3072 bits, F = 256 bits, MAC = 128 and N_{RAND}= 64 bits. These values were chosen based on the cryptographic functions that are most likely to be used, taking into account the minimal recommendations for choosing secure algorithms up to the year 2030 by the National Institute of Standards and Technology [132]. For K_{SYM} and MAC AES128 is used, and to achieve equivalent 128-bit security, F is generated using SHA-256 and the public key K_{PUB} is an RSA key of length 3072. If K_{PUB} was implemented using elliptic curve cryptography the key length would be 256 - 383 bits. To allow for equal comparison of all protocols, n needs to be smaller than $F/3 \approx 85$ (Munilla's protocol needs to split F into three equal length registers). The number of iterations in the exchange phase is therefore chosen to be $n = 64$ (simply the closest number smaller than 85 that is a power of 2).

Kara *et al.* [133] show that the success probability of an adversary to perform distance fraud against HK protocol is $(3/4)^n$. Therefore, in Table 5.1, this value will be considered for all protocols that have a lookup from registers, such as HK, RNTS and MP protocols. A summary of the comparison is shown in Table 5.1. If required, the reader can generate a similar table tailored to his system design by assigning his system's values and substituting these values into the equations given in Table 5.2.

Table 5.1. Comparison of selected distance-bounding protocols

Protocol	Attacks			SPA			ER	Mem bits	Computation			TD
	DF	MF	TF	**DF**	**MF**	**TF**			**RN**	**PRF**	**Asm**	**bits**
BC	√	√		$(1/2)^n$	$(1/2)^n$		½	704	1	2		576
BB	√	√	√	$(1/2)^n$	$(1/2)^n$	$(1/2)^n$	½	6912	3	4	1	3392
CBH	√	√		$(1/2)^n$	$(1/2)^n$		½	1152	2	4		1024
HK	√	√		$(3/4)^n$	$(3/4)^n$		√	258	1	1		256
RNTS	√	√	√	$(3/4)^n$	$(3/4)^n$	$(3/4)^n$	√	384	1	1		256
TP	√	√	√	$(1/2)^n$	$(9/16)^n$	$(9/16)^n$	½	640	1	5		1280
MP	√	√		$(3/4)^n$	$(5/8)^n$		√	641	1	2		512
KA	√	√		$(7/8)^n$	$\sim(1/2)^n$		√	260	1	1		256
KAKSP	√	√	√	$(1/2)^n$	$(1/2)^n$	$(3/4)^n$	√	960	1	3		832
MSC	√	√		$(1/2)^n$	$(1/2)^n$		½	320	1	2		384
MUSE	√	√		$(1/4)^n$	$(7/16)^n$		√	768	1	1		384
AT	√	√		$(1/2)^n$	$(1/2)^n * (n/2+1)$		√	4336	1	1		512
TMA	√	√					√	512	1	1		256
Hitomi	√	√	√	$(1/2)^n$	$(1/2)^n$	$(3/4)^n$	√	1152	3	4		1024

Legend

DF	Distance fraud
MF	Mafia fraud
TF	Terrorist fraud
SPA	Success probability of an adversary
n	Number of iteration
ER	Error resilience
Mem	Memory
RN	Random number
PRF	Pseudo-random function
Asm	Asymmetric cryptography
TD	Transmitted data
½	Possible with modifications, added overhead

Table 5.2 summarises the amount of memory required and the transmitted data exchanged between the prover and verifier for each protocol based on the cryptographic primitives defined in the framework.

Table 5.2. Memory and transmitted data based on the defined primitives

Protocol	Memory	Transmitted data
BC	$K_{SYM} + F + 3n + MAC$	$F + 3n + MAC$
BB	$2\ K_{PUB} + K_{SYM} + 6n + F$	$5n + K_{PUB}$
CBH	$K_{SYM} + 2F + 4n + 2MAC$	$2F + 4n + 2MAC$
HK	$K_{SYM} + 2N_{RAND} + 2$	$2N_{RAND} + 2n$
RNTS	$K_{SYM} + 2N_{RAND} + 2n$	$2N_{RAND} + 2n$
TP	$K_{SYM} + 2N_{RAND} + 2n + F$	$2N_{RAND} + 2n + 4F$
MP	$K_{SYM} + 2N_{RAND} + 2n + 1 + F$	$2N_{RAND} + 2n + F$
KA	$K_{SYM} + 2N_{RAND} + 4$	$2N_{RAND} + 2n$
KAKSP	$K_{SYM} + 2N_{RAND} + 3n + 2F$	$2N_{RAND} + 3n + 2F$
MSC	$K_{SYM} + 2N_{RAND} + n$	$2n + MAC + 2N_{RAND}$
MUSE	$K_{SYM} + 2N_{RAND} + nplog_2\ (p)$	$2N_{RAND} + 2nlog_2\ (p)$
AT	$K_{SYM} + 2N_{RAND} + 2^{n+1} - 2$	$2N_{RAND} + F + 2n$
TMA	$K_{SYM} + 2N_{RAND} + 4n$	$2N_{RAND} + 2n$
Hitomi	$K_{SYM} + 4N_{RAND} + 4n + 2F$	$4N_{RAND} + 4n + 2F$

Legend	
K_{SYM}	Symmetric key
K_{PUB}	Asymmetric key
N_{RAND}	Random number
MAC	Message Authentication Code
F	Pseudo-random function
n	Number of iteration
p	Number of state

5.6.1 Security

In order to compare the security of these distance-bounding protocols, given a fixed number of challenge-response exchanges, the comparative success probability of an adversary to perform distance, mafia and terrorist fraud with number of iterations $n = 64$ is computed. The results are shown in Table 5.3. The probability of an attack succeeding decreases as the number of iterations increases. Increasing the number of iterations does, however, involve a trade-off with both memory used and the amount of data transmitted increasing with the number of iterations, as shown in Figure 5.6 and 5.7.

Table 5.3. The success probability of an adversary (SPA)

Protocol	SPA		
	DF	MF	TF
BC	5.42×10^{-20}	5.42×10^{-20}	
BB	5.42×10^{-20}	5.42×10^{-20}	5.42×10^{-20}
CBH	5.42×10^{-20}	5.42×10^{-20}	
HK	1.01×10^{-08}	1.01×10^{-08}	
RNTS	1.01×10^{-08}	1.01×10^{-08}	1.01×10^{-08}
TP	5.42×10^{-20}	1.02×10^{-16}	1.02×10^{-16}
MP	1.01×10^{-08}	8.64×10^{-14}	
KA	1.94×10^{-04}	5.42×10^{-20}	
KAKSP	5.42×10^{-20}	5.42×10^{-20}	1.01×10^{-08}
MSC	5.42×10^{-20}	5.42×10^{-20}	
MUSE	2.94×10^{-39}	1.05×10^{-23}	
AT	5.42×10^{-20}	1.79×10^{-18}	
Hitomi	5.42×10^{-20}	5.42×10^{-20}	

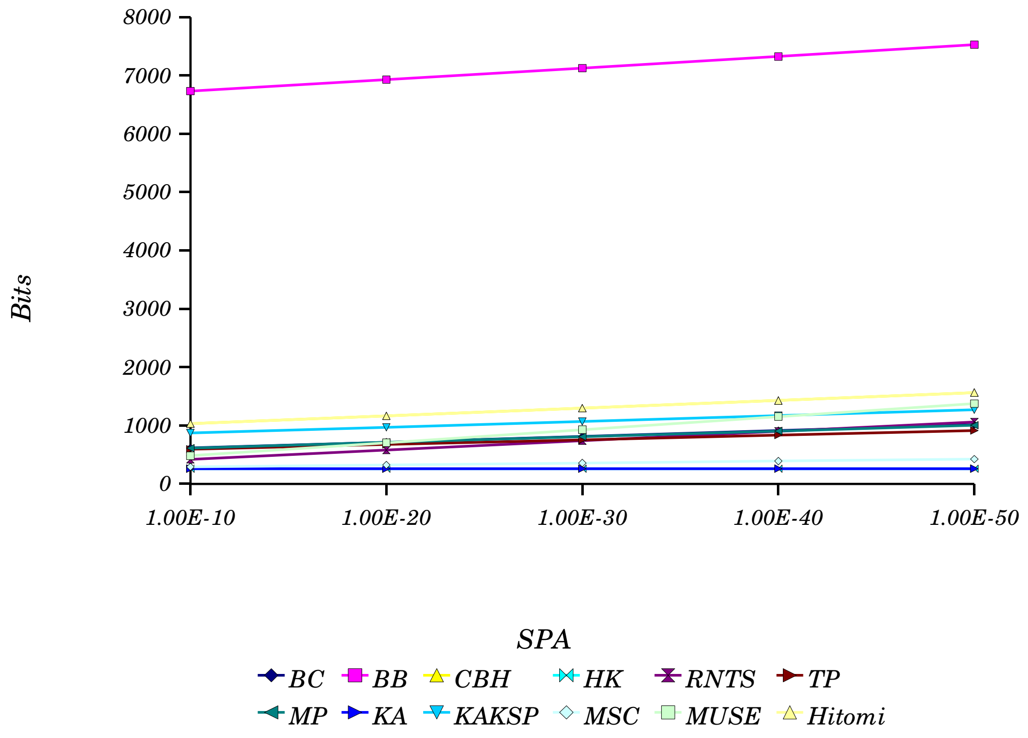


Figure 5.6. Memory vs probability of attack for mafia fraud (SPA)

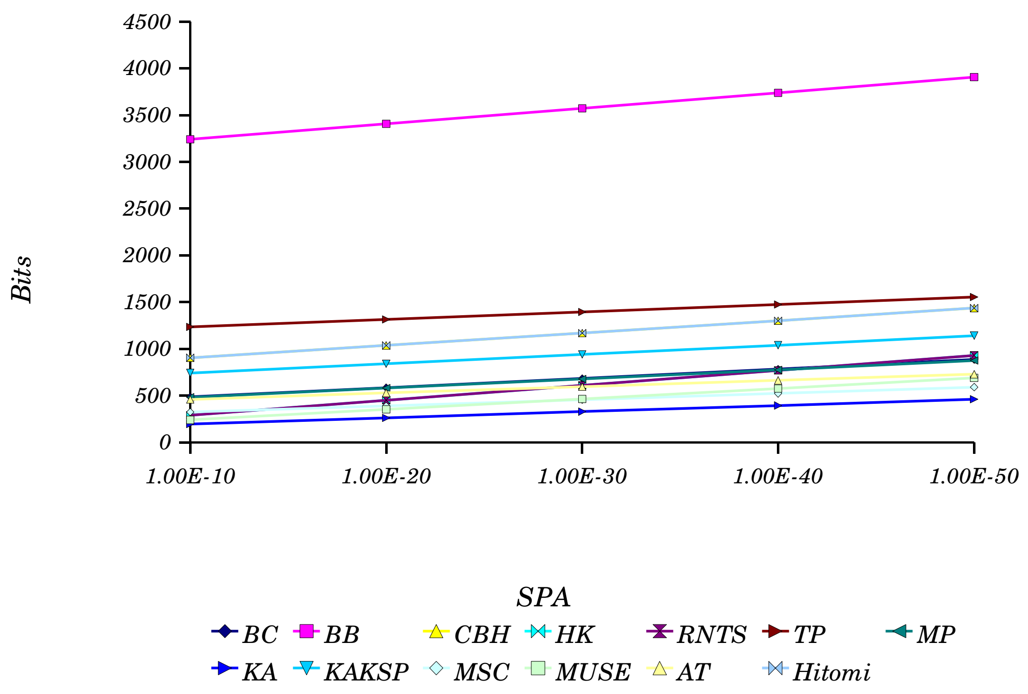


Figure 5.7. Transmitted data vs success probability of attack for mafia fraud (SPA)

5.6.1.1 Current vs k-previous challenge dependent

Recently, Kara *et al.* [133] classified distance-bounding protocols having bitwise fast phases and no final signature into two types: Firstly, *current challenge dependent* (CCD), where each response bit depends only on the current challenge, and secondly, *k-previous challenge dependent* (*k*-PCD), where each response bit depends on the current and *k*-previous challenges. Kara *et al.* showed the theoretical security bounds for these two types. For CCD protocols, they showed that there is a trade-off between mafia fraud and distance fraud, namely $P_{maf} + P_{dis} \geqslant 3/2$, where P_{maf} and P_{dis} are the success probability for mafia fraud and distance fraud respectively. Also, they proved that there is a security limit concerning mafia fraud such that $P_{maf} \geqslant 3/4$. Therefore, if the security level for distance fraud is ideal (i.e. $P_{dis} = 1/2$), then the protocol is completely vulnerable to mafia fraud (i.e. $P_{maf} = 1$).

