# Indoor Localization in Wireless Sensor Networks


*Author* : Martin van de Goor

*Supervisor* : Dr. David N. Jansen



*Date* : March 20, 2009

*Thesis number* : 600

# Acknowledgments

First, I would like to thank David Jansen for his valuable feedback and suggestions. With his knowledge on various subjects our discussions were always interesting, whether it concerned directional antennas, statistics or the German language. My gratitude also goes out to Martijn Vlietstra, whose quick and timely help throughout the project ensured I was able to continue my work at all times. With his unending enthusiasm, it has always been a pleasure working with him. Co Kooijman and Frits Vaandrager have provided me with useful suggestions and remarks on how to improve my thesis in various ways and for that I wish to thank them. Thanks also go to Serhat Gülçiçek, who has helped me understanding the architecture and workings of the previous WSN project.

Further, I want to express my appreciation towards the scientific community. Help from TinyOS developers on how to solve a tough problem on serial communications has been much appreciated. Gratitude is also expressed to all the authors who kindly gave their permission to use their figures, and to Michael Thomas Flanagan, who has written a comprehensive scientific and numerical library in Java.

And last but definitely not least, I would like to thank my parents Ger and Wies and sister Marcia for their continuous support – in every sense of the word – during my studies.

# Contact Details


**Examinee**

| | |
|---|---|
| Name | : Martin van de Goor |
| E-mail | : martin.vandegoor@gmail.com |
| Telephone | : 06 4867 4233 |

**University Supervisor**

| | |
|---|---|
| Name | : Dr. David N. Jansen |
| E-mail | : D.Jansen@science.ru.nl |
| Telephone | : 024 365 2271 |

**Company Information**

| | |
|---|---|
| Name | : Logica |
| Address | : Meander 901<br>Postbus 7015<br>6801 HA Arnhem |

**Company Supervisors**

| | |
|---|---|
| Name | : Martijn Vlietstra |
| E-mail | : martijn.vlietstra@logica.com |
| Telephone | : 026 376 5472 |
| | |
| Name | : Co Kooijman |
| E-mail | : co.kooijman@logica.com |
| Telephone | : 026 376 5411 |

# Contents

# Chapter 1

# Introduction

In this chapter, we will introduce the notion of a wireless sensor network, describe the problem context and give the problem statement. Then, the relevance of solving this problem will be explained. Last, we explain terminology used in the context of wireless sensor networks.

## 1.1 Wireless Sensor Networks

A Wireless Sensor Network (WSN) is a network of many small sensing and communicating devices called sensor nodes (or motes). Each node has a CPU, a power supply and a radio transceiver for communication. Interconnection between nodes is achieved via the transceiver. Typically, a WSN contains one node, the *base station*, that connects the network to a more capable computer (Figure 1.1), and probably to a network of general purpose computers through it. Sensors attached to these nodes allow them to sense various phenomena within the environment. The typical purpose of a sensor network is to collect data via sensing interfaces and propagate those data to the central computer, allowing easy monitoring of an environment.

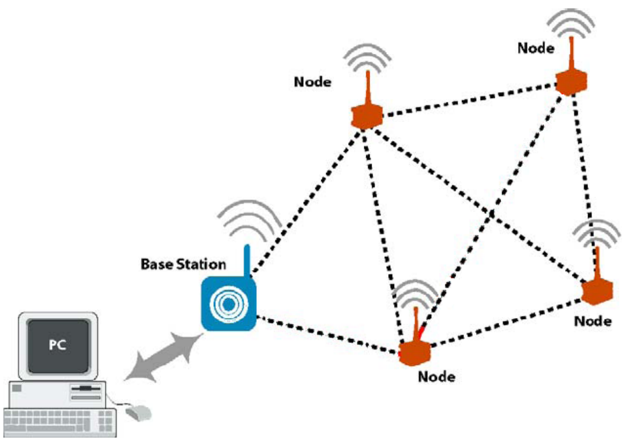



Figure 1.1: Example of a Wireless Sensor Network.

Although a node is capable of dealing with a variety of jobs, it has many shortcomings as well. The majority of the nodes currently available in the market are battery-operated, and hence they have a limited life-time. Moreover, the memory capacity of a node is also limited. Life-time, processing and storage restrictions directly affect the algorithms designed for sensor networks. As an example, a routing algorithm for WSNs must be energy and memory efficient. Since radio transmissions consume a significant amount of energy, researchers generally seek ways to reduce radio communication as much as possible. However, when more information is stored and more computation is done as to reduce the communication costs, energy consumption of the processor and memory components are becoming an important issue. Design choices have to be made, and these also depend on the intended application.

## 1.2 Problem Context

Currently there is a prototype of a system available, developed within Logica's Working Tomorrow program[1], which uses motion sensors to secure an area [34] based on the Smart Dust concept. The idea of the system is to monitor an area or room by a network of sensors with the size of a dust particle. To be more precise, the Smart Dust project is 'exploring whether an autonomous sensing, computing, and communication system can be packed into a cubic-millimeter mote (a small particle or speck) to form the basis of integrated, massively distributed sensor networks' [42]. In the prototype the size of a sensor is significantly bigger than a dust particle. The moment a sensor detects movement in the area a message is sent to a central server. The server processes the data and then uses Google Maps to produce a map which shows the detected movement. A GPS receiver is used to determine an absolute position, while RSSI (Received Signal Strength Indicator) is used to locate the sensors relative to the GPS receiver. RSSI uses the decrease in energy of the radio signal as it propagates in space to estimate the distance [7]. Experimentation with the prototype system shows this method becomes unreliable when the batteries of the sensors are getting weaker [34]. Simply using GPS receivers for all sensors is not an option as GPS cannot function in indoor and many outdoor applications, especially when there is no direct line of sight from nodes to terrestrial satellites. Besides, the use of these devices on sensor nodes is still a challenging issue due to their size, energy and price constraints [4]. As a result, there is a need for reliable localization in WSNs without the use of GPS receivers.

## 1.3 Problem Statement

The question which follows from the problem context is: How can we do localization in WSNs without GPS? We will focus our research on algorithms suitable for mobile indoor networks. These algorithms will be compared with each other, based on a literature study. The goal is to develop a prototype in which localization is reliable and which can be used in a convincing demonstration. For the purpose of a demonstration it is preferred that the deployment is ad-hoc and little configuration or calibration is required. The research questions reflect the twofold approach:

- Which systems and algorithms exist for reliable localization in mobile indoor wireless sensor networks that use a minimal number of beacon nodes and how do they compare?
- Can we develop a prototype by implementing such an algorithm or an improvement thereof based on an evaluation of algorithms?

## 1.4 Motivation

Usually, a Wireless Sensor Network is deployed to monitor its environment and for disaster response and recovery systems. Applications include health monitoring systems, monitoring of wildlife habitats [27] and nature reserves such as the Great Barrier Reef [21], and forest fire detection systems [11, 17, 24]. Examples of military applications are battlefield surveillance [5, 18] and the previously mentioned securing of an area or room.

Our focus, however, lies on localization in mobile indoor WSNs. Localization can be used for tracking objects or people. For example, our research may help people navigating indoors where

[1] Working Tomorrow is Logica's graduate program that focuses on the feasibility and opportunities of innovative ICT solutions.

GPS is not available. Also, mobile devices such as laptops may be tracked within a building in order to locate them easily. Location-dependent network services, with application examples ranging from building automation to targeted advertising or augmented reality, also require reliable localization techniques [23].

Localization in WSNs is also of use for context aware applications in Wireless Personal Networks (WPNs) [1]. In a WPN the user and all his devices are constantly and securely connected and the services/applications are adapted so that the sessions are transported seamlessly (without the user's intervention) depending on the context or situation. The context-aware solutions try to exploit information regarding the geographical location, the time of the day, available equipment, history of user's interaction/usage, environmental changes and the presence of other people. They provide the user with the service which is best suited to the person's present situation.

A good example of such a context aware application is the Smart Signs system. Smart Signs are a new type of electronic door- and way-signs based on wireless sensor networks [25]. The system uses context information such as user's mobility limitations, the weather, and possible emergency situations to improve guidance and messaging. For example, it can adapt the route if it suddenly starts raining. One of the important inputs for a context aware application is the knowledge of the physical location of the person, where localization in WSNs comes in.

## 1.5 Terminology

The first part of this section introduces terminology related to algorithms used in the context of localization in WSNs. The second part provides some background knowledge in the field of wireless communication via electromagnetic waves.

### 1.5.1 Localization

Localization algorithms can be categorized according to a number of different aspects [4, 35, 43]:

- *Input data*: range-free vs. range-based
  *Range-free* localization algorithms simply rely on connectivity information (whether nodes can hear each other or not and radio-range information). *Range-based* methods extract distance information from radio signals.