To improve the security level of these protocols, Kara *et al.* suggested extending these protocols to become *k*-PCD protocols. In these protocols, $P_{maf} + P_{dis} \geqslant 5/4$, while $P_{maf} \geqslant 5/8$. As a case study, they illustrated two natural extensions on HK protocol among 1-PCD protocols. From the first extension, they achieved $P_{dis} \geqslant 1/2$ and

$P_{maf} \geqslant 3/4$. For the second one, they achieved $P_{dis} \geqslant 5/8$ and $P_{maf} \geqslant 5/8$.

The authors conjectured that the attacks used in the analysis are the best generic attacks mounted on CCD and k-PCD protocols. However, this could be not the case, and so finding other ways to implement these attacks to produce different trade-off curves can be regarded as an open problem.

5.6.2 Memory and transmitted data

In order to compare the memory requirements and data transmission of the selected distance-bounding protocols, the comparative amount of memory required by the prover to run each protocol, and the number of bits exchanged between the prover and the verifier are computed. For the purpose of comparison the following values are used: K_{SYM} = 128 bits, K_{PUB} = 3072 bits, F = 256 bits, MAC = 128 and N_{RAND} = 64 bits and the number of iterations is n = 64. The results are shown in Figure 5.8. The total transmission time can be calculated by multiplying the number of transmitted bits with the channel bit period P_B. Figure 5.9 shows the effect of increasing the number of iterations on the data transmitted between the prover and the verifier. Figure 5.10 shows the effect of increasing the number of iterations n on the memory required by each protocol.

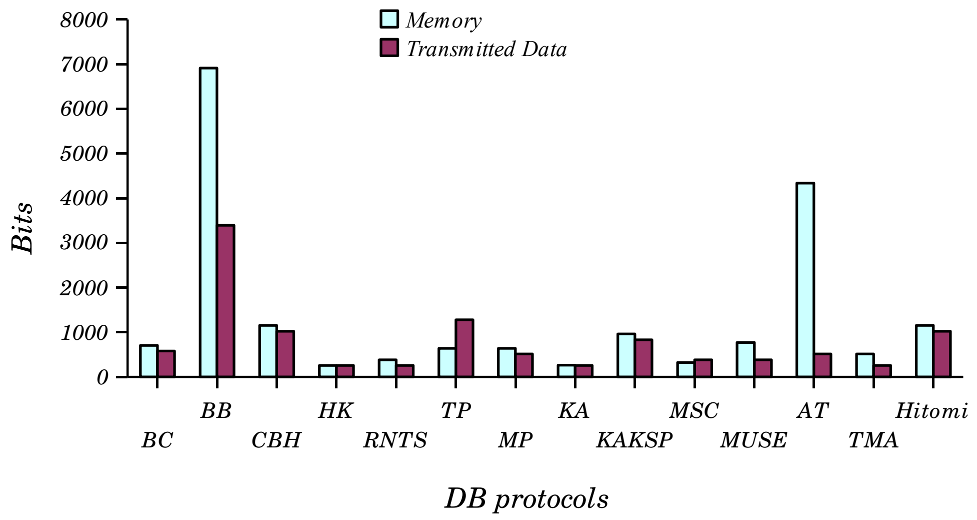


Figure 5.8. The required memory and transmitted data by the selected DB protocols (n = 64)

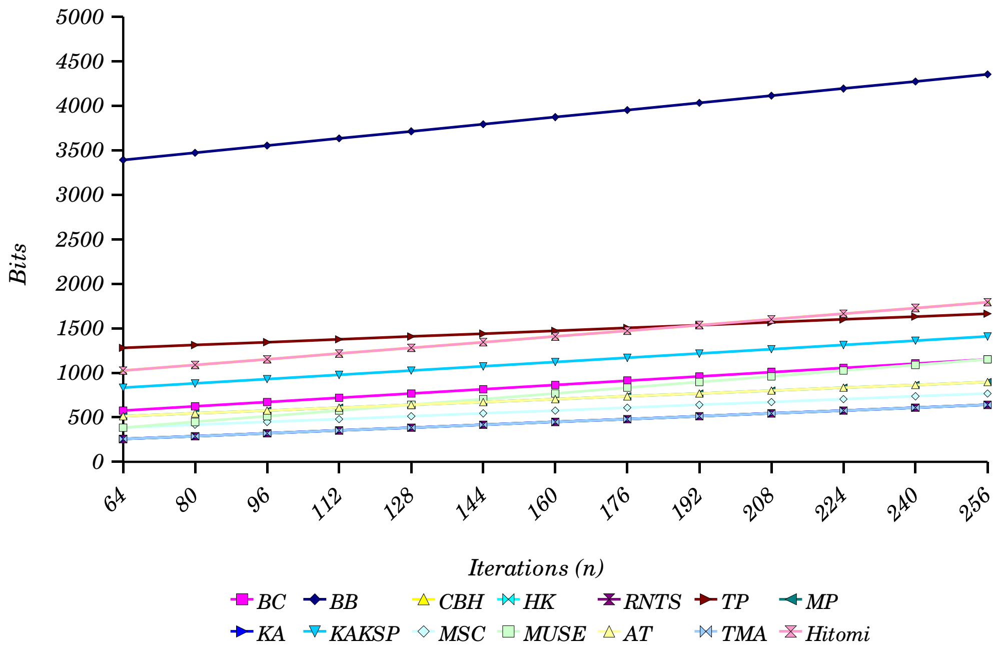


Figure 5.9. Transmitted data vs number of iterations (n)

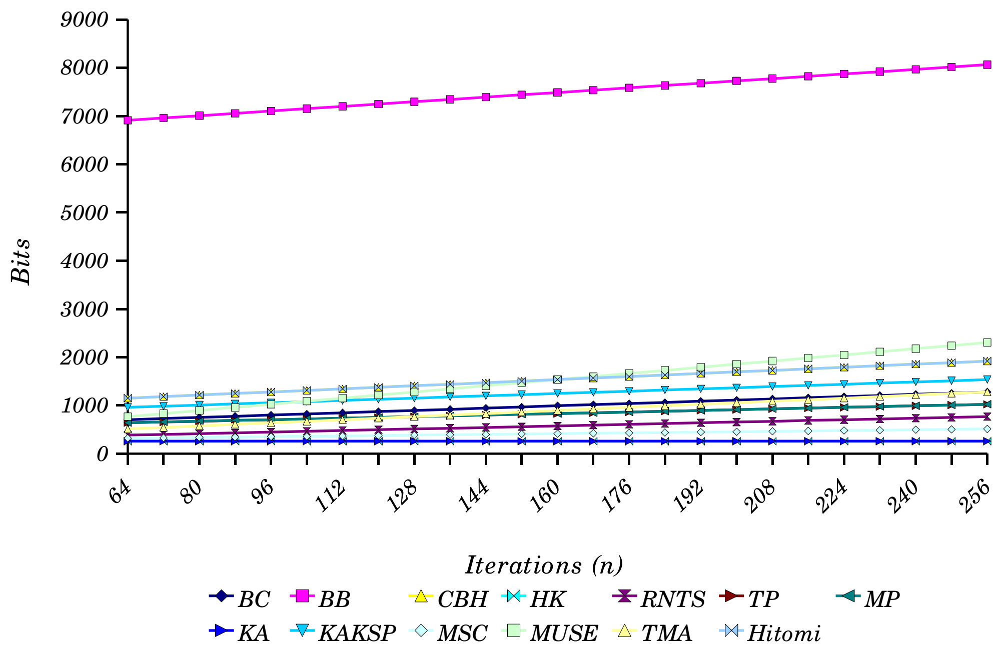


Figure 5.10. Memory vs number of iterations (n)

5.6.3 Computation

To compare the computational efficiency of each distance-bounding protocol, the number of variables or values that need to be computed during each protocol run are considered. The values considered are hash or pseudo-random function results, a symmetric encryption operation, MAC computation and random number generation. Figure 5.11 shows the number of computations needed by the prover for each protocol run. This is a slightly crude metric, as the exact times of executing these functions may vary in practice, especially considering the different algorithm options for each computation. However, in the case of the example given, the number of exchanges is not considerably larger than the input block sizes of the primitives, and it is therefore feasible that execution times could be comparable, e.g. a keyed pseudo-random function and a MAC should require a comparable time to compute. The protocol by BB does not appear in this figure, since it requires asymmetric cryptography, which requires more resources and cannot be compared like-with-like with the other processes.

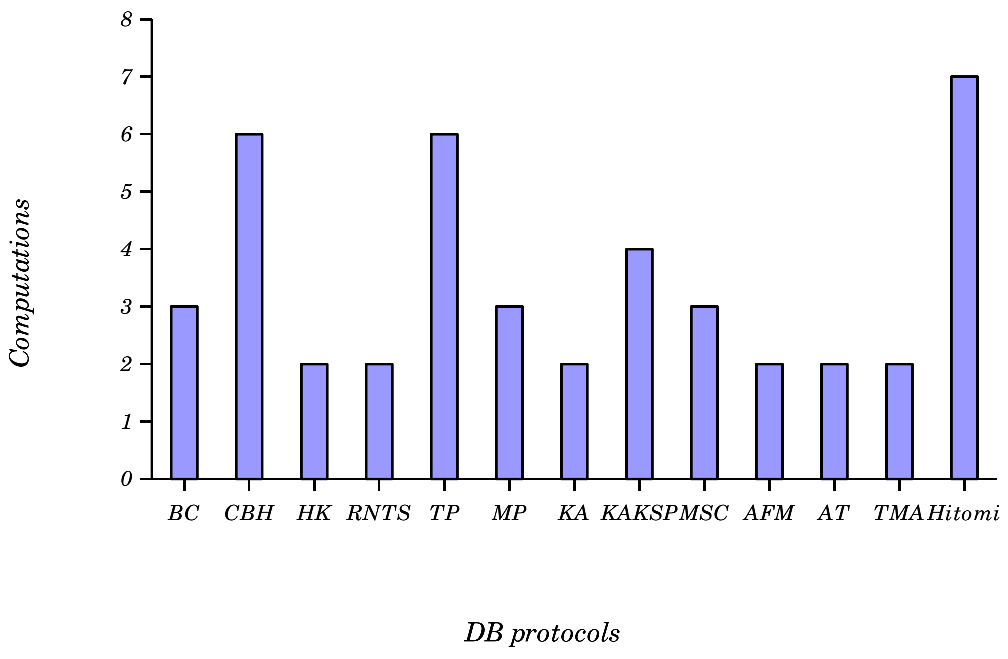


Figure 5.11. The number of computations required by the prover

5.6.4 Choosing a suitable protocol

There is no single protocol that is ideal under all design conditions. Each of the selected protocols discussed in Section 5.5 has advantages and disadvantages, and given the right design conditions any of these protocols can be argued to be ‘most suitable’. Given the basic trade-offs and results presented in this section, a system engineer should be able to get a good indication of which protocol could be used in his system. For example, if transaction time and resource constraints were not an issue, the protocol by BB provides the best level of security for all types of attack (as shown in Table 5.2). If the designer wanted to use minimal resources and wanted the protocol to complete in the shortest time possible, taking into account error correction, the HK, RNTS and KA proposals are possible choices. These three protocols require the least computation (Figure 5.11), transmit the lowest number of bits (Table 5.1, Figure 5.8) and require the least memory of protocols that do not have to be modified to accommodate bit errors (Table 5.1, Figure 5.8). In this case, the most suitable protocol choice could depend on the attacks that are to be mitigated(Table 5.1) and attack success (Table 5.1). RNTS protocol is the only choice for protection against terrorist fraud and offers equal security to HK protocol in mafia and distance fraud, at a cost of needing almost 50% additional memory. KA offers the strongest security against mafia fraud but is comparatively weak against distance fraud. If the attack probability needs to be lowered, Figure 5.6 and 5.7 would give an indication of the additional resources and transaction time required. There are also design factors beyond security, memory and transaction time, which were highlighted in Section 5.4. For example, if mutual distance bounding is required, CBH's MAD protocol would be the only option.

In general, the amount of resources required and the time needed for execution are reasonable for a WSN environment. If implemented using typical cryptographic algorithms in use today, 10 out of the 14 protocols require less than a kilobit of data to be stored and transmitted, which appears feasible for all but the most resource-constrained platforms. At the same time, the security level obtained using these algorithm choices is relatively strong, with the probability of an attack succeeding exceeding 10^{-8} in all but one case.

5.7 ATTACK RESISTANCE OF ALWADHA ALGORITHM

The previous two sections provided a comparative study of selected distance-bounding protocols that achieve the four principles proposed by the authors of [89]. This comparison could act as a guide for choosing a suitable distance-bounding protocol for implementing a secure distance estimation component in WSNs. However, implementing a secure localisation system should not rely only on using a secure distance-bounding protocol, for the following reasons:

- It could be difficult to achieve the four principles proposed by Clulow *et al.* [89]. For example, using a rapid bit exchange phase, in which the recipient can instantly react to the reception of each individual bit, requires a special type of channel that may not be available because of a certain limitation in the sensor nodes used.

- Distance-bounding protocols can only prevent malicious nodes from pretending that they are closer. However, an attacker can still pretend that it is more distant from the node that sent the "location request" packet than it really is.

- Using the RTT technique requires interaction between the two parties involved in the localisation system (the unknown and the other references), i.e. the unknown sends a "location request" packet, the neighbouring references send the "location response" packets and then the unknown estimates the round-trip time of the sending and receiving packets. However, some scenarios require using a passive localisation system, in which the references only cooperate with one another to estimate the unknown location. For example, references could locate the position of an attacker which is trying to interfere with the network. This type of attacker would not respond to the request sent by these references, and so they could simply use the RSS of the signals sent by this attacker to estimate the distance to it and then estimate its position.

- A correct distance estimation does not mean that the location information received is correct. As illustrated in Figure 5.12, a compromised beacon node (b_j) sends incorrect location information to an unknown pretending that it is in the location (x', y'). The distance between the beacon node and unknown node is d, the distance between the incorrect location (x', y') and the unknown is also d. Distance-bounding protocol will indicate that the estimated distance is correct. However, the unknown node could determine its location incorrectly.

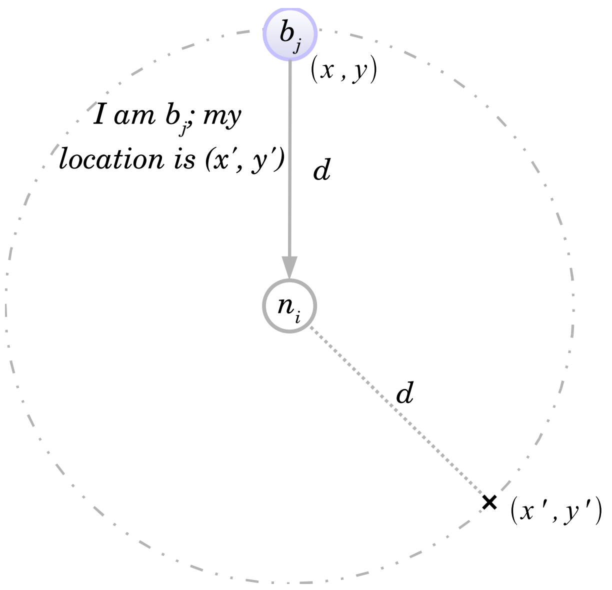


Figure 5.12. Compromised beacon node sends incorrect location information that is the same distance (d) to the unknown node

Therefore, the ALWadHA algorithm does not rely only on using a secure distance-bounding protocol but it also uses a robust position computation component that tolerates the existence of malicious nodes. The ALWadHA algorithm uses three types of filter to select the proper subset of references, as shown in Figure 4.3. Filter three is used to eliminate those references with a high distance error. On the other hand, this filter could be used to eliminate malicious nodes from the selected subset of references. The compromised nodes provide incorrect location information to mislead other nodes; however; pretending to be in an incorrect location increases the difference between the measured and estimated distance, which makes it easy to be detected by filter three, and as a result these malicious nodes will not participate in the localisation process. In fact, the goal of the ALWadHA algorithm is not to detect these malicious nodes; rather it is to enable nodes to live with them without disturbing the location estimation.

5.8 SIMULATION RESULTS

This section evaluates the resistance of the localisation algorithms against the two types of attack shown in Figure 5.1. In both types the attacker provides an incorrect location reference to mislead other nodes, which could determine their location incorrectly. The performance of the localisation algorithms is evaluated based on two metrics, firstly location error created by malicious nodes, and secondly the number of malicious nodes in the network. The random deployment characteristics described in Section 3.3.3 will be used.

5.8.1 Dishonest reference nodes

Four malicious nodes were distributed randomly in the network, with each sub-area having one malicious node (0.25 malicious node per r_{tx}^2). These malicious nodes pretended to be honest references and sent incorrect location references. The error of their location was generated randomly, using a normal random variant with a mean of 0.1% of the real location and a standard deviation changed from 1% to 50% of the real location. The mean error of location estimation is recorded at the end of the run time.

Figure 5.13 shows that increasing the erroneousness of the malicious nodes' location dramatically increases the mean error of estimated location in all algorithms except ALWadHA. The maximum mean error of the ALWadHA algorithm is equal to 4.76% of the transmission range, which occurs when the standard deviation is equal to 20% of the malicious nodes' location ($L_{malicious}$). Increasing the standard deviation reduces the mean error gradually till it reaches 3.51% at the standard deviation of 50% of $L_{malicious}$. The reason for this improvement is that increasing the erroneousness of the malicious nodes' advertised location also increases the difference between the measured and the estimated distance, and so filter three detects these nodes and eliminates them from the selected subset.

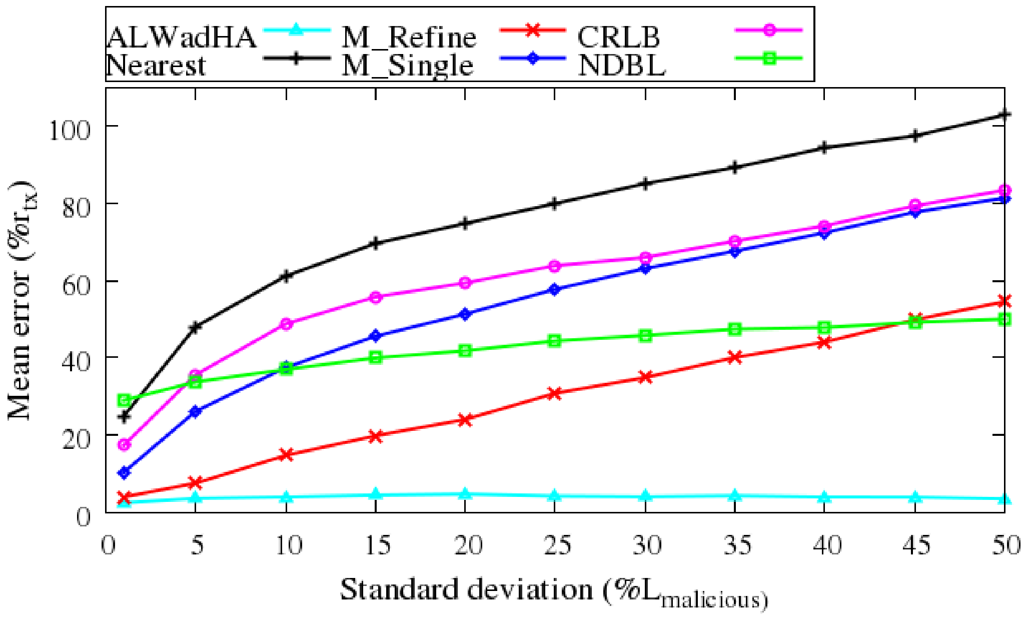


Figure 5.13. Location estimation error, four dishonest references

In the second experiment, the standard deviation was fixed to 50% of $L_{malicious}$ and the number of malicious references was changed from 4 to 16. Note that the number of beacon nodes used in the network is only 12. Figure 5.14 shows that ALWadHA outperforms other localisation algorithms and still achieves good accuracy of location estimation in spite of the increase in the number of malicious references. The mean error of location estimation of the ALWadHA algorithm in the presence of 16 malicious references is equal to 7.72% of the transmission range, which is much lower than that of the other localisation algorithms.

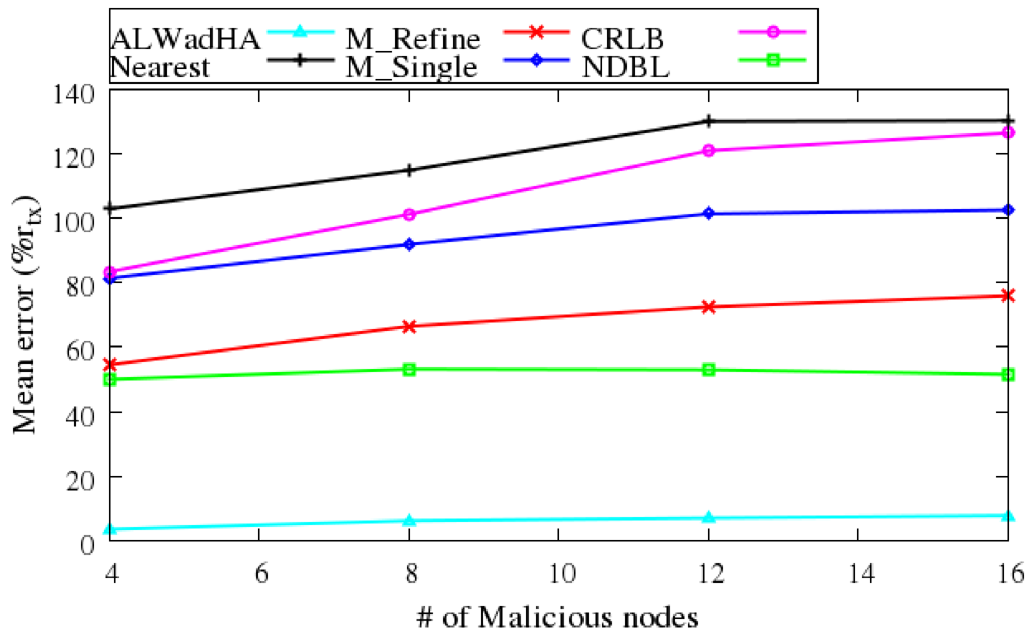


Figure 5.14. Location estimation error, standard deviation is equal to 50% of $L_{malicious}$

5.8.2 Compromised beacon nodes

The previous two experiments were repeated, using compromised beacon nodes instead of dishonest references. The results of these two experiments are shown in Figure 5.15 and 5.16. Compared with the previous two experiments, Figure 5.13 and 5.14, the Nearest, CRLB, M_Single and M_Refine algorithms yielded the same results, because these localisation algorithms do not distinguish between the reference and beacon nodes. For example, the Nearest algorithm selects the closest three nodes regardless of the type of these nodes, whether they are references or beacons. Therefore, an attacker has no need to perform the second type of attack and simply performs the first one. ALWadHA and NDBL algorithms distinguish between the reference and beacon nodes. Therefore, the mean error of location estimation is higher in the second type of attack. Figure 5.15 shows again that the maximum mean error is also at the standard deviation of 20% with a value of 6.43% of transmission range, while the mean error at a standard deviation of 50% is only 4.45% of the transmission range. Figure 5.16 shows that, in spite of there being 16 compromised beacons, which outnumber the existing benign beacons, the mean error of location estimation using ALWadHA algorithm is only 15.39% of the transmission range. This error is much less than that achieved by the other localisation algorithms.