- *Accuracy*: fine-grained vs. coarse-grained
  A location discovery algorithm should estimate sensor position accurately. *Accuracy*, or grain size, can be expressed as percentage of sensor transmission range, or simply in meters. The level of accuracy usually depends on range measurement errors. Range measurements with less error will lead to more accurate position estimates. How often we can expect a certain accuracy is the *precision*, which is expressed in a percentage. For example, some inexpensive GPS receivers can locate positions to within 10 meters for approximately 95 percent of measurements. More expensive units usually do much better, reaching 1- to 3-meter accuracies 99 percent of the time. The distances denote accuracy, the percentages precision. If we can live with less accuracy, we may be able to trade it for increased precision [19].

- *Dynamics*: mobile vs. fixed
  In fixed networks, nodes can establish their location in the initialization phase. Thereafter, their only task is to report events or relay information sent by other nodes. In mobile networks, however, nodes need to be aware of changes in their position and perhaps of position changes of other nodes. In general, systems provide more accurate location information when a node is at rest than when it is in motion: tracking a moving node is harder because the inevitable errors that occur in the distance samples are easier to filter out if the node's position itself does not change during the averaging process [37].

- *Beacons*: beacon-free vs. beacon-based
  Nodes with known positions are called *beacon* or *anchor* nodes. Beacon-based algorithms usually produce an absolute location system where *absolute positions* of nodes are known, for example, latitude, longitude, and altitude. However, the accuracy of the estimated position is highly affected by the number of anchor nodes and their distribution in the sensor field. The ratio of beacon nodes to *blind* nodes (nodes with unknown positions) is generally quite small. The location of a beacon node can be determined using an attached GPS device or by manual deployment.

  Beacon-free algorithms do not make any assumptions regarding node positions. In this case, instead of computing absolute node positions, *relative positioning* is used in which the coordinate system is established by a reference group of nodes. Each object can also have its own frame of reference [19]. For example, a mountain rescue team searching for avalanche victims can use handheld computers to locate victims' avalanche transceivers. Each rescuer's device reports the victims' positions relative to itself.

- *Computational model*: centralized vs. distributed
  If an algorithm collects localization related data from the network and processes the data collectively at a single station, then it is said to be *centralized*. If, on the other hand, each node collects partial data relevant to it and executes an algorithm to locate itself, then the localization algorithm is categorized as *distributed*. An intermediate form are so called *locally centralized* algorithms, which are distributed algorithms that achieve a global goal by communicating with nodes in some neighborhood only. For example, the sensor network can be divided into local clusters, where each cluster has a head. All the range measurements in a certain cluster are forwarded to the cluster head, where computation takes place.

- *Hops*: single-hop vs. multi-hop
  A direct link between two neighbor nodes is called a *hop*. When the distance between two nodes is larger than the radio range but there are other nodes that create a continuous path between them, the path is called a *multi-hop* path.

### 1.5.2 Wireless Communication

As sensor nodes use electromagnetic waves to communicate with each other we need to understand the basics of how these waves propagate. Basic signal propagation and multipath propagation are discussed.

#### Signal Propagation

A signal emitted by an antenna travels in the following three types of propagation modes: ground-wave propagation, sky-wave propagation, and line-of-sight (LOS) propagation. MW and LW radio is a kind of ground-wave propagation, where signals follow the contour of the Earth. Shortwave radio is an example of sky-wave propagation, where radio signals are reflected by ionosphere and the ground along the way. Beyond 30 MHz, line-of-sight propagation dominates, meaning that signal waves propagate on a direct, straight path in the air. Radio signals of line-of-sight propagation can also penetrate objects, especially signals with frequencies just above 30 MHz [44].

Sensor motes support tunable frequencies in the range of 300 to 1000 MHz and the 2.4-GHz band. This means LOS propagation is dominant. The industrial, scientific and medical (ISM) radio bands were originally reserved internationally for the use of RF electromagnetic fields for industrial, scientific and medical purposes other than communications. They have become a part of the radio spectrum that can be used by anybody without a license in most countries.

#### Multipath Propagation

For visible light we are well aware of the following effects: shadowing, reflection and refraction. In general, electromagnetic waves (including light) are also subject to diffraction and scattering [44].

Radio communication is affected by the physical properties of waves; the combined effects may cause a transmitted radio signal to reach a receiver by two or more paths. This is called *multipath propagation* and is illustrated in Figure 1.2.

- *Shadowing* and *reflection* occur when a signal encounters an object that is much larger than its wavelength. Though the reflected signal and the shadowed signal are comparatively weak, they in effect help to propagate the signal to spaces where line-of-sight is impossible [44]. Reflections occur from the surface of the earth and from buildings and walls.

- *Refraction* occurs when a wave passes across the boundary of two media [44]. Compare this to how sunlight refracts when it enters water.

- *Diffraction* occurs at the edge of an impenetrable body that is large compared to the wavelength of the radio wave. When a radio wave encounters such an edge, waves propagate in different directions with the edge as the source [38]. Thus, signals can be received even when there is no line-of-sight path between transmitter and receiver. For example, a wave can 'bend' around a corner due to this effect.

- *Scattering* occurs when the medium through which the wave travels consists of objects with dimensions that are small compared to the wavelength, and where the number of obstacles per unit volume is large. Scattered waves are produced by rough surfaces, small objects, or by other irregularities in the channel [36]. Typical objects that induce scattering are foliage, street signs, and lamp posts.

If there is line-of-sight between receiver and transmitter, then diffraction and scattering are generally minor effects, although reflection may have a significant impact. If there is no clear LOS, such as in an urban area at street level, then diffraction and scattering are the primary means of signal reception [38].

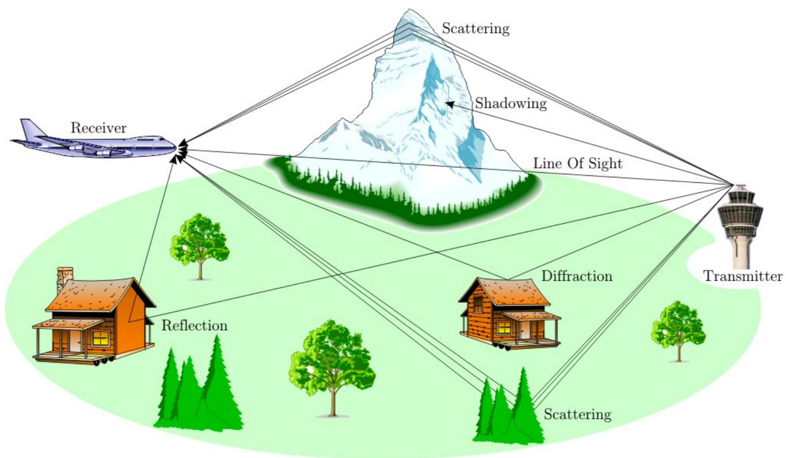



Figure 1.2: Multipath propagation: various effects give rise to additional radio propagation paths beyond the direct optical line-of-sight path between the transmitter and receiver. Image courtesy of Haas [16].

# Chapter 2

# Localization Methods

Triangulation, scene analysis, and proximity are the three principal techniques for automatic location-sensing [19]. Location systems may employ them individually or in combination. The triangulation location-sensing technique uses the geometric properties of triangles to compute object locations. Triangulation is divisible into the subcategories of lateration, using distance measurements, and angulation, using primarily angle or bearing measurements. Scene analysis observes features of its surroundings in order to determine the location of an object. In localization based on proximity, an object's presence is sensed using a physical phenomenon with limited range, for example infrared or direct contact. We will cover lateration, angulation, and scene analysis in more detail.

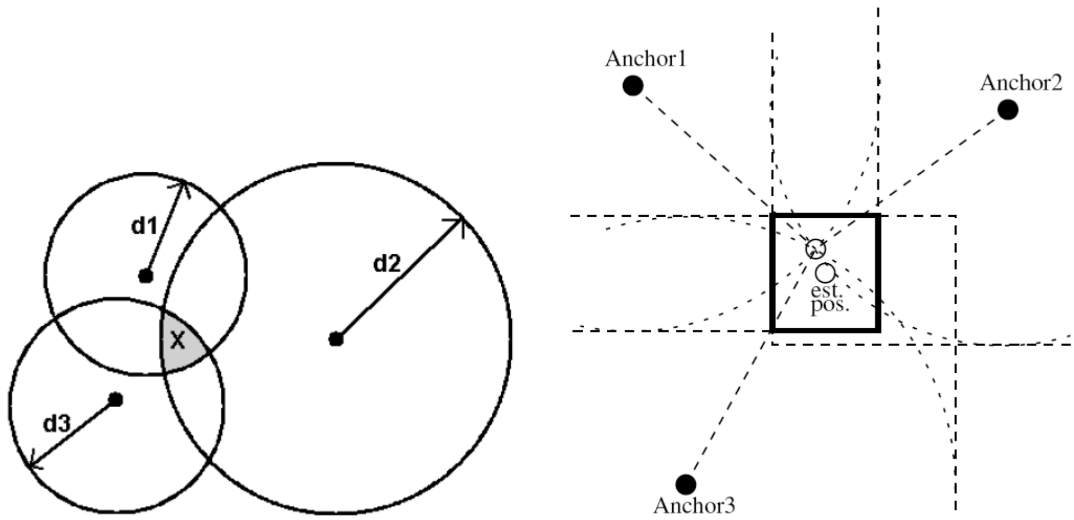
## 2.1 Lateration

*Lateration* computes the position of an object by measuring its distance from multiple reference positions [19]. Calculating an object's position in two dimensions requires distance measurements from 3 points that do not all lie on a single line (*non-collinear* points). In three dimensions, distance measurements from 4 points not lying in the same plane (*non-coplanar* points) are required. Domain-specific knowledge may reduce the number of required distance measurements (e.g., in GPS, one computed position is in outer space).