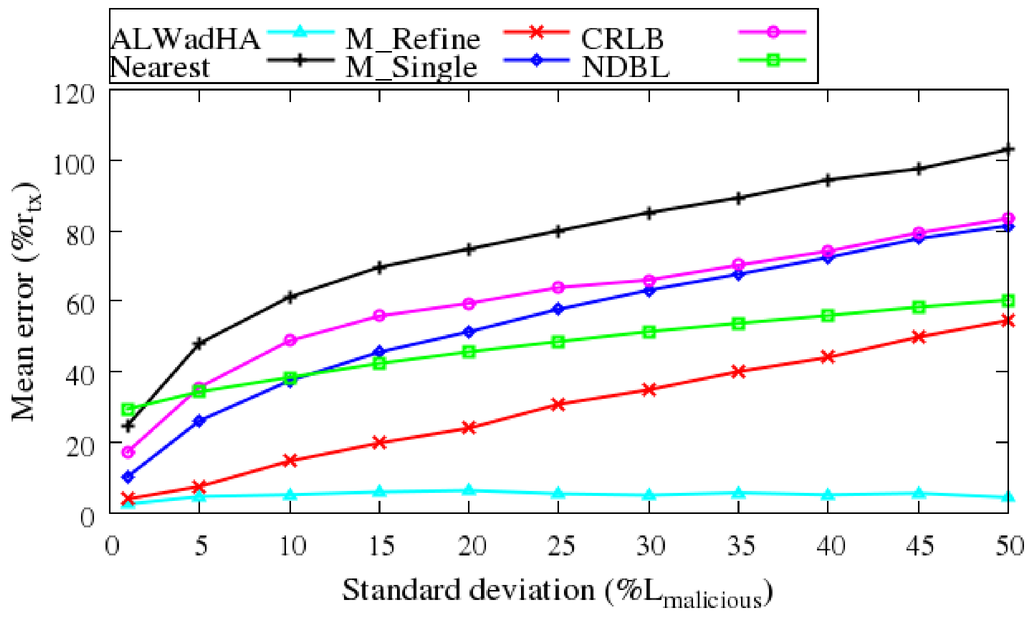


Figure 5.15. Location estimation error, four compromised beacon nodes

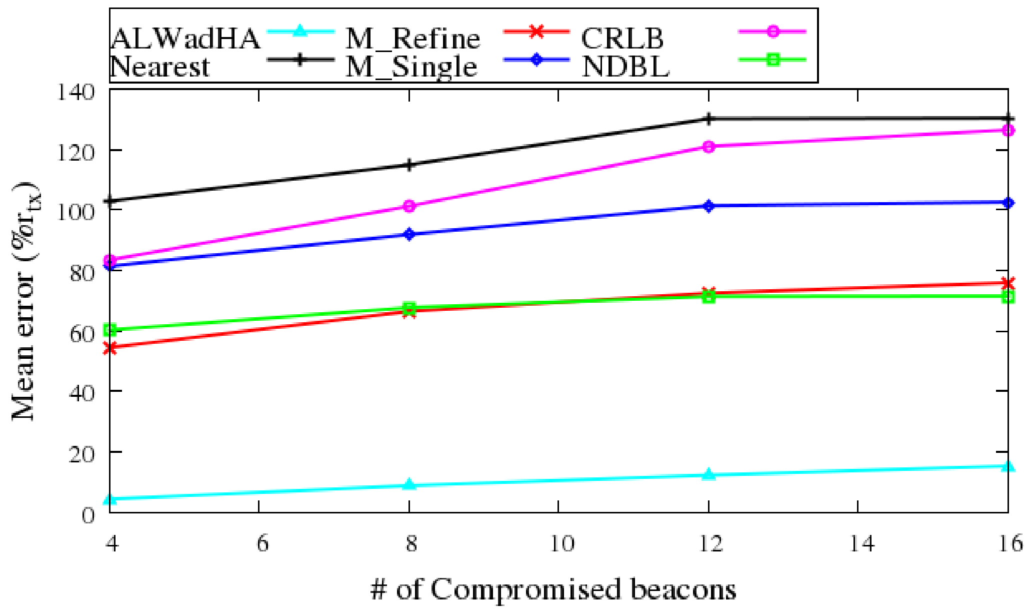


Figure 5.16. Location estimation error, standard deviation is equal to 50% of $L_{malicious}$

5.9 CHAPTER CONCLUSIONS

The security of a localisation system is a critical issue, because compromising the localisation system could disturb the entire functioning of WSNs. Attacks against localisation systems can be classified mainly into two categories: attacks on distance-estimation components and attacks on position-computation components. Compromising one of them could affect the entire localisation system. Therefore, a secure localisation system needs to take into account the security of these two components.

Several techniques can be used to implement secure distance-estimation components. However, this investigation suggested the use of a distance-bounding approach as a promising solution for ALWadHA. The chapter discussed the basics and the characteristics of a distance-bounding approach. It evaluated and compared several secure distance-bounding protocols, using different metrics. This discussion showed that distance-bounding protocols can be chosen that will perform well in a WSN environment and add minimal overheads to ALWadHA. Moreover, using a distance-bounding method will assist ALWadHA to accomplish several design objectives. Distance bounding can be done within an ad-hoc, mobile environment with any number of nodes and network topology, which does not introduce any conflict with the self-organising design objective targeted by ALWadHA. Distance bounding is a simple protocol that does not need synchronised clocks. Distance bounding involves only two nodes to estimate the distance between them, therefore it assists ALWadHA in achieving the simplicity, energy-efficiency and localised-design objectives. Lastly, using a secure distance-bounding protocol that adheres to the "principles of secure distance bounding" as defined by Clulow *et al.* [89], will assist ALWadHA to implement a secure localisation system.

ALWadHA uses a robust position computation technique that tolerates the existence of malicious nodes. The attack resistance of the ALWadHA algorithm was investigated simulating two types of attack. In the first type an attacker pretended to be an honest reference node, while in the second type an attacker compromised a beacon node. In both types of attack, the attacker sent incorrect location information to neighbouring nodes to mislead them in their location estimations. Simulation results showed that ALWadHA is able to determine the location of nodes where malicious nodes exist without undermining accuracy. Moreover, increasing the advertised location erroneousness of these malicious nodes makes it easier to detect these nodes and prevent them from taking part in the position computation.

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

Chapter 6

CONCLUSION

This investigation studied localisation systems and the different categories of position-discovery algorithms. On the basis of this study, the following approaches were employed to design a novel localisation algorithm: beacon-based, incremental, fine-grained, distributed, successive-refinement approaches, using a subset of references. Incorporating these approaches greatly improves the performance of the localisation algorithm, achieving several design objectives and suitability for WSNs.

Information fusion plays a critical role in WSNs. Therefore, the possibility of integrating information fusion into the localisation algorithm was studied. It was found that the best way to achieve this goal was to use a localised algorithm, which is a special type of distributed algorithm in which only a subset of nodes is involved in the position discovery process. This integration helped the proposed localisation algorithm to achieve two main goals simultaneously: location discovery, and the main objectives of information fusion, which are the improvement of accuracy and saving energy.

To achieve the above integration, a novel localisation algorithm, called ALWadHA, was proposed. ALWadHA is based on a smart reference-selection method that is able to select the best subset of references. The selected subset consists of the references that are most likely to contribute to accurate position computation of an unknown. The proposed algorithm was implemented and evaluated using simulation. Several experiments were conducted to evaluate the performance of ALWadHA, using criteria such as accuracy of estimation, number of references used and energy efficiency. The information fusion properties of the proposed scheme were analysed and several techniques used to make information fusion play a leading role were discussed.

The technique used by the proposed scheme to make it resistant to malicious attacks was also investigated.

The hypothesis of this study was that a *localisation algorithm can rely on using a low number of references to achieve an accurate estimation without compromising the simplicity, security, robustness or the energy efficiency of the algorithm.*

6.1 CONCLUSIONS

Most of the existing localisation algorithms rely on using a high number of references to estimate the position of nodes. On the one hand, this approach could enhance the accuracy of estimation. On the other hand, it has several drawbacks, especially for resource-constrained WSNs. Using a high number of references requires more computations and more memory space and consumes more energy. Therefore, this approach could be infeasible for resource-constrained WSNs. Moreover, the availability of a high number of references is a critical issue that cannot be guaranteed in WSNs. In a hostile environment, excluding malicious references would lead to more accurate estimation than using all the available references. In a noisy environment, nodes could enhance the accuracy of estimation if they exclude those references with high distance-measurement error that could bias the estimation toward an inaccurate location.

Using only a subset of references with a chance of higher accuracy could help to overcome the problems associated with using all the available references. Moreover, following this approach (i.e., using a subset of references) will help to achieve several design objectives, such as accuracy, robustness, simplicity, security, localisation and energy efficiency. However, selecting a subset of references is not an easy task or a straightforward technique that can simply be applied by selecting a low number of references. For instance, the technique proposed by the authors of [37] is based on selecting the nearest three references as a subset. However, in spite of its simplicity, it cannot be considered as a practical solution, because the accuracy, robustness and security of this technique are questionable. So it is a real challenge to achieve several design objectives at once. In fact, most of the existing techniques fulfil only a few design objectives, while compromising others. For example, a complex technique is used in [42] to enhance the accuracy and robustness of the localisation system, but this affects the simplicity, security and energy efficiency of this system.

An efficient localisation algorithm was designed. This algorithm relies on using a low number of references to achieve an accurate estimation without compromising other design objectives (such as simplicity, self-organising properties, robustness, security and energy efficiency) of the algorithm. This algorithm was termed **a**n efficient **l**ocalisation algorithm for **w**ireless **ad** hoc sensor networks with **h**igh **a**ccuracy (ALWadHA). The ALWadHA algorithm is based on a smart reference-selection method, which has the following characteristics:

- Requires a very simple computation. This method selects references based on location error using the probability of accuracy of the available references. Estimating the probability of accuracy requires only one division and few addition operations and it does not require any collaboration between neighbour nodes.

- Selects a low number of references, almost equal to the minimum possible number of references, which is three references. This reduction in the number of references used greatly improves the simplicity, accuracy and energy efficiency of the localisation algorithm.

- Is not an elimination method, based on eliminating a few references, but a real selection method. (The main disadvantage of the elimination method is that it could end up using all the available references or delete only a few of them.) Unlike most of the other techniques based on eliminating some references that do not satisfy certain conditions, the proposed method initially uses the minimum number of references, which is three references, and then checks if the selected subset satisfies a certain condition. If not, it adds one more reference and rechecks.

- Is smart, in the sense that it specifies the number of required references dynamically during the run time, based on the accuracy levels of the available references. The nodes close to beacons with high accuracy may use only a subset of three references, while those that have neighbour references with low accuracy will slightly increase the number of references used to overcome the error.

- Is not based on a specific distance-measurement model and can be used with any technique (e.g. RSS, TOA, RTT) without any modifications.

- Selects the best subset of references that enables nodes to estimate their position with high accuracy without involving a high number of references.

Using this smart reference-selection method not only enhances the accuracy of position estimation but also improves robustness, energy efficiency and security. The ALWadHA algorithm meets the following design objectives:

- **Accuracy:** The main objective of the ALWadHA algorithm is to determine the nodes' position with high accuracy. To achieve this, the localisation algorithm should not introduce a high estimation error and should be able to deal with accumulative computation error. The ALWadHA algorithm uses several techniques to accomplish these requirements. ALWadHA uses a smart reference-selection method that selects the best subset of references. This method requires simple computations that do not cause a high error rate. Finite precision is one source of error that influences localisation performance in WSNs. This type of error is due to inaccuracies induced by the limited computation precision of digital computers. These errors are important in WSNs because the sensors are resource constrained [16]. Therefore, reducing the complexity of computation could reduce the errors originating from this source.