The 2D lateration technique works well when the three circles intersect at a single point, but this is rarely the case when estimates are used in ranging. When the range of anchor nodes is sufficiently large, the object to be located falls into a geometric region that is the intersection of three circles. This is called *bounded intersection* by Terwilliger [41] and is illustrated in Figure 2.1a. It is also possible that the region of intersection is empty. This will occur if at least one ranging estimate is too small. Maximum likelihood methods overcome this problem by selecting the point for localization that gives the minimum total error between measured estimates and distances.

Lateration is quite expensive in the number of floating point operations that is required [22]. A similar, but computationally less expensive solution is to use a *bounding box* approach. The main idea is to construct a bounding box for each anchor using its position and distance estimate, and then to determine the intersection of these boxes. The position of the node is estimated to be the center of the intersection box. Figure 2.1b illustrates the bounding box method for a node with distance estimates to three anchors. Note that, in this example, the estimated position by the bounding box is close to the true position computed through lateration.

We will discuss two general approaches to measuring the distances (called *ranging*) required by the lateration technique, being attenuation and time-of-flight.




(a) The location of 'X' is computed by taking the center of the intersection of the three circles.

(b) Intersecting boxes requires less computation than intersecting circles, but is somewhat less accurate.

Figure 2.1: Bounding the location of a node

### 2.1.1 Attenuation

The intensity of an emitted signal decreases as the distance from the emission source increases. The decrease relative to the original intensity is the *attenuation* [19]. The signal strength decays polynomially with respect to distance. In the most ideal circumstances (in vacuum), signal power attenuation is proportional to $d^2$, where $d$ denotes the distance between the transmitter and the receiver. This effect is sometimes referred to as *free space loss* [44].

Using Received Signal Strength Indicator (RSSI) is one of the most commonly studied approaches for localization purposes because almost every node in the market has the ability to analyze the strength of a received message [7]. Given a function correlating attenuation and distance for a type of emission and the original strength of the emission, it is possible to estimate the distance from an object to some point P by measuring the strength of the emission when it reaches P. The widely used radio propagation model, the log-distance path loss model, considers the received power as a function of the transmitter-receiver distance raised to some power. Since this model is a deterministic propagation model and gives only the average value, another propagation model, the log-normal shadowing model, is introduced to describe the RSSI irregularity [39]:

$$RSSI(d)[dBm] = RSSI_{ref} - 10n\log_{10}\left(\frac{d}{d_{ref}}\right) + X_\sigma \tag{2.1}$$

In Equation 2.1, $d$ is the transmitter-receiver distance, $n$ the attenuation constant (rate at which the signal decays), $X_\sigma$ a zero-mean Gaussian (in dB) with standard deviation $\sigma$ (multipath effects), and $RSSI_{ref}$ the signal strength value at reference distance $d_{ref}$. Usually, $n$ and $\sigma$ are obtained through curve fitting of empirical data. RSSI is measured in dBm, which is a logarithmic measurement of signal strength. Note that the RSSI value does not only depend on the distance, but also on the environment, antenna orientation, and the power supply [1].

A commonly used model for calculating the distance $d$ is given in Equation 2.2, in which $RSSI_{ref}$ is measured at $d_{ref} = 1$ m. It is based on Equation 2.1, but multipath effects are omitted ($X_\sigma$ is assumed to be 0 with probability 1).

$$d(RSSI) = 10^{\frac{RSSI_{ref} - RSSI}{10n}} \tag{2.2}$$

In this scheme the attenuation constant is around 2 in an open-space environment, but its value increases if the environment is more complex (walls, large metallic objects, etc.). In environments with many obstructions such as an indoor office space, measuring distance using attenuation is usually less accurate than time-of-flight [19]. An approximation of the attenuation constant for an indoor environment is around 3.5 [36]. There is empirical evidence [12] that due to the unreliability of measurements, at best, accuracy in the scale of meters can be achieved regardless of the used algorithm or approach.

In the localization system Ferret, described by Terwilliger [41], two different ranging techniques (potentiometer and RSSI) are used to help locate an object to within one meter. In the potentiometer technique, the object to be located (a mobile node) begins by transmitting the beacon at the lowest power level and listens for replies from the infrastructure nodes. Increasing the power level with each transmission, once the mobile node gets three replies, it forwards its data to the base station for position computation. A calibration tool needs to be run each time the system is moved to a new environment in order to establish the communication ranges for given transmission power levels. Terwilliger also presents a location discovery algorithm that provides, for every node in the network, a position estimate, as well as an associated error bound and confidence level.

### 2.1.2 Time-of-Flight

Measuring distance from an object to some point P using time-of-flight means measuring the time it takes to travel between the object and point P at a known velocity. The object itself may be moving, such as an airplane traveling at a known velocity for a given time interval, or, as is far more typical, the object is approximately stationary and we are instead observing the difference in transmission and arrival time of an emitted signal [19]. GPS is a well-known system which uses the time-of-flight technique.

There are two main issues in using time-of-flight. The first issue is to distinguish direct pulses from reflected ones because they look identical. Reflected measurements may be pruned away by aggregating multiple receivers' measurements and observing the environment's reflective properties. The second issue is agreement about the time. Since the propagation speed of radio signals is very high (being equal to the speed of light), time measurements must be very accurate in order to avoid large uncertainties. For example, a localization accuracy of 1 meter requires timing accuracy on the level of $\frac{1}{3*10^8} \approx 3.3$ nanoseconds. This means a minimum clock rate of 300 MHz ($3 * 10^8$ Hz) is required for hardware. As far as time synchronization goes, state-of-the-art protocols such as FTSP [29] 'only' synchronize nodes in the order of microseconds. To avoid this issue, a node could reflect the radio signal back, but this once again requires constant delay for reflecting the signal.

One can also measure the time difference of arrival. Cricket [33, 37], a location-support system for in-building, mobile, location-dependent applications, uses concurrent radio and ultrasound signals and measures the difference between the received times of the two types of signals. As sound waves travel at the speed of sound less precise timing than in the case of RF time-of-flight is required. A difference with radio signals is that an ultrasound signal does not go through walls; a similarity is that ultrasonic reception also suffers from severe multipath effects caused by reflections from walls and other objects. Cricket allows applications running on mobile and static nodes to learn their physical location by using listeners that hear and analyze information from beacons spread throughout a building. A case distinction is made for various situations in order to overcome multipath and interference effects. Practical beacon configuration and positioning techniques are used to improve accuracy up to the centimeter level.

## 2.2 Angulation

*Angulation* is similar to lateration except, instead of distances, angles are used for determining the position of an object. This technique is also called angle-of-arrival. In general, two-dimensional angulation requires two angle measurements and one length measurement such as the distance between the reference points as shown in Figure 2.2. In three dimensions, one length measurement, one azimuth measurement, and two angle measurements are needed to specify a precise position [19]. Although the definition of azimuth depends on the coordinate system, in this case, the azimuth is the horizontal component of an angle, measured around the horizon, from the north toward the east. Angulation implementations sometimes choose to designate a constant reference vector (e.g., magnetic north) as 0°.

All of the proposed solutions require special hardware (and are thus costly solutions). In general phased antenna arrays are used to measure the angle. Antenna arrays consist of multiple antennas with known separation in which each antenna measures the time of arrival of a signal. Given the differences in arrival times and the geometry of the receiving array, it is then possible to compute the angle from which the emission originated. If there are enough elements in the array and large enough separations, the angulation calculation can be performed [19]. Other approaches described in literature (see Basaran [4]) are compass sensors, rotating antennas, and rotating light emitters combined with optical sensors.

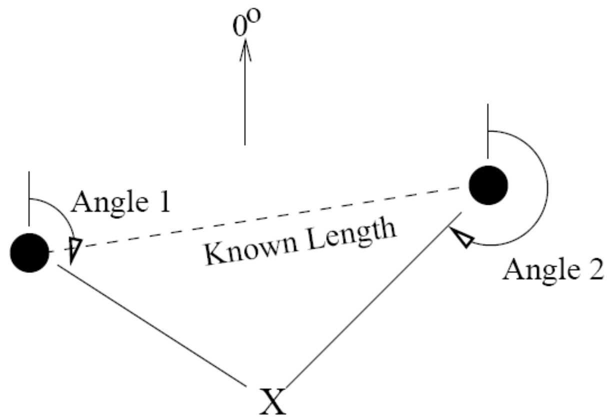



Figure 2.2: This example of 2D angulation illustrates locating object 'X' using angles relative to a 0° reference vector and the distance between two reference points. 2D angulation always requires at least two angle and one distance measurement to unambiguously locate an object [19].