  In the successive-refinement approach, which is used by ALWadHA, M_Refine and NDBL algorithms, the nodes keep re-estimating their position to enhance the accuracy of estimation. However, this approach could be influenced by the accumulated error, which increases gradually after each iteration. ALWadHA reduces the impact of the cumulative computation error by using a termination criterion that halts the process of estimation as soon as the node estimates its position with good accuracy. Moreover, unlike other successive-refinement algorithms, ALWadHA does not use the current estimated position to estimate the refined position in the next iteration; instead, it helps to specify the references that will be used and to check if the new refined position will be accepted. Simulation results show that, in cases where measurements are error free, the mean error using the ALWadHA algorithm is close to zero (0.00078 % of r_{tx}). In comparisons with other localisation algorithms, the mean error of the M_Refine algorithm increased gradually (0.97 % of r_{tx}) because of the computation error that accumulated during the refinement phases. The NDBL algorithm has a high mean error rate (27.95 % of r_{tx}), caused by the algorithm itself and the cumulative error during the refinement phases.

The accuracy of the ALWadHA algorithm was investigated with regard to several factors, such as node deployment, node density, network size and distance-measurement error. In all of these scenarios, the ALWadHA algorithm has excellent accuracy compared with other localisation algorithms. Simulation results also show that using the ALWadHA algorithm allows nodes to determine their position with high accuracy after a low number of iterations. This reduction in the required number of iterations dramatically reduces the computation cost. Therefore, it also could reduce the impact of errors that come from a finite precision source.

- **Self-organising properties:** Localisation algorithms should be independent of global infrastructure and beacon placement. Several localisation algorithms require particular beacon placement or require the beacons to be placed in a specific pattern. For example, [30] requires a triangle placement of beacons in a certain location. Several works in the literature have investigated the proposed localisation algorithms using only a specific scenario, such as a grid deployment, a low number of nodes or a small network.

  The ALWadHA algorithm does not depend on any specific node deployment and does not require a particular beacon placement. To investigate this design objective, two types of deployment were simulated: random and grid deployment. In random deployment the nodes are distributed randomly on the network, while in a grid deployment the beacons are placed inside grid cells selected randomly and the other nodes are then placed in the rest of the grid cells. In each experiment, the simulation was run 100 times and at each run the nodes were redistributed randomly. The ALWadHA algorithm was also investigated using different scenarios, varying the number of unknowns and beacons, and size of network. Simulation results showed that ALWadHA algorithm outperforms other localisation algorithms using these same scenarios in terms of estimation accuracy, energy efficiency and the number of references used.

- **Simplicity:** The ALWadHA algorithm took into account the limited resources of sensor nodes and it was designed to be as simple as possible without compromising other design objectives. The core of the ALWadHA algorithm is the smart reference-selection method. This method follows several techniques to achieve simplicity, such as selecting references using their probability of accuracy, estimating this probability using only one division and few addition operations; and selecting only a few references, which greatly reduces the computation cost.

  The ALWadHA algorithm deals with location error and distance-measurement error separately, using filter two and filter three respectively. Filter two is used to select a subset of references based on location error, while filter three is used to eliminate those references with high distance-measurement error (see Section 4.4 for a detailed discussion of these filters). The ALWadHA algorithm always uses filter two to estimate the initial position. Filter two is based on the probability of accuracy, which requires very simple computations. The ALWadHA algorithm uses filter three only when there is at least a reference in the subset used that has a high distance-measurement error. Dealing with the two types of error separately also reduces the computation cost.

  Simulation results show that the ALWadHA algorithm uses a low number of references, almost equal to the minimum possible number of references (which is three references). Moreover, it performs a lower number of iterations compared with other refinement-localisation algorithms, such as the NDBL and M_Refine algorithms. The simplicity of the ALWadHA algorithm could enhance not only the accuracy of estimation but also resource usage, such as CPU usage and memory space required.

- **Robustness:** Relying on using all or most of the available references to enhance the accuracy of estimation could reduce the robustness of the localisation algorithm. A high number of references might not be available in WSNs because sensor nodes are prone to failure from lack of power or physical damage, or they could be unreachable because of obstacles or node movements. The ALWadHA algorithm allows nodes to determine their positions using only a few references. This makes ALWadHA very tolerant of node failures.

The ALWadHA algorithm uses three types of filter to deal with localisation errors. The first filter is used by known nodes, while the other two filters are used by the node itself. Filter one is used to ensure that only the known nodes with high accuracy will send their "location response" packets. Filter two is used to deal with location error based on the probability of accuracy. Filter three is used to deal with distance-measurement error. In addition to these filters, the ALWadHA algorithm uses the successive-refinement approach, which also enhances the robustness of the ALWadHA algorithm. Simulation results proved the robustness of the ALWadHA algorithm and showed that even with underlying measurement error, ALWadHA achieves acceptable accuracy that is, on average, not much worse than the basic measurements. When an error was added with a standard deviation equal to 10% of the measured distance, the mean error of ALWadHA was only 18.39% of transmission range (r_{tx}); for M_Refine it was 21.62% of r_{tx} and for NDBL it was 41.58% of r_{tx}. The mean error of the single-estimation algorithms was much higher; for example the mean error for the Nearest algorithm was 77.54% of r_{tx}.

- **Energy efficiency:** In order to make ALWadHA an energy-aware localisation algorithm, several techniques are followed to reduce computation and communication overheads. To reduce the computation overheads, the ALWadHA algorithm uses simple computation, requires few references and uses a termination criterion to reduce the number of iterations. To reduce the communication overheads, the ALWadHA algorithm reduces the number of propagated messages required by the localisation system, i.e. the "location request" and "location response" packets. The ALWadHA algorithm uses a termination criterion to halt the process of localisation as soon as the node has determined its position with good accuracy. The use of this criterion reduces the number of "location request" packets. Filter one is used to reduce the number of "location response" packets by allowing only references with high accuracy to send their responses. Simulation results showed that the ALWadHA algorithm required a lower number of "location request" and "location response" packets and consumed less energy than other successive-refinement localisation algorithms.

- **Localised algorithm:** The localised algorithms used for location discovery should comply with the following three conditions: Firstly, they should request and process information with regard to the localisation algorithm only locally, without any central coordination overhead. Secondly, only a subset of nodes should take part in the position estimation process. Finally, the selected subset should be the one most likely to contribute to high, accurate position estimation. The ALWadHA algorithm completely satisfies these three conditions. Therefore, it can be regarded as a localised algorithm. It has moreover been shown that using a localised information-fusion technique based on an efficient reference-selection method would enhance the performance of the algorithms and attain several design objectives. Information fusion can play a leading role in localisation algorithms by guiding the location-discovery process simultaneously with the fusion process.

- **Information fusion:** The ALWadHA algorithm does not use information fusion in a supporting role that assists only in position estimation. Information fusion in the ALWadHA algorithm plays a leading role that guides the location-discovery process and the fusion process simultaneously. When the possibility of integrating information fusion with the localisation system was investigated; it was realised that the best way to achieve this goal would be to use a localised algorithm. Information fusion used in localisation algorithms was classified into three levels and the discussion showed how the ALWadHA algorithm used these three levels to allow nodes to determine their position with high accuracy and at the same time achieve several information-fusion objectives. Simulation results showed that using these three levels of information fusion greatly improves the accuracy, simplicity, robustness, security and energy efficiency of the ALWadHA algorithm.

- **Security:** Wireless sensor networks require a secure localisation system that is able to work in a hostile environment and to prevent compromised nodes from participating in the localisation process. A secure localisation system should consider the security aspects of the three components of this system, namely the distance/angle estimation, position computation and localisation algorithm. To provide secure distance estimation, the use of a distance-bounding approach was suggested for ALWadHA. Distance bounding is simple to integrate, does not require synchronised clocks, performs well in the WSNs environment, involves only two nodes to estimate the distance between them, and adds minimal overheads to ALWadHA. However, using a secure distance-bounding protocol is not enough to secure the entire localisation system. Therefore, the ALWadHA algorithm also uses a robust position-computation component that tolerates the existence of malicious nodes.

The filter three that is used in the ALWadHA algorithm not only enhances localisation accuracy, but could also enhance the security of the algorithm. Malicious nodes provide incorrect location information to mislead other nodes. However, pretending to be in a different location increases the difference between the measured and estimated distance, which makes it easy for filter three to detect it, and these malicious nodes will consequently not participate in the localisation process. Simulation results showed the attack-resistance of the ALWadHA algorithm and proved that ALWadHA is able to determine nodes' position in the hostile environment without undermining the estimation accuracy.

Therefore, it can be said that the hypothesis of this study, that a *localisation algorithm can rely on using a low number of references to achieve an accurate estimation without compromising the simplicity, security, robustness or the energy efficiency of the algorithm,* has been proved.

6.2 SUMMARY OF CONTRIBUTIONS

This section will summarise the main contributions of this work into three areas: smart selection localisation; information fusion in localisation systems; and distance-bounding protocols.

- **Smart selection localisation:**

  This thesis proposed a novel localisation algorithm (ALWadHA) to solve the problem of determining sensor nodes' location. ALWadHA is based on a smart reference-selection method. The main unique characteristics of this algorithm may be outlined as follows:

  - It relieves the burden of using a high number of references to achieve good accuracy. Relying on using a high number of references in WSNs could lead to several problems.

- It uses a novel smart reference-selection method that is not based on measured distance (which could be corrupted by multiplicative noise).
- It considers the three types of error (i.e. computation error, distance-measurement error and location error) and deals with them separately.
- It employs the three levels of information fusion to enhance the performance of the algorithm.
- It achieves good performance under various operational conditions, such as an error-free environment, noisy environment and hostile environment.
- Unlike other successive-refinement algorithms, it does not use the current estimated position to estimate the refined position in the next iteration; instead it uses it to help specify the references that will be used and to check whether the new refined position will be accepted.

Most of the localisation algorithms in the literature are based on using all the available references. In contrast, only limited research has been done on selecting a subset of references. Each of these previous algorithms targets only one or a few design objectives. However, none of them is able to achieve all the design objectives. The proposed algorithm accomplishes several design objectives, which can be considered an achievement that provides more motivation for this investigation.

- **Information fusion in localisation systems:**

Research on analysing localisation systems from the information-fusion perspective is hardly reflected in the literature. This thesis has presented an overview of localisation systems that are based on information fusion, and has shown that using a localised algorithm makes information fusion play a leading role. The thesis also analysed a number of approaches used by localised information-fusion algorithms, highlighted some of their strengths and weaknesses, briefly compared them and showed how designers could decide which approach should be followed to implement their localised algorithms. An intensive literature review showed that no similar analysis has been published.

Information fusion used in localisation algorithms was classified into three levels. Most of the existing localisation algorithms, especially those using all of the available references, use only the first level of information fusion, and that in a supporting role. The localised algorithms usually use the first two levels of information fusion. A very limited number of localisation algorithms have used level three of information fusion. However, these algorithms required additional interaction between nodes to achieve this level. The ALWadHA algorithm employs the three levels of information fusion and does not require any interaction between nodes to achieve the third level.

- **Distance-bounding protocols:**

This thesis has discussed a range of techniques for distance estimation, with more focus on the distance-bounding technique. It conducted an analysis of aspects and performance with regard to using distance bounding in WSN localisation. Thereafter, the use of a distance-bounding approach was suggested as a promising solution for localisation systems, and especially for the proposed algorithm. The thesis gave an overview of distance bounding and provided a comparative study of selected distance-bounding protocols. The result illustrates the practical resource requirements and performance trade-offs involved in different protocols, and could act as a guide for choosing a suitable distance-bounding protocol when implementing a secure distance-estimation component for localisation systems in WSNs. Although a few works have given a limited comparison of distance-bounding protocols, they are not as comprehensive as the one in this thesis. Moreover, none of them discusses aspects of using distance bounding in WSN localisation.

6.3 FUTURE WORK

Determining the position of nodes in WSNs is an interesting and challenging research area with many unsolved problems. The work presented in this thesis answers a few questions, while it opens the door to many more new riddles. Future challenges and a possible continuation of this work can be summarised as follows:

- **Mobile sensor networks:** The ALWadHA algorithm was developed and investigated with static WSNs in mind. An extension of this algorithm could be done by modifying and investigating the applicability of the ALWadHA algorithm to mobile sensor networks.