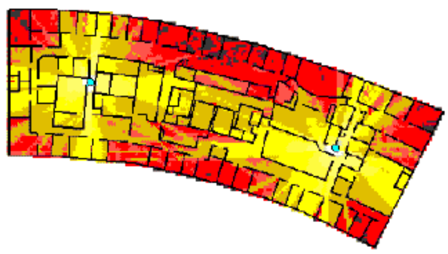


Figure 2.3: Example of a probabilistic radio map. Given this map and signal strength measurements to fixed nodes, the likeliest location of the user can be predicted.

## 2.3 Scene Analysis

In general, the scene analysis location-sensing technique uses features of a scene observed from a particular vantage point to draw conclusions about the location of the observer or of objects in the scene [19]. In WSNs the measured feature of the scene is typically the signal strength value at a particular position and orientation. Scene analysis consists of an offline learning phase and an online localization phase. During the offline phase RSSI values to different anchor nodes are recorded at various positions. The recorded RSSI values and the known locations of the anchor nodes are used either to construct an RF-fingerprint database, or a probabilistic radio map (Figure 2.3). In the online phase, the node to be localized measures RSSI values to different anchor nodes. With RF-fingerprinting, the location of the user is determined by finding the recorded reference fingerprint values that are closest to the measured one. The unknown location is then estimated to be the one paired with the closest reference fingerprint or in the (weighted) centroid of k-nearest reference fingerprints. Location estimation using a probabilistic radio map includes finding the point(s) in the map that maximize the location probability [20].

The Microsoft Research RADAR location system is an example of RF-fingerprinting. RADAR uses a dataset of signal strength measurements created by observing the radio transmissions of an 802.11 wireless networking device at many positions and orientations throughout a building. The location of other 802.11 network devices can then be computed by performing table lookup on the prebuilt dataset. The median resolution of RADAR is in the range of 2 to 3 meters [3].

MoteTrack [26] extends the approach and claims to be more robust than RADAR. Still, base stations at fixed locations are used and a form of fingerprinting is used for determining the location of mobile nodes. However, the approach can tolerate the failure of up to 60% of the beacon nodes without severely degrading accuracy. Moreover, it is resilient to information loss, it can cope with perturbations in RF signals (which may be caused by changes in the environment, e.g., collapsed walls in a disaster scenario), and is decentralized to prevent single point of failure.

Although fingerprinting can give accurate results, it is not appropriate for scenarios where offline calibration is infeasible (for example, if the area is hard to access). Furthermore, collecting all the RSSI samples is quite time-consuming.

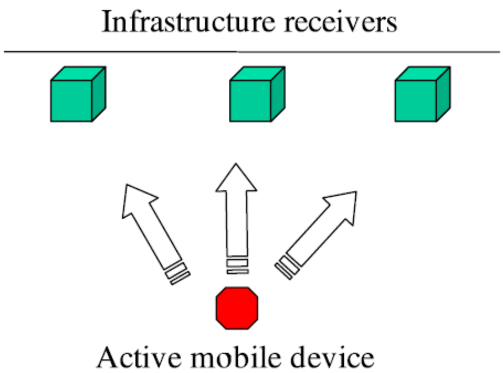
# Chapter 3

# Related Work

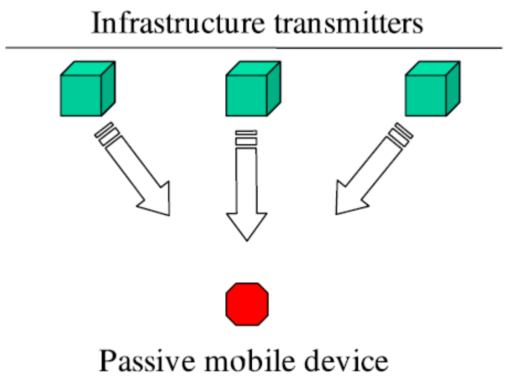
This chapter is devoted to related work in mobile indoor localization. All the discussed approaches are range-based, because the accuracy of range-free algorithms is often limited by requiring dense deployments of sensor nodes [23].

## 3.1 Cricket

The tracking of moving devices has been studied by Smith et al. [37] under an active mobile and a passive mobile infrastructure (Figure 3.1) using the Cricket location system (already briefly described in section 2.1.2). Cricket uses the time difference in arrival of concurrent radio and ultrasound signals to estimate distances. In the active mobile architecture, the mobile device actively chirps, and the fixed infrastructure nodes then reply either over a radio channel or a cabled infrastructure, reporting the measured distances to the mobile device or some central processor. In the passive variant, the infrastructure has beacons that periodically transmit signals to a passively listening mobile device, which in turn estimates distances to the beacons.




(a) In an active mobile architecture, an active transmitter on each mobile device periodically broadcasts a message on a wireless channel.




(b) In a passive mobile architecture, fixed nodes at known positions periodically transmit their location (or identity) on a wireless channel, and passive receivers on mobile devices listen to each beacon.

Figure 3.1: Infrastructure types for locating mobile devices.

Because in the active mobile architecture fixed nodes receive simultaneous distance estimates from the mobile device, it performs better at tracking than the passive mobile system in which the device obtains only one distance estimate at a time and may have moved between successive estimates. However, a passive mobile system scales better with the number of mobile devices and puts users in control of whether their whereabouts are tracked. The authors devise a hybrid approach that tries to preserve the benefits of both approaches. During normal operation the passive mobile system is used due to its scalability and guaranteed user-privacy. At start-up time, and when the system gets in a bad state and needs to be restarted, the listener transitions to active mobile operation to obtain multiple simultaneous beacon distance samples. In an experimental setup, a moving node was tracked in a single room. Six different speeds up to 1.43 m/s were tested. The accuracy is high in general but decreases somewhat as the speed increases.

Priyantha et al. [32] note it is almost impossible to deploy nodes in a typical office or home to achieve sufficient connectivity across all nearby nodes. For example, it is hard to obtain ranging between nodes placed inside and outside a room in a standard building. Due to the directionality of the ultrasonic transmitters used, the ultrasonic-based ranging system has a 12 m range when the transmitter and the receiver are facing each other but less than 2 m mutual range when they are on the same horizontal plane facing away from the plane (e.g., downwards from a ceiling).

## 3.2 Self-Positioning Algorithm

Čapkun et al. introduce the Self-Positioning Algorithm (SPA) [6]. SPA defines and computes relative positions of nodes in a mobile ad-hoc network without using GPS. It is a distributed algorithm that does not use nodes with fixed or known positions. It assumes some method to estimate the distances between nodes and builds a relative coordinate system.

As a first step, each node builds a local coordinate system which has the node as its center. Node $i$ defines its local coordinate system by choosing nodes $p$ and $q$ such that the distance between $p$ and $q$ ($d_{pq}$) is known and larger than zero and such that nodes $i$, $p$, and $q$ do not lie on the same line. The system is defined to have $p$ lying on the positive x axis and $q$ having a positive y coordinate (Figure 3.2). The real-world directions of $p$ and $q$ are not needed because a relative coordinate system is constructed; this system would have to be rotated and maybe reflected afterwards to correspond with physical node locations. The authors do not specify how non-collinear nodes are picked, but one could ensure a triangle is formed by choosing $p$ and $q$ such that, given distances $d_{pq}$, $d_{iq}$, and $d_{ip}$, the maximum distance is not equal to the sum of the two remaining distances. Furthermore, the choice of $p$ and $q$ should maximize the number of the nodes for which the position can be computed. Geometric properties of triangles are used to determine positions of other nodes.

In the second step, the directions of the local coordinate systems are adjusted to obtain the same direction for all the nodes in the network. Two coordinate systems are said to have the same direction if the direction of their x and y axes are the same. The direction of a local coordinate system can be adjusted to a second system by rotating and possibly mirroring the system. One network coordinate system – say, the system of node $i$ – is chosen (see below how) which acts as a reference for other systems to adjust to. Nodes can then compute their positions in the referent system. Imagine we observe node $l$, a neighbor of $k$ and a two-hop neighbor of node $i$. Node $k$ knows its position in the coordinate system of node $i$, and knows the position of node $l$ in its own coordinate system. As the coordinate systems of nodes $k$ and $i$ have the same directions, the position of the node $l$ in the coordinate system of the node $i$ is simply obtained as a sum of two vectors. This is illustrated in Figure 3.3.

A problem arises once node $i$ moves as this causes a large inconsistency between the real and computed positions of the nodes, requiring all the nodes to recompute their positions. To overcome this, the authors define a set of nodes called the Location Reference Group (LRG) chosen to be stable and less likely to disappear from the network (Figure 3.4). The LRG is composed of $n$ neighbor nodes having the highest density in the network, where $n$ is set by the user ($n \in \{2, 3\}$ in simulations, see below). The LRG center is the mean of the LRG nodes' positions and is the origin of the network coordinate system. The direction of the network coordinate system is defined as the average value of the directions of the local coordinates systems of the LRG nodes. The average speed of the LRG center is expected to be much smaller than the average speed of the nodes. In this way, the position inconsistency introduced by motion can be reduced.

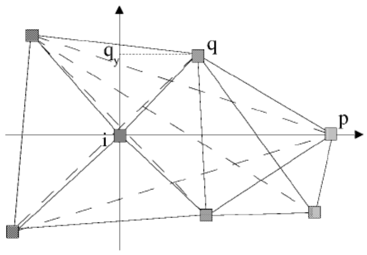



Figure 3.2: The local coordinate system of node $i$ is defined by choosing nodes $p$ and $q$.

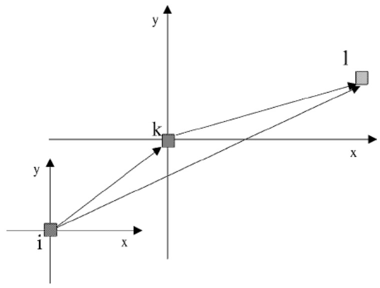



Figure 3.3: Position computing when the local coordinate systems have the same direction.

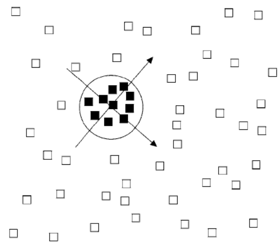


Figure 3.4: The location reference group.

A simulation with 400 nodes was performed by the authors. The nodes follow a random movement pattern: they move using a random velocity, wait for a fixed time, and then move again. It is shown that if a larger (three-hop) neighborhood is used instead of a two-hop neighborhood, the mobility of the center of the network decreases (thus increasing stability). No accuracy information is provided; reducing the position error is being mentioned as subject of future work (but has not been published). Furthermore, as the algorithm is focused on providing location information to support basic network functions (such as forwarding packets in the right direction) accuracy requirements should not be high. Communication costs are relatively high in multi-hop networks as the algorithm requires a broadcast to all the nodes in the network.

## 3.3 Online Person Tracking

An Online Person Tracking (OPT) system for an indoor environment is presented by An et al. [1]. OPT employs a passive mobile architecture such as displayed in Figure 3.1b. RSSI is used for ranging; an empirical relation was established between RSSI and distance up to 16 meters (Figure 3.5). The average RSSI of 200 measurements was used to estimate the location. An et al. only used the three strongest received signal strengths (in general the three closest anchor motes) because they claim using more does not guarantee a higher accuracy. Experiments with a static sender and receiver were performed to measure the influence of the antenna orientation on the strength of the received signal. The RSSI value varied up to 15 dBm depending on the antenna's orientation. This leads to bigger error on distance estimation when two motes are farther apart, because the variation in RSSI becomes smaller as the distance becomes larger.

The authors applied a bounding box algorithm (Figure 2.1b) to select an area in which the optimal position was sought. If there was no overlapping area of circles, the estimation area was expanded to make sure that the potential target position was included in the search area (Figure 3.6). The Minimum Mean Square Error (MMSE) algorithm was employed for target location estimation within the selected area. This method is commonly used in statistics and signal processing.

In the conventional MMSE method (dubbed C-MMSE) all range estimates were given the same importance in minimizing the position error. In the faster, modified version (M-MMSE) only the first two highest RSSI values were involved in the MMSE estimation process. This gives two possible positions and the third mote was used to choose between the two. A weighted version (W-MMSE) is also proposed by the authors. The higher the slope of the empirical curve between distance and RSSI, the higher the assigned weight. In other words, higher RSSI values are considered to be more reliable than low ones.

In the controlled experiments W-MMSE outperformed all the algorithms and C-MMSE provided the least performance. In real-world experiments the W-MMSE algorithm was tested to track the real-time position of a slowly 'moving' person. The person moved from position to position, but had to halt in order for the system to get an approximate position. Ten motes were

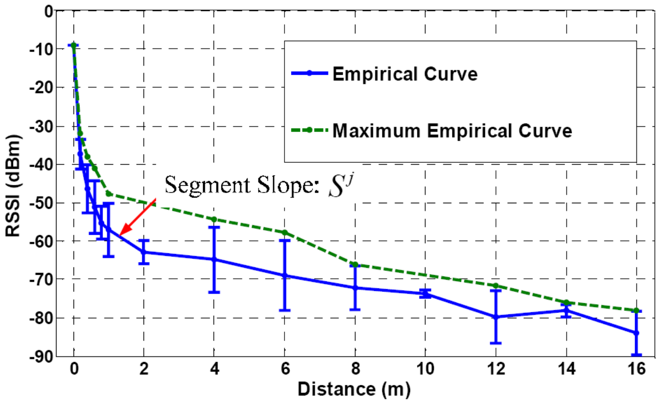



Figure 3.5: Empirical relation curve.

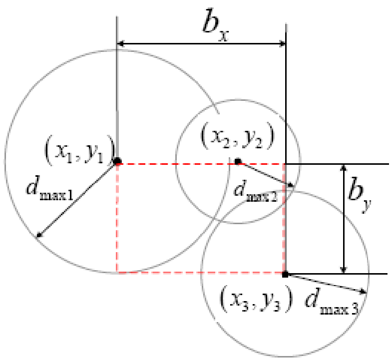



Figure 3.6: Boundary selection without overlapping area.

placed at fixed positions with a distance of 4 m between them. The dimension of the floor is 70 m × 12 m with a narrow corridor of 60 m × 2 m in the middle. Offices are located on each side of the corridor. The attenuation of walls was taken into account if the target mote was estimated to be in an office. Of 36 positions considered in the corridor, 50% of the estimated locations were within 2 m of the real location, and 90% within 4.5 m. When the person was in an office room, 16 experimental positions were used. The median accuracy was approximately 3.8 m and 90% of the time the accuracy was 6.0 m.

## 3.4 Trajectory Matching

Lee et al. [23] present an algorithm enabling localization of moving wireless devices in an indoor setting. An active mobile infrastructure (Figure 3.1a) is employed; a burst of 5 packets in 50 ms is sent by the mobile node every 0.6 seconds. Ten nodes were deployed at fixed locations and one mobile node was being localized.

The mobility of the users is modeled by learning a function which maps a short history of signal strength values to a 2D position. During the training phase, ground truth locations of the mobile user are required; however, locations of infrastructure nodes are not needed. The authors used radial basis function fitting to learn a reliable estimate of a mobile node's position given its past signal strength measurements. RSSI measurements were prefiltered by a box (mean) filter and then fed into the learned function to obtain the position of the mobile node.

Nine different trajectories were evaluated: five for training and four for testing. An area of approximately 30 m × 25 m was used for experimentation. Experimental data shows that the variance due to reflections is particularly severe when either transmitter or receiver was moving, even at low speeds. Several parameters of the algorithm were optimized. The number of past measurements determines how much historical information about the trajectories is available. Using four past values was found to be optimal. In 50% and 97.5% of the cases the accuracy is 1.0 m and 4.1 m, respectively. Besides RSSI, PRR (Packet Reception Ratio) and LQI (Link Quality Indicator) were also used to learn the mapping function. PRR is computed by counting the number of received packets in each burst, LQI values remain high whenever there is radio coverage and drop sharply as soon as connectivity is about to be lost. Unsurprisingly, PRR and LQI were found to perform worse than RSSI.

Since they use past measurements at fixed time intervals, the authors implicitly assume that the speed of the mobile user at a given position is similar during training and localization. Explicitly handling speed differences is subject of future work.

## 3.5 Comparison

In Table 3.1 the localization systems which we have seen so far are compared with respect to accuracy, node density and technique. The table is partially based on a comparison by Kaseva et al. [20], but some values have been corrected after having carefully reviewed the cited papers. The Self-Positioning Algorithm is not included in the overview because it has only been tested in simulations and no accuracy measurements were provided. Although trajectory matching uses RSSI measurements it is considered to perform scene analysis as it requires training for a specific environment.

| Localization system | Accuracy (m) | Anchor node density ($m^2$ per node) | Technique |
|---|---|---|---|
| Ferret [40] | 0.6–1.0 (A) | 2–4 | RSSI/potentiometer |
| Cricket [37] | 0.02–0.2 (M) | 2 | Ultrasound time-of-flight |
| MoteTrack [26] | 2 (M) | 87 | Scene analysis |
| RADAR [3] | 2.9 (M) | 326 | Scene analysis |
| Online Person Tracking [1] | 2/3.8 (M) | 8/48 | RSSI |
| Trajectory Matching [23] | 1.0 (M) | 52 | Scene analysis |

Table 3.1: Characteristics of different indoor localization systems. Accuracy is either a median (M) or an average (A) value.

A number of comments should be made to put the accuracy of the different systems into perspective. The anchor node density is defined as the number of square meters one anchor node has to cover (on average). It is important for making a comparison because the lower the value, the easier it is to obtain a relatively high accuracy. RADAR is an exception in the sense that it uses WLAN technology for anchor nodes as opposed to sensor nodes. In Ferret between 5 and 11 nodes are used, which explains the variation in accuracy and node density. The accuracy level of Cricket depends on the mobile node's speed. OPT has been evaluated in a corridor and office rooms; as the corridor covers a smaller area and has no interfering walls, a higher accuracy is obtained. Cricket and the Trajectory Matching algorithm are the only systems having tested accuracy of moving nodes; MoteTrack, RADAR, and OPT track devices which may change their location but need to be stationary for localization. Ferret and Cricket were tested in only one room, while other systems were evaluated in office environments having multiple obstructions and realistic error sources.