- **Energy efficiency:** One of the most important challenges is enhancing the energy efficiency of the proposed algorithm. Despite the ALWadHA algorithm's consuming less energy than other successive-refinement localisation algorithms, future research is required to achieve more energy efficiency.

- **Distance-bounding protocol:** Several proposals are still vulnerable because of the way the communication channel is implemented. Currently, there are very few practical implementations of distance-bounding channels suitable for WSNs; and while the number of distance-bounding protocols continues to increase, practical channels suitable for distance bounding remain a relatively unexplored topic for the future. Tippenhauer and Čapkun [118] have demonstrated a distance-bounding channel using off-the-shelf UWB components, which takes into account the security uncertainty introduced at the packet format layer, but otherwise channels have mostly been defined for near-field RFID systems [134] or contact smart cards [135].

- **Secure localisation system:** Implementing a secure localisation system is another challenging area. This system requires the following: firstly, integrating distance bounding into a localisation scheme (such as ALWadHA or an extension of it). The distance-bounding protocol used should adhere to the ‘principles of secure distance bounding’ defined in [89] and suitable for WSNs. Secondly, the performance must be evaluated, both in terms of overheads and resistance to attack.

- **Implementation with real motes:** This thesis assessed the performance evaluation and the design objectives of the proposed algorithm analytically and by computer simulations. An extension of this work to actual implementation of the ALWadHA algorithm and field testing on wireless sensor nodes would be a challenging task. Moreover, testing the ALWadHA algorithm in real applications, which require location knowledge of equipment or people using WSNs in areas such as robust situational awareness in underground mining, would be an interesting topic for future research.

REFERENCES

[1] J. Yick, B. Mukherjee, and D. Ghosal, "Wireless sensor network survey," *Computer Networks,* vol. 52, no. 12, pp. 2292-2330. April 2008.

[2] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Communications Magazine,* vol. 40, no. 8, pp. 102-114. February 2002.

[3] M. Hussain, P. Khan, and K. Sup, "WSN research activities for military application," in *Proceedings of the 11th International Conference on Advanced Communication Technology — ICACT '09,* 15-18 February, Phoenix Park, Korea, vol. 1, 2009, pp. 271-274.

[4] S. H. Toh, K. H. Do, W. Y. Chung, and S. C. Lee, "Health decision support for biomedical signals monitoring system over a WSN," in *Proceedings of the 2nd International Symposium on Electronic Commerce and Security — ISECS '09,* 22-24 May, Nanchang City, China, vol. 1, 2009, pp. 605-608.

[5] N. Wirawan, S. Rachman, I. Pratomo, and N. Mita, "Design of low cost wireless sensor networks-based environmental monitoring system for developing country," in *Proceedings of the 14th IEEE Asia-Pacific Conference on Communications — APCC '08,* 14-16 October, Tokyo, Japan, 2008, pp. 1-5.

[6] Werner-Allen, K. Lorincz, M. Ruiz, O. Marcillo, J. Johnson, J. Lees, and M. Welsh, "Deploying a wireless sensor network on an active volcano," *IEEE Internet Computing,* vol. 10, no. 2, pp. 18-25. 2006.

[7] S. Ratnasamy, B. Karp, L. Yin, F. Yu, D. Estrin, R. Govindan, and S. Shenker, "GHT: A geographic hash table for data-centric storage," in *Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications — WSNA '02,* 28 September, Atlanta, Georgia, USA, 2002, pp. 78-87.

[8] B. Karp and H. T. Kung, "GPSR: Greedy perimeter stateless routing for wireless networks," in *Proceedings of the 6th Annual International Conference on Mobile

Computing and Networking — MobiCom '00, 06-11 August, Boston, Massachusetts, USA, 2000, pp. 243-254.

[9] J. Li, J. Jannotti, D. S. J. De Couto, D. R. Karger, and R. Morris, "A scalable location service for geographic ad hoc routing," in *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking — MobiCom '00,* 06-11 August, Boston, Massachusetts, USA, 2000, pp. 120-130.

[10] Y. C. Hu, A. Perrig, and D. B. Johnson, "Packet leashes: A defense against wormhole attacks in wireless networks," in *Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies — INFOCOM '03,* 30 March – 03 April, San Francisco, California, USA, vol. 3, 2003, pp. 1976-1986.

[11] Y. Xu, J. Heidemann, and D. Estrin, "Geography-informed energy conservation for ad hoc routing," in *Proceedings of the 7th Annual International Conference on Mobile Computing and Networking — MobiCom '01,* 16-21 July, Rome, Italy, 2001, pp. 70-84.

[12] A. Srinivasan and J. Wu, "Wireless sensor networks (WSNs): Secure localization," *Encyclopedia of Wireless and Mobile Communications,* pp. 1545-1571. April 2008.

[13] D. Lieckfeldt, J. You, and D. Timmermann, "An algorithm for distributed beacon selection," in *Proceedings of the 6th Annual IEEE International Conference on Pervasive Computing and Communication — PerCom '08,* 17-21 March, Hong Kong, China, 2008, pp. 318-323.

[14] B. V. Dasarathy, "More the merrier... or is it? sensor suite augmentation benefits assessment," in *Proceedings of the 3rd International Conference on Information Fusion,* 10-13 July, Paris, France, vol. 2, 2000, pp. WeC3/20-WeC3/25.

[15] K. Langendoen and N. Reijers, "Distributed localization in wireless sensor networks: A quantitative comparison," *Journal of Computer Networks,* vol. 43, no. 4, pp. 499-518. November 2003.

[16] S. Slijepcevic, S. Megerian, and M. Potkonjak, "Location errors in wireless embedded sensor networks: Sources, models, and effects on applications," *ACM SIGMOBILE Mobile Computing and Communications Review,* vol. 6, no. 3, pp. 67-78. 2002.

[17] S. Meguerdichian, S. Slijepcevic, V. Karayan, and M. Potkonjak, "Localized algorithms in wireless ad-hoc networks: Location discovery and sensor exposure," in *Proceedings of the 2nd ACM International Symposium on Mobile Ad Hoc Networking & Computing,* 04 – 05 October, Long Beach, CA, USA, 2001, pp. 106-116.

[18] A. Boukerche, H. Oliveira, E. F. Nakamura, and A. A. Loureiro, "Secure localization algorithms for wireless sensor networks," *IEEE Communications Magazine,* vol. 46, pp. 96-101, 2008.

[19] K. K. Chintalapudi, A. Dhariwal, R. Govindan, and G. Sukhatme, "Ad-hoc localization using ranging and sectoring," in *Proceedings of the 23rd Annual Joint Conference of the IEEE Computer and Communications Societies — INFOCOM '04,* 07-11 March, Hong kong, China, vol. 4, 2004, pp. 2662-2672.

[20] J. Wang, R. K. Ghosh, and S. K. Das, "A survey on sensor localization," *Journal of Control Theory and Applications,* vol. 8, no. 1, pp. 2-11. 2010.

[21] F. Franceschini, M. Galetto, D. Maisano, and L. Mastrogiacomo, "A review of localization algorithms for distributed wireless sensor networks in manufacturing," *International Journal of Computer Integrated Manufacturing,* vol. 22, no. 7, pp. 698-716. 2009.

[22] Q. Shi, H. Huo, T. Fang, and D. Li, "A distributed node localization scheme for wireless sensor networks," *Wireless Personal Communications,* vol. 53, no. 1, pp. 15-33. 2010.

[23] D. Moore, J. Leonard, D. Rus, and S. Teller, "Robust distributed network localization with noisy range measurements," in *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems,* 03-05 November, Baltimore, MD, USA, 2004, pp. 50-61.

[24] J. Liu, Y. Zhang, and F. Zhao, "Robust distributed node localization with error management," in *Proceedings of the 7th ACM International Symposium on Mobile Ad Hoc Networking and Computing,* 22-25 May , Florence, Italy, 2006, pp. 250-261.

[25] N. Patwari, J. N. Ash, S. Kyperountas, A. O. Hero III, R. L. Moses, and N. S. Correal, "Locating the nodes: cooperative localization in wireless sensor networks," *IEEE Signal Process. Mag.,* vol. 22, pp. 54-69, 2005.

[26] X. Ji and H. Zha, "Sensor positioning in wireless ad-hoc sensor networks using multidimensional scaling," in *Proceedings of the 23rd Annual Joint Conference of the IEEE Computer and Communications Societies — INFOCOM '04,* 07-11 March, Hong Kong, PR China, vol. 4, 2004, pp. 2652-2661.

[27] M. Z. Rahman and L. Kleeman, "Paired measurement localization: A robust approach for wireless localization," *IEEE Transactions on Mobile Computing,* vol. 8, no. 8, pp. 1087-1102. 2009.

[28] H. Lim and J. C. Hou, "Localization for anisotropic sensor networks," in *Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies — INFOCOM '05,* 13-17 March, Miami, Florida, USA, vol. 1, 2005, pp. 138-149.

[29] S. Capkun and J. P. Hubaux, "Secure positioning in wireless networks," *IEEE Journal on Selected Areas in Communications,* vol. 24, no. 2, pp. 221-232. 2006.

[30] S. Zhang, G. Li, W. Wei, and B. Yang, "A novel iterative multilateral localization algorithm for wireless sensor networks," *Journal of Networks,* vol. 5, no. 1, pp. 112-119. January 2010.

[31] C. Tran-Xuan, V. H. Vu, and I. Koo, "Calibration mechanism for RSS based localization method in wireless sensor networks," in *Proceedings of the 11th International Conference on Advanced Communication Technology — ICACT '09,* 15-18 February, Gangwon-Do, South Korea, vol. 1, 2009, pp. 560-563.

[32] C. Chai, "Efficient intelligent localization scheme for distributed wireless sensor networks," *Journal of Computers,* vol. 4, no. 11, pp. 1159-1166. November 2009.

[33] Q. Shi, C. He, H. Chen, and L. Jiang, "Distributed wireless sensor network localization via sequential greedy optimization algorithm," *IEEE Transactions on Signal Processing,* vol. 58, no. 6, pp. 3328-3340. 2010.

[34] J. Wan, N. Yu, R. Feng, Y. Wu, and C. Su, "Localization refinement for wireless sensor networks," *Computer Communications,* vol. 32, no. 13-14, pp. 1515-1524. 2009.

[35] H. Jiang, R. Cao, and X. Wang, "Apply modified method of nonlinear optimization to improve localization accuracy in WSN," in *Proceedings of the 5th International Conference on Wireless Communications, Networking and Mobile Computing — WiCOM '09,* 24-26 September, Beijing, China, 2009, pp. 1-6.

[36] A. Savvides, C. C. Han, and M. B. Strivastava, "Dynamic fine-grained localization in ad-hoc networks of sensors," in *Proceedings of the 7th Annual International Conference on Mobile Computing and Networking — MobiCom '01,* 16-21 July, Rome, Italy, 2001, pp. 166-179.

[37] K. Y. Cheng, V. Tam, and K. S. Lui, "Improving aps with anchor selection in anisotropic sensor networks," in *Proceedings of the Joint International Conference on Autonomic and Autonomous Systems and International Conference on Networking and Services — ICAS-ICNS '05,* 23-28 October, Papeete, Tahiti, 2005, pp. 49-54.

[38] K. Sinha and A. Chowdhury, "A beacon selection algorithm for bounded error location estimation in ad hoc networks," in *Proceedings of the IEEE International Conference on Computing: Theory and Applications — ICCTA '07,* 05-07 March, Kolkata, India, 2007, pp. 87-93.

[39] L. M. Kaplan, "Local node selection for localization in a distributed sensor network," *IEEE Transactions on Aerospace and Electronic Systems,* vol. 42, no. 1, pp. 136-146. 2006.

[40] J. Albowicz, A. Chen, and L. Zhang, "Recursive position estimation in sensor networks," in *Proceedings of the 9th IEEE International Conference on Network Protocols — ICNP '01,* 11-14 November, Riverside, California, USA, 2001, pp. 35-41.