Which system is best depends on the application. In the next chapter we will discuss the requirements for the application. Based on this, the most appropriate hardware and software setup is chosen.

# Chapter 4

# System Setup

The system setup depends on the intended application. Therefore, the requirements of the application are given first. Then, the considerations for the hardware choice are discussed, followed by the hardware and software setup.

## 4.1 Requirements

The purpose of the application is to give a demonstration at a stand when Logica presents itself at events ("Bedrijvenbeursdagen"). The *primary* goal is to show the relative positions of deployed sensor nodes on a map, which is displayed on a PDA. The *secondary* goal is to develop a device which points the user to Logica's stand and shows how far away it is located; for example, a display attached to a sensor node indicates the direction by an arrow and shows the distance in meters.

The environment in which the WSN will operate and requirements with respect to accuracy, mobility, and deployment are described below. I have established these requirements in consultation with my supervisor at Logica, Martijn Vlietstra, and verified them after writing them down.

**Environment** The events take place at various indoor locations which tend to be the same every year, although the location of the stand may change. The event floor is spacious and usually features pillars, but walls may also be present. Other obstructions include stands and (moving) people. Typically, the floor covers approximately 2500 $m^2$ (50 m $\times$ 50 m). All stands are located on the same level.

**Accuracy** The mean accuracy must be 5 meters or less. The maximum error allowed is 10 meters.

**Mobility** At least one node is mobile and its position needs to be updated as often as is needed to achieve the required accuracy. The maximum speed of the node is walking speed (1.4 m/s).

**Deployment** A number of static nodes will be deployed to help locate the mobile device(s). Deployment is done manually and must take no longer than 10 minutes. Preferably, static nodes are to be placed at Logica's stand, but other deployment locations are also possible. If the secondary goal is achieved, a second location can be used for handing out devices.

**Availability** The hardware should be commercially available; it will not be custom-built. A range of RF motes and one ultrasound solution is currently available.

**Cost** A limited number of nodes can be bought.

## 4.2 Hardware

### 4.2.1 Considerations

The principal choice for a hardware solution is between an ultrasound (Cricket) and an RF-based approach, because this determines which localization methods are feasible. I will compare both approaches based on the requirements of the intended application. Per the availability requirement, we only consider off-the-shelf hardware.

**Environment** Both ultrasound and RF suffer from multipath effects caused by obstructions. Ultrasound is more limited, however, as the receiver and transmitter require line-of-sight. Furthermore, the range is fairly short: 12 meters in the most favorable case in the Cricket system. RF signals can be received up to at least 50 meters indoor [9], but this depends on the hardware and environment. Because of their larger range, we decided that radio signals are more suitable for the depicted environment.

**Accuracy** As far as accuracy is concerned, the use of either technique is plausible. Using ultrasound can give accurate positions up to the centimeter level, but the requirements are not that stringent. Radio-based approaches can also deliver the required accuracy (see section 3.5), but this depends on the used algorithms and test setup. For example, if 10 nodes are used, the anchor node density will be 250 $m^2$ per node, which leads to a much sparser network (negatively influencing the accuracy) than is used in most of the discussed systems.

**Mobility** Both approaches can be used for tracking a mobile node. There are no specific advantages of either technique.

**Deployment** Limited time for deployment is available, so the system setup and calibration must be efficient. In Cricket careful orientation of the directional receiver is required, because the angle at which a signal is received is important for both accuracy and connectivity. RF motes are less susceptible to erroneous placement as they have omnidirectional antennas in general.

**Interference** The ultrasonic transmitter in Cricket operates at 40 kHz; it is found that some fluorescent lamps also generate 40 kHz ultrasonic waves which cause interference [30]. Other than this no interference is expected.

RF motes operate in the 868-MHz band or 2.4-GHz band. Not many devices operate in the first band, but in the second band WLAN is also present. Laptops and other devices at an event are likely to use WLAN technology, and cannot be shut down.

**Cost** To cover the described area a dense network of Cricket motes would be required, which would lead to relatively high costs. The average RF mote costs two-thirds of one Cricket mote.

Based on this comparison, an RF-based approach is considered to be the most appropriate, mainly because the available ultrasound-based mote has limited range, requires careful deployment, and is an expensive solution. The various available RF motes are evaluated on the basis of the above requirements. Mobility and deployment do not influence the decision as the choice for a mote has no impact on these requirements. For the environment a decent indoor range is useful. Because RSSI will be used for ranging and influences accuracy, good RSSI support is a must. Interference should be absent or measures should be taken to minimize it. Table 4.1 gives an overview of considered RF motes. All motes use TinyOS as their operating system (see section 4.3.1 for a description).

I have chosen the IRIS mote [9], which is produced by Crossbow, because it has a relatively large indoor range of 50 m and a wide RSSI dynamic range. It is compliant with the IEEE 802.15.4 standard which means it supports techniques such as direct sequence spread spectrum to make sure the mote is resistant to RF interference. When properly configured, RF interference and lost

| RF mote | Frequency (MHz) | Maximum indoor range (m) | RSSI dynamic range (dBm) | RSSI accuracy (dB) | Cost (euro) |
|---|---|---|---|---|---|
| BTnode | 868 | ±30 | −105 to −50 | ±6 | 165 |
| IRIS | 2405 | ±50 | −91 to −10 | ±5 | 120 |
| Mica2 | 868 | ±30 | −105 to −50 | ±6 | 120 |
| MicaZ | 2405 | ±30 | −100 to 0 | ±6 | 105 |
| TinyNode 184 | 868 | ±50 | −100 to −30 | ±3 | 73 |
| TinyNode 584 | 868 | ±100 | −110 to −85 | – | 91 |

Table 4.1: Transceiver-related specifications and cost of considered RF motes.

data can be reduced through channel selection [8]. TinyNode 184 is also a good option, but is not chosen because driver support in TinyOS is limited for its transceiver at the time of writing.

The complete hardware setup is presented in the next section.

### 4.2.2 Setup

In Figure 4.2 the hardware setup is shown. Dashed and solid lines represent wireless and wired connections, respectively. One IRIS mote is attached to an interface board and acts as a *base station*. This mote and seven other motes form the IRIS mote network. The base station collects information from the network and relays this to the PC. In turn, the PC processes network events and then updates the information on the server. The server computes the positions of nodes. The PDA asks the server for an update of the nodes' coordinates with a certain interval.

#### IRIS

The IRIS mote [9] uses a 2.4 GHz Atmel radio transceiver which has programmable output power from -17 dBm up to 3 dBm and receiver sensitivity of -101 dBm. A data transfer rate of up to 250 kbps is supported. The processor board is based on ATmega1281, a low-power microcontroller which has access to 8 kB RAM and 128 kB flash memory. An expansion connector allows a connection to a variety of external peripherals (e.g., a sensor board connected to sensors). As a power supply, two AA batteries are typically used, but a mote is powered through USB bus if connected to an interface board. The mote fits in the palm of one's hand with its size of 58 × 32 × 7 mm (Figure 4.1).

#### Interface Board

The USB interface board (MIB520) provides connectivity to one IRIS mote at a time. Two serial ports are emulated over USB, one for communication with a mote and one for programming. A mote can also be reprogrammed over-the-air to receive an update of a program, but has to be programmed through the interface board first with the specific program.

#### PC

The PC communicates with the base station over the USB connection and with the server over an Internet connection. In effect, it allows for communication between the base station and the server.

#### Server

The external server is a dedicated server running Microsoft Windows Server 2003, Web Edition. It is used as an application server to which clients, such as the PDA, can connect. The reason a server is used is because the PDA must be able to obtain the data from the PC over a wireless connection, which can be done relatively easy using this setup. Running the server application on the PC would be possible, but connecting to it from outside the network the PC is in may prove difficult if the network is protected with a firewall.

**PDA**

The PDA is a HTC Advantage X7500 running Windows Mobile 5 at 624 MHz. It uses GPRS to connect to the server. It is used to register the location of nodes in the deployment and learning phase (see section 5.2 for a description of the phases). This saves deployment time compared to using a PC to connect to the server because the user does not have to keep walking back and forth to the PC between node registrations. In the localization phase, a mobile node and the PDA can be used together to show the position of the PDA on the map, or the PDA can be used to track another person holding the mobile node. Note that the PDA is not connected to any sensor node.