[41] D. Liu, P. Ning, A. Liu, C. Wang, and W. K. Du, "Attack-resistant location estimation in wireless sensor networks," *ACM Transactions on Information and System Security,* vol. 11, no. 4, 2008.

[42] J. A. Costa, N. Patwari, and A. O. Hero III, "Distributed weighted-multidimensional scaling for node localization in sensor networks," *ACM Transactions on Sensor Networks,* vol. 2, no. 1, pp. 39-64. 2006.

[43] S. Han, S. Lee, S. Lee, J. Park, and S. Park, "Node distribution-based localization for large-scale wireless sensor networks," *Wireless Networks,* vol. 16, no. 5, pp. 1389-1406. 2010.

[44] L. Hu and D. Evans, "Using directional antennas to prevent wormhole attacks," in *Proceedings of the 11th Network and Distributed System Security Symposium — NDSS '04,* 05-06 February, San Diego, CA, USA, 2004, pp. 131-141.

[45] K. B. Rasmussen and S. Capkun, "Implications of radio fingerprinting on the security of sensor networks," in *Proceedings of 3rd IEEE International Conference on Security and Privacy in Communications Networks — SecureComm '07,* 17-21 September, Nice, France, 2007, pp. 331-340.

[46] R. Maheshwari, J. Gao, and S. R. Das, "Detecting wormhole attacks in wireless networks using connectivity information," in *Proceedings of the 26th IEEE International Conference on Computer Communications — INFOCOM '07,* 06-12 May, Anchorage, AL, USA, 2007, pp. 107-115.

[47] C. Meadows, P. Syverson, and L. W. Chang, "Towards more efficient distance bounding protocols for use in sensor networks," in *Proceedings of the 2nd International Conference on Security and Privacy in Communication Networks — SecureComm '06,* 28 August – 01 September, Baltimore, MD, USA, 2006, pp. 1-5.

[48] S. Tian, X. Zhang, X. Wang, P. Sun, and H. Zhang, "A selective anchor node localization algorithm for wireless sensor networks," in *Proceedings of the International Conference on Convergence Information Technolog — ICCIT '07,* 21-23 November, Gyeongju, Korea, 2007, pp. 358-362.

[49] N. B. Priyantha, H. Balakrishnan, E. Demaine, and S. Teller, "Anchor-free distributed localization in sensor networks," MIT Laboratory for Computer Science, Tech. Rep. 892, April 2003 .

[50] X. Nguyen, M. I. Jordan, and B. Sinopoli, "A kernel-based learning approach to ad hoc sensor network localization," *ACM Transactions on Sensor Networks (TOSN),* vol. 1, no. 1, pp. 134-152. 2005.

[51] Y. Shang, W. Rumi, Y. Zhang, and M. Fromherz, "Localization from connectivity in sensor networks," *IEEE Transactions on Parallel and Distributed Systems,* vol. 15, no. 11, pp. 961-974. 2004.

[52] P. Biswas and Y. Ye, "Semidefinite programming for ad hoc wireless sensor network localization," in *Proceedings of the 3rd International Symposium on Information Processing in Sensor Networks — IPSN '04,* 26-27 April, Berkeley, California, USA, 2004, pp. 46-54.

[53] D. Niculescu and B. Nath, "Ad hoc positioning system (APS)," in *Proceedings of the IEEE Global Telecommunications Conference — GLOBECOM '01,* 25-29 November, San Antonio, Texas, USA, vol. 5, 2001, pp. 2926-2931.

[54] J. Blumenthal, D. Timmermann, C. Buschmann, S. Fischer, J. Koberstein, and N. Luttenberger, "Minimal transmission power as distance estimation for precise localization in sensor networks," in *Proceedings of the International Conference on Wireless Communications and Mobile Computing,* 03-06 July, Vancouver, British Columbia, Canada, 2006, pp. 1331-1336.

[55] J. Blumenthal, F. Reichenbach, and D. Timmermann, "Precise positioning with a low complexity algorithm in ad hoc wireless sensor networks," *Praxis Der Informationsverarbeitung Und Kommunikation,* vol. 28, no. 2, pp. 80-85. 2005.

[56] D. Liu, P. Ning, and W. Du, "Detecting malicious beacon nodes for secure location discovery in wireless sensor networks," in *Proceedings of the 25th IEEE International Conference on Distributed Computing Systems — ICDCS ’05,* 06-10 June, Columbus, Ohio, USA, 2005, pp. 609-619.

[57] N. Patwari, A. O. Hero, M. Perkins, N. S. Correal, and R. J. O'Dea, "Relative location estimation in wireless sensor networks," *IEEE Transactions on Signal Processing,* vol. 51, no. 8, pp. 2137-2148. 2003.

[58] D. Lieckfeldt, J. You, and D. Timmermann, "Distributed selection of references for localization in wireless sensor networks," in *Proceedings of the 5th Workshop on*

Positioning, Navigation and Communication — WPNC '08, 22-27 March, Hannover, Germany, 2008, pp. 31-36.

[59] T. Issariyakul and E. Hossain, *Introduction to Network Simulator (NS2).* New York, NY, USA: Springer, 2009.

[60] "The network simulator - ns-2," 12 February 2010, http://nsnam.isi.edu/nsnam/index.php/User_Information. Last accessed on 26 November 2010.

[61] K. Fall and K. Varadhan, "The Ns Manual," 9 May 2010, http://www.isi.edu/nsnam/ns/doc/ns_doc.pdf. Last accessed on 26 November 2010.

[62] P. Morávek, "Ns-2 simulator capabilities in nodes localization in wireless networks," 2009, http://www.feec.vutbr.cz/EEICT/2009/sbornik/03-Doktorske%20projekty/01-Elektronika%20a%20komunikace/06-xmorav08.pdf. Last accessed on 26 November 2010.

[63] "The doxygen documentation system," 12 October 2010, http://www.stack.nl/~dimitri/doxygen/. Last accessed on 26 November 2010.

[64] "The perl programming language," 2010, http://www.perl.org/. Last accessed on 26 November 2010.

[65] "The gnuplot software," September 2010, http://www.gnuplot.info. Last accessed on 26 November 2010.

[66] B. V. Dasarathy, "Information fusion – what, where, why, when, and how?" *Information Fusion (Editorial),* vol. 2, no. 2, pp. 75-76. 6 2001.

[67] E. F. Nakamura, A. A. F. Loureiro, and A. C. Frery, "Information fusion for wireless sensor networks: Methods, models, and classifications," *ACM Computing Surveys (CSUR),* vol. 39, no. 3, pp. Article 9. 2007.

[68] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed diffusion: A scalable and robust communication paradigm for sensor networks," in *Proceedings of the 6th International Conference on Mobile Computing and Networking,* 06-11 August, Boston, MA, USA, 2000, pp. 56-67.

[69] J. Kulik, W. Heinzelman, and H. Balakrishnan, "Negotiation-based protocols for disseminating information in wireless sensor networks," *Wireless Networks,* vol. 8, no. 2, pp. 169-185. 2002.

[70] S. Kumar, F. Zhao, and D. Shepherd, "Collaborative signal and information processing in microsensor networks," *IEEE Signal Processing Magazine,* vol. 19, pp. 13-14, 2002.

[71] M. Sichitiu and V. Ramadurai, "Localization of wireless sensor networks with a mobile beacon," in *Proceedings of the IEEE International Conference on Mobile Ad Hoc and Sensor Systems — MASS '04,* 25-27 October, Fort Lauderdale, FL, USA, 2004, pp. 174-183.

[72] L. Fang, W. Du, and P. Ning, "A beacon-less location discovery scheme for wireless sensor networks," in *Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies — INFOCOM '05,* 13-17 March, Miami, Florida, USA, vol. 1, 2005, pp. 161-171.

[73] Z. Li, W. Trappe, Y. Zhang, and B. Nath, "Robust statistical methods for securing wireless localization in sensor networks," in *Proceedings of the 4th IEEE International Symposium on Information Processing in Sensor Networks — IPSN '05,* 24-27 April, Los Angeles, CA, USA, 2005, pp. 91-98.

[74] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Transactions of the ASME - Journal of Basic Engineering,* vol. 82, no. 82 (Series D), pp. 35-45. 1960.

[75] C. Hongyang, D. Ping, X. Yongjun, and L. Xiaowei, "A robust location algorithm with biased extended kalman filtering of TDOA data for wireless sensor networks," in *Proceedings of the International Conference on Wireless Communications, Networking and Mobile Computing — WCNM '05,* 23-26 September, Wuhan, China, vol. 2, 2005, pp. 883-886.

[76] E. Olson, J. J. Leonard, and S. Teller, "Robust range-only beacon localization," *IEEE Journal of Oceanic Engineering,* vol. 31, no. 4, pp. 949-958. 2006.

[77] A. Savvides, H. Park, and M. B. Srivastava, "The n-hop multilateration primitive for node localization problems," *Mobile Networks and Applications,* vol. 8, no. 4, pp. 443-451. 2003.

[78] A. Doucet, N. Freitas, and N. Gordon, Eds., *Sequential Monte Carlo Methods in Practice.* New York, NY, USA: Springer Science + Business Media, Inc, 2001.

[79] P. J. Nordlund, F. Gunnarsson, and F. Gustafsson, "Particle filters for positioning in wireless networks," in *Proceedings of the XI European Signal Processing Conference — EUSIPCO '02,* 03-06 September, Toulouse, France, vol. 2, 2002, pp. 311-314.

[80] L. Hu and D. Evans, "Localization for mobile sensor networks," in *Proceedings of the 10th Annual International Conference on Mobile Computing and Networking — MobiCom '04,* 26 September – 01 October, Philadelphia, PA, USA, 2004, pp. 45-57.

[81] F. Gustafsson and F. Gunnarsson, "Positioning using time-difference of arrival measurements," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing — ICASSP '03,* 06-10 April, Hong Kong, China, vol. 6, 2003, pp. 553-556.

[82] F. Gustafsson, F. Gunnarsson, N. Bergman, U. Forssell, J. Jansson, R. Karlsson, and P. J. Nordlund, "Particle filters for positioning, navigation, and tracking," *IEEE Transactions on Signal Processing,* vol. 50, no. 2, pp. 425-437. 2002.

[83] J. Miguez and A. Artes-Rodriguez, "A monte carlo method for joint node location and maneuvering target tracking in a sensor network," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing — ICASSP '06,* 14-19 May, Toulouse, France, vol. 4, 2006, pp. 989-992.

[84] S. W. Smith, *The Scientist, and Engineer's Guide to Digital Signal Processing.* San Diego, CA, USA: California Technical Publishing, 1999.

[85] A. Elfes, "Using occupancy grids for mobile robot perception and navigation," *IEEE Computer,* vol. 22, no. 6, pp. 46-57. 1989.

[86] M. Ribo and A. Pinz, "A comparison of three uncertainty calculi for building sonar-based occupancy grids," *Robotics and Autonomous Systems,* vol. 35, no. 3-4, pp. 201-209. 2001.

[87] C. Wongngamnit and D. Angluin, "Robot localization in a grid," *Information Processing Letters,* vol. 77, no. 5-6, pp. 261-267. 2001.

[88] B. Schiele and J. L. Crowley, "A comparison of position estimation techniques using occupancy grids," *Robotics and Autonomous Systems,* vol. 12, no. 3, pp. 163-172. 1993.

[89] J. Clulow, G. P. Hancke, M. G. Kuhn, and T. Moore, "So near and yet so far: Distance-bounding attacks in wireless networks," in *Security and Privacy in Ad-Hoc and Sensor Networks,* ser. Lecture Notes in Computer Science, L. Buttyan, V. Gligor, and D. Westhoff, Eds., vol. 4357, Berlin, Heidelberg: Springer, 2006, pp. 83-97.

[90] L. Loukas and P. Radha, "HiRLoc: High-resolution robust localization for wireless sensor networks," *IEEE Journal on Selected Areas in Communications,* vol. 24, no. 2, pp. 233-246. 2006.