Figure 4.1: IRIS mote

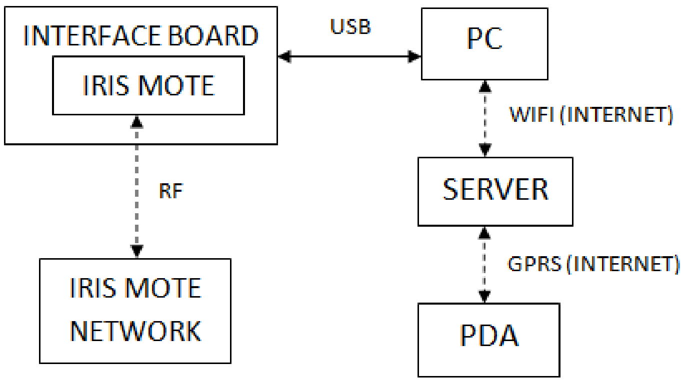



Figure 4.2: Hardware setup

## 4.3 Software Setup

### 4.3.1 Motes

The way motes are programmed depends on their function. There are three types: base, static, and mobile. The base mote is connected to the interface board and has to handle the communication between the PC and the mote network. All the non-mobile nodes listen for messages sent by mobile nodes. Each message contains the sender identification, packet number and sequence number. The mobile node sends a packet burst with a regular interval and increases the packet number by one each time this is done. The sequence number is used to identify a packet within a burst. RSSI and LQI information is requested for each packet by the receiver. All the data of one packet burst is aggregated into one message and then sent to the base station. The sending is done using a multi-hop routing protocol, because not every mote may be in range of the base station. I have written the software for the nodes, except the routing protocol. The motes use TinyOS.

TinyOS is an open-source, event-driven operating system designed for wireless embedded sensor networks. It is written in nesC, which is an extension to the C programming language designed to embody the structuring concepts and execution model of TinyOS. Programs are built out of components, which are assembled to form whole programs. TinyOS's component library includes network protocols, distributed services, sensor drivers, and data acquisition tools.

There are two multi-hop routing protocols in TinyOS available: TYMO and the Collection Tree Protocol. TYMO is the implementation on TinyOS of the DYMO protocol, a point-to-point routing protocol for mobile ad-hoc networks. The current TYMO version is not stable, however. Therefore we have chosen to use the Collection Tree Protocol (CTP) [13, 14].

CTP is a tree-based collection protocol. Messages are collected at the roots of trees. Nodes form a set of routing trees to the tree roots. In our case, the only root is the base station. CTP is a best effort protocol: it does not promise 100% reliable delivery and there are no ordering guarantees. CTP assumes that it has link quality estimates of some number of nearby neighbors. As a link estimator we use an implementation of the four-bit wireless link estimation, which can maintain a 99% delivery ratio with a transmission power of 0 dBm over large, multi-hop testbeds [15].

CTP works as follows. Nodes generate routes to roots using a routing gradient (information used to decide how to route). The protocol uses the expected number of transmissions (ETX) as its routing gradient (the lower the value, the better the link). CTP represents ETX values as 16-bit fixed-point real numbers with a precision of hundredths. A root has an ETX of 0. The ETX of a node is the ETX of its parent plus the ETX of its link to its parent. In general, CTP chooses the node with the lowest ETX value, unless it has reasons to do otherwise (e.g., after losing connectivity with a candidate parent). CTP data frames also have a time has lived (THL) field, which the routing layer increments on each hop. CTP uses the ETX and THL fields to deal with routing loops and packet duplication.

### 4.3.2 PC

The PC connects to the server as a client and forwards messages it has received from the base station. TinyOS provides classes to read and interpret data sent over the USB port. I have written a Java program which sets up the connection to the server. Furthermore, it drops duplicate packets and then sends the unique ones to the server.

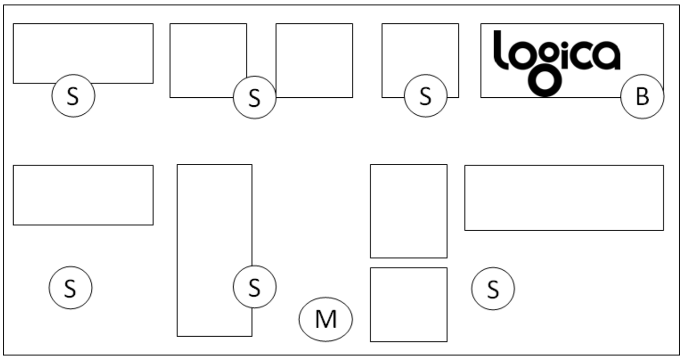



Figure 4.3: Schematic illustration of the map displayed on the PDA. B, S, and M represent the base station, a static node, and a mobile node, respectively.

### 4.3.3 Server

A Java web application is deployed on the server. The used application server is Apache Tomcat. Tomcat is a Servlet container and provides an environment for Java code to run. The Java application is based on the previous WSN project. The user authentication, authorization, and administration parts have been reused. Users can be granted permission to access certain pages by giving them one or more roles. A user with role ‘user’ can only view a loaded environment map and register node locations, while an ‘admin’ has access to all functionality. I have added the following functions:

1. *Environment Map*
   One can add, edit, delete, and load a map of the environment. When adding or editing a map, the (physical) location that the map represents and a general description may be

specified. The name and image file location must be specified. The width and height that the map represents in the physical world are also required. When a map is loaded the image is displayed. A schematic example of what could be displayed is shown in Figure 4.3. The position of the mobile node is updated regularly.

2. *Node Registration*
   If a map has been loaded, the user can enter the location of a static node on the map by clicking on it and entering the node number. These locations are saved and displayed. When the map is reloaded or another map is loaded, the nodes and their positions are deleted.

### 4.3.4 PDA

The PDA uses Opera Mobile 9.5 as a browser to view the Java web application. This browser is used because of its good support of web standards on a mobile device.

# Chapter 5

# Results

## 5.1 Experimental Results

To determine the relation between signal strength and distance we need to perform measurements. I used two motes: a sending mote and a base station for receiving the messages and transferring them to a laptop. The base station collects three values: RSSI, LQI, and PRR. RSSI is explained in section 2.1.1 and is a value between $-91$ and $-10$ dBm. LQI stands for Link Quality Indication. The IEEE 802.15.4 standard defines the LQI measurement as a characterization of the strength and/or quality of a received packet. LQI values are integers ranging from 0 to 255 (the higher the value, the better the link) [2]. PRR is the Packet Reception Rate and is the ratio of received packets to the total number of packets. RSSI and LQI are provided by the mote's transceiver; PRR is computed. No WiFi networks were present which could interfere.

A series of parameters influence the RSSI measurements. We describe the ones identified by Stoyanova et al. [39]:

**RF frequency** A center frequency of 2.405 GHz (channel 11) is used. Note that channels 11, 25, and 26 are suited to avoid interference with WiFi [8]. Channel 11 has been found to be the most reliable channel for the IRIS mote's transceiver by TinyOS developers. The center frequency $F_{CH}$ is defined as follows: $F_{CH} = 2405 + 5 \times (channel - 11)$[MHz] [2].

**Antenna orientation** Both motes were in a horizontal position. The sending mote was always in front of the person holding it. This means the person was an obstruction in case of the sender moving away from the receiver.

**Variation of transceivers** The same motes were used each time for the sender and base station.

**Transmission power** The transmission power is set to the maximum output power of the IRIS mote, being 3.0 dBm, unless noted otherwise.

**Environment** We conducted the first set of experiments in an empty room of 50 m $\times$ 16 m, measuring 2.7 m in height. The second set was measured in an open workspace environment of the same size (illustrated in Figure 5.1 and 5.2), the only difference being five extra office rooms located farthest away from the base station.

**Height from the ground** The base station was placed on a chair at a height of 0.58 m. The sending mote was held 1.0–1.1 m above the ground.

### 5.1.1 Empty Room

I performed two experiments in the empty room. The sending mote was static in the first experiment and moving in the second. The transmitter sends a packet burst of five messages as fast as possible (in general within 50 ms). PRR is computed per packet burst.

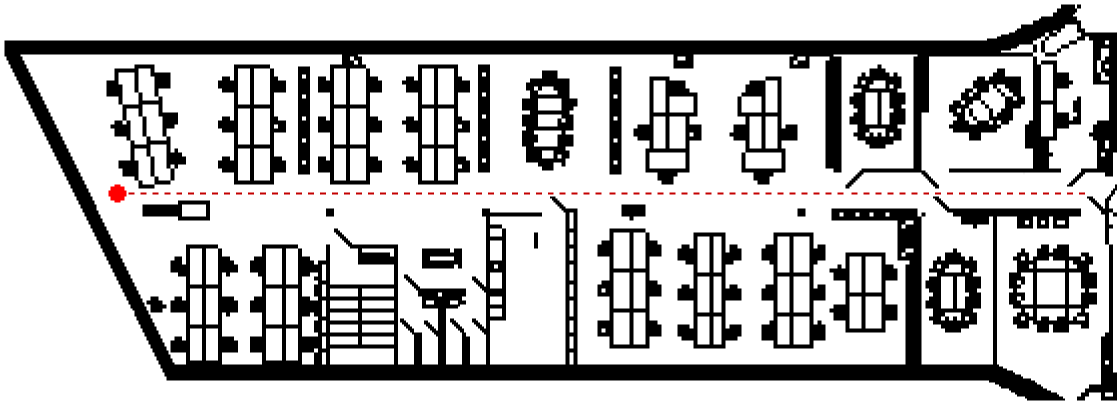


Figure 5.1: The office in which tests were performed. Measurements were done along the dashed line. The base station was located at the round dot at the start of this line.

In the first experiment, the sending mote was kept in a fixed position during sending. The closest measurement was done at 0.5 m. We then took measurements every meter in the range 1–10 m and every three meters from 10–46 m. RSSI values were averaged over four packet bursts. The LQI value was a consistent 255, indicating the link quality was good at all times. Furthermore, all packets were received. Therefore, we concentrate on the RSSI values.

The RSSI values are shown in Figure 5.3, together with two theoretical models. Fitting the log-normal shadowing model (see section 2.1.1) to the experimental data using the least squares method results in an attenuation constant of 1.64. This means the signal decays at a lower rate than a signal in free space. This is caused by the reflection of the signal off of walls, the ground, and the ceiling. Reflection also causes the variation in the RSSI value as it strengthens or weakens the signal. We can model this variation to some extent using the two-ray ground reflection model described by Stoyanova et al. [39]. This model takes the reflection of the signal via the ground into account. By considering the height of the transmitter and the receiver one can compute the length difference between the reflected and the direct (line-of-sight) signal. This difference determines if the electric fields of the two signals reinforce each other or cancel each other out. Assuming the ground reflection is perfect, the resulting combined electric field is used to calculate the received power. As we can see in Figure 5.3, the two-ray model matches the variation of the measured data to a certain degree. However, least squares fitting shows the log-normal shadowing model better fits the measured data than the two-ray model.

Figure 5.2: The experimental setup at the base station.

In the second experiment in the empty room, the transmitter was moving and sending a packet burst every second. The time it took to walk 46 m was measured to compute the average speed. The distance was covered ten times: five times back and forth. The LQI value dropped incidentally below 255 and only a few packets were missed. Figure 5.4 and 5.5 show the results of the RSSI measurements.

We differentiate between moving away from the base station and moving towards it, because in the latter case the measured values are higher in general. Still, beyond 10 meters it is very hard to make a reliable distance estimate based on a given RSSI value. And although the graphs seem to follow the same pattern at certain points (e.g., the dip at 40 m followed by a slight rise in Figure 5.5), they are not consistent with each other in general.

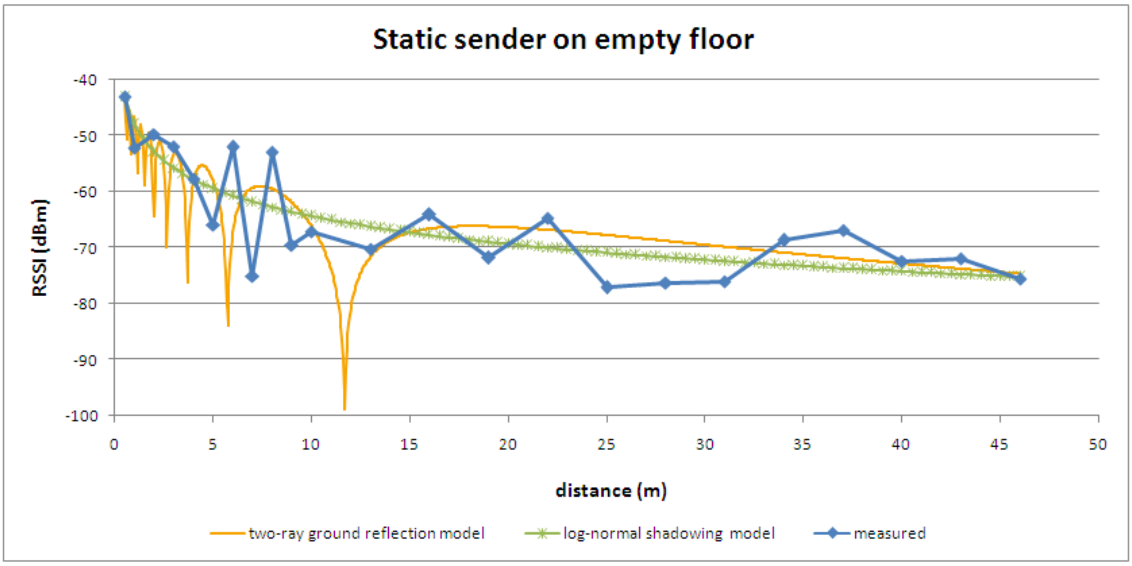



Figure 5.3: The relation between RSSI and distance according to measurements. Two theoretical models are fitted to the experimental data.

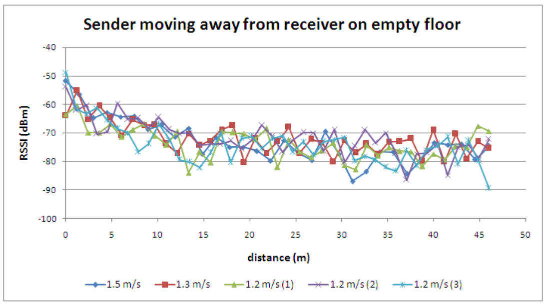



Figure 5.4: The sender moves away from the base station.

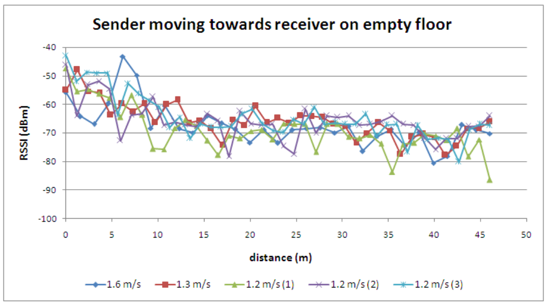



Figure 5.5: The sender moves towards the base station.

### 5.1.2 Office

Because the application will operate in an environment with many obstructions, we have also conducted experiments in an open workspace to test in a more realistic environment. Furthermore, if one is given a signal strength measurement it is hard to derive a unique distance between the sender and receiver from Figure 5.3. Therefore we want to experiment with various transmission powers to obtain more diverse data and use this information to narrow down the number of possible distances.

The transmitting node sends a burst of ten packets with an interval of 25 ms between packets. Every two packets a different transmission power is set. Results were averaged over ten packet bursts, that is twenty messages per level of transmission power. The closest measurement was done at 0.2 m. We then took measurements at 0.5 m, every meter in the range 1–3 m and every three meters from 3–43 m.

The average LQI and PRR dropped below 255 and 1, respectively, when connectivity was about to be lost. In these cases the RSSI value was $-91$ dBm (the minimum value) most of the times. LQI and PRR may be of use to differentiate between messages having the minimum RSSI value.

The minimum, average, and maximum RSSI values for the different transmission powers are shown in Figure 5.7. In case (d), no packets are received beyond 31 m. The log-normal shadowing model is fitted to the data using the least squares method (Figure 5.6), giving an attenuation constant of 1.84. The form of the graphs is very similar for all transmission power levels. Unfortunately, this has the consequence of not giving us much extra information. For example, it does not help with determining whether a packet has been sent from a distance of 10 or 20 meter, because none of the graphs can make this distinction properly.

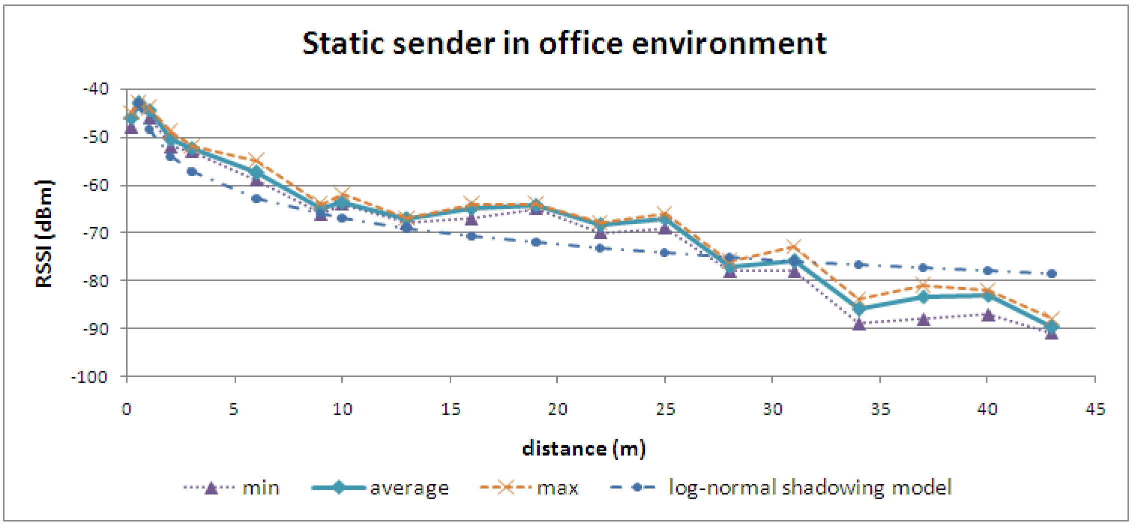