[91] L. Lazos, R. Poovendran, and S. Čapkun, "ROPE: Robust position estimation in wireless sensor networks," in *Proceedings of the 4th International Symposium on Information Processing in Sensor Network — IPSN '05,* 24-27 April, Los Angeles, California, 2005, pp. 324-331.

[92] S. Capkun and J. P. Hubaux, "Secure positioning of wireless devices with application to sensor networks," in *Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies — INFOCOM '05,* 13-17 March, Miami, Florida, USA, vol. 3, 2005, pp. 1917-1928.

[93] L. Lazos and R. Poovendran, "SeRLoc: Secure range-independent localization for wireless sensor networks," in *Proceedings of the 3rd ACM Workshop on Wireless Security,* 01 October, Philadelphia, PA, USA, 2004, pp. 21-30.

[94] Y. Zeng, S. Zhang, S. Guo, and X. Li, "Secure hop-count based localization in wireless sensor networks," in *Proceedings of the International Conference on Computational Intelligence and Security — CIS '07,* 15-19 December, Harbin, Heilongjiang, China, 2008, pp. 907-911.

[95] A. Srinivasan, J. Teitelbaum, and J. Wu, "DRBTS: Distributed reputation-based beacon trust system," in *Proceedings of the 2nd IEEE International Symposium on Dependable, Autonomic and Secure Computing — DASC '06,* 29 September – 01 October, Indianapolis, IN, USA, 2006, pp. 277-283.

[96] A. Srinivasan, J. Wu, and J. Teitelbaum, "Distributed reputation-based secure localization in sensor networks," *Journal of Autonomic and Trusted Computing,* 2008.

[97] S. Zhong, M. Jadliwala, S. Upadhyaya, and C. Qiao, "Towards a theory of robust localization against malicious beacon nodes," in *Proceedings of the 27th IEEE Conference on Computer Communications — INFOCOM '08,* 13-18 April, Phoenix, AZ, USA, 2008, pp. 1391-1399.

[98] Y. Zeng, J. Cao, S. Zhang, S. Guo, and L. Xie, "Pollution attack: A new attack against localization in wireless sensor networks," in *Proceedings of the IEEE Wireless Communications and Networking Conference — WCNC '09,* 05-08 April, Budapest, Hungary, 2009, pp. 1-6.

[99] S. Misra, G. Xue, and S. Bhardwaj, "Secure and robust localization in a wireless ad hoc environment," *IEEE Transactions on Vehicular Technology,* vol. 58, no. 3, pp. 1480-1489. 2009.

[100] W. Du, L. Fang, and N. Peng, "Lad: Localization anomaly detection for wireless sensor networks," *Journal of Parallel and Distributed Computing,* vol. 66, no. 7, pp. 874-886. 2006.

[101] J. Hwang, T. He, and Y. Kim, "Detecting phantom nodes in wireless sensor networks," in *Proceedings of the 26th IEEE International Conference on Computer Communications — INFOCOM '07,* 06-12 May, Anchorage, AK, USA, 2007, pp. 2391-2395.

[102] Y. Wei, Z. Yu, and Y. Guan, "Location verification algorithms for wireless sensor networks," in *Proceedings of the 27th International Conference on Distributed Computing Systems — ICDCS '07,* 25-27 June, Toronto, Canada, 2007, pp. 70-77.

[103] E. Ekici, S. Vural, J. McNair, and D. Al-Abri, "Secure probabilistic location verification in randomly deployed wireless sensor networks," *Ad Hoc Networks,* vol. 6, no. 2, pp. 195-209. 2008.

[104] P. Papadimitratos, M. Poturalski, P. Schaller, P. Lafourcade, D. Basin, S. Capkun, and J. P. Hubaux, "Secure neighborhood discovery: A fundamental element for mobile ad hoc networking," *IEEE Communications Magazine,* vol. 46, pp. 132-139, 2008.

[105] S. Brands and D. Chaum, "Distance-bounding protocols," in *Advances in Cryptology — EUROCRYPT '93,* ser. Lecture Notes in Computer Science, T. Helleseth, Ed., vol. 765, Berlin, Heidelberg: Springer, 1994, pp. 344-359.

[106] G. P. Hancke and M. G. Kuhn, "An RFID distance bounding protocol," in *Proceedings of the 1st International Conference on Security and Privacy for Emerging Areas in Communication Networks — SecureComm '05,* 05-09 September, Athens, Greece, 2005, pp. 67-73.

[107] J. Munilla and A. Peinado, "Distance bounding protocols for RFID enhanced by using void-challenges and analysis in noisy channels," *Wireless Communications and Mobile Computing,* vol. 8, no. 9, pp. 1227-1232. 2008.

[108] S. Čapkun, L. Buttyán, and J. P. Hubaux, "SECTOR: Secure tracking of node encounters in multi-hop wireless networks," in *Proceedings of the 1st ACM Workshop on Security of Ad-Hoc and Sensor Network — SASN '03,* 31 October, Fairfax, Virginia, USA, 2003, pp. 21-32.

[109] Y. Desmedt, "Major security problems with the "unforgeable" (feige)-Fiat_Shamir proofs of identity and how to overcome them," in *Proceedings of the 6th Worldwide Congress on Computer and Communications Security and Protection — SecuriCom '08,* 15-17 March, 1988, pp. 147-159.

[110] Y. C. Hu, A. Perrig, and D. B. Johnson, "Rushing attacks and defense in wireless ad hoc network routing protocols," in *Proceedings of the 2nd ACM Workshop on Wireless Security — WiSec '03,* 19 September, San Diego, CA, USA, 2003, pp. 30-40.

[111] C. Tang, D. O. Wu, T. Syst, and L. Oswego, "Distance-bounding based defense against relay attacks in wireless networks," *IEEE Transactions on Wireless Communications,* vol. 6, no. 11, pp. 4071-4078. 2007.

[112] B. Waters and E. Felten, "Secure, private proofs of location," Princeton University, Tech. Rep. TR-667-03, 2003 .

[113] V. Nikov and M. Vauclair, "Yet another secure distance-bounding protocol," Cryptology ePrint Archive, Tech. Rep. 319, July 23 2008 .

[114] G. P. Hancke and M. G. Kuhn, "Attacks on time-of-flight distance bounding channels," in *Proceedings of the 1st ACM Conference on Wireless Network Security — WiSec '08,* 31 March – 02 April, Alexandria, VA, USA, 2008, pp. 194-202.

[115] D. Singelée and B. Preneel, "Distance bounding in noisy environments," in *Security and Privacy in Ad-Hoc and Sensor Networks,* ser. Lecture Notes in Computer Science, F. Stajano, C. Meadows, S. Capkun, and T. Moore, Eds., vol. 4572, Berlin, Heidelberg: Springer, 2007, pp. 101-115.

[116] J. Munilla and A. Peinado, "Attacks on a distance bounding protocol," *Computer Communications,* vol. 33, no. 7, pp. 884-889. 2010.

[117] J. Hightower and G. Borriello, "Location systems for ubiquitous computing," *IEEE Computer,* vol. 34, pp. 57-66, 2001.

[118] N. O. Tippenhauer and S. Capkun, "ID-based secure distance bounding and localization," in *Computer Security – ESORICS '09,* ser. Lecture Notes in Computer Science, M. Backes and P. Ning, Eds., vol. 5789, Berlin, Heidelberg: Springer, 2009, pp. 621-636.

[119] M. Flury, M. Poturalski, P. Papadimitratos, J. P. Hubaux, and J. Y. Le Boudec, "Effectiveness of distance-decreasing attacks against impulse radio ranging," in *Proceedings of the 3rd ACM Conference on Wireless Network Security — WiSec '10,* 22–24 March, Hoboken, New Jersey, 2010, pp. 117-128.

[120] L. Bussard and W. Bagga, "Distance-bounding proof of knowledge to avoid real-time attacks," in *Security and Privacy in the Age of Ubiquitous Computing,* ser. IFIP Advances in Information and Communication Technology, R. Sasaki, S. Qing, E.

Okamoto, and H. Yoshiura, Eds., vol. 181, Berlin, Heidelberg: Springer, 2005, pp. 223-238.

[121] J. Pieprzyk, T. Hardjono, and J. Seberry, *Fundamentals of Computer Security.* Berlin, Heidelberg: Springer, 2003.

[122] J. Reid, J. M. G. Nieto, T. Tang, and B. Senadji, "Detecting relay attacks with timing-based protocols," in *Proceedings of the 2nd ACM Symposium on Information, Computer and Communications Security,* 20-22 March, Singapore, 2007, pp. 204-213.

[123] Y. J. Tu and S. Piramuthu, "RFID distance bounding protocols," in *Proceedings of the 1st International EURASIP Workshop on RFID Technology,* 24-25 September, Vienna, Austria, 2007.

[124] C. H. Kim, G. Avoine, F. Koeune, F. X. Standaert, and O. Pereira, "The swiss-knife RFID distance bounding protocol," in *Information Security and Cryptology – ICISC 2008,* ser. Lecture Notes in Computer Science, P. J. Lee and J. H. Cheon, Eds., vol. 5461, Berlin, Heidelberg: Springer, 2009, pp. 98-115.

[125] C. H. Kim and G. Avoine, "RFID distance bounding protocol with mixed challenges to prevent relay attacks," in *Cryptology and Network Security — CANS '09,* ser. Lecture Note in Computer Science, J. A. Garay, A. Miyaji, and A. Otsuka, Eds., vol. 5888, Berlin, Heidelberg: Springer, 2009, pp. 119-133.

[126] M. Bellare and P. Rogaway, "Entity authentication and key distribution," in *Advances in Cryptology — CRYPTO '93,* ser. Lecture Notes in Computer Science, D. R. Stinson, Ed., vol. 773, Berlin, Heidelberg: Springer, 1994, pp. 232-249.

[127] J. D. Guttmana, F. J. Thayera, and L. D. Zuckb, "The faithfulness of abstract protocol analysis: Message authentication," *Journal of Computer Security,* vol. 12, no. 6, pp. 865-891. 2004.

[128] P. Peris-Lopez, J. C. Hernandez-Castro, J. M. E. Tapiador, and J. C. A. van der Lubbe, "Shedding some light on RFID distance bounding protocols and terrorist attacks," arXiv.org, Tech. Rep. arXiv:0906.4618v2, June 20 2010 .

[129] G. Avoine, C. Floerkemeier, and B. Martin, "RFID distance bounding multistate enhancement," in *Progress in Cryptology — INDOCRYPT '09,* ser. Lecture Notes in Computer Science, B. Roy and N. Sendrier, Eds., vol. 5922, Berlin, Heidelberg: Springer, 2009, pp. 290-307.

[130] G. Avoine and A. Tchamkerten, "An efficient distance bounding RFID authentication protocol: Balancing false-acceptance rate and memory requirement," in *Information Security,* ser. Lecture Notes in Computer Science, P. Samarati, M. Yung, F. Martinelli, and C. A. Ardagna, Eds., vol. 5735, Berlin, Heidelberg: Springer, 2009, pp. 250-261.

[131] R. Trujillo-Rasua, B. Martin, and G. Avoine, "The poulidor distance-bounding protocol," in *The 6th Workshop on RFID Security — RFIDsec '10,* 07-09 June, Istanbul, Turkey, 2010.

[132] E. Barker and A. Roginsky, "Recommendation for the transitioning of cryptographic algorithms and key sizes," CiteSeerx, USA, Tech. Rep. NIST Special Publication 800-131, January 2010 .

[133] O. Kara, S. Kardas, M. A. Bingol, and G. Avoine, "Optimal security limits of RFID distance bounding protocols," in *The 6th Workshop on RFID Security — RFIDSec '10,* 07-09 June, Istanbul, Turkey, 2010.

[134] G. P. Hancke, K. E. Mayes, and K. Markantonakis, "Confidence in smart token proximity: Relay attacks revisited," *Computers & Security,* vol. 28, no. 7, pp. 615-627. 2009.

[135] S. Drimer and S. J. Murdoch, "Keep your enemies close: Distance bounding against smartcard relay attacks," in *Proceedings of 16th USENIX Security Symposium,* 06–10 August, Boston, MA, 2007, pp. 87-102.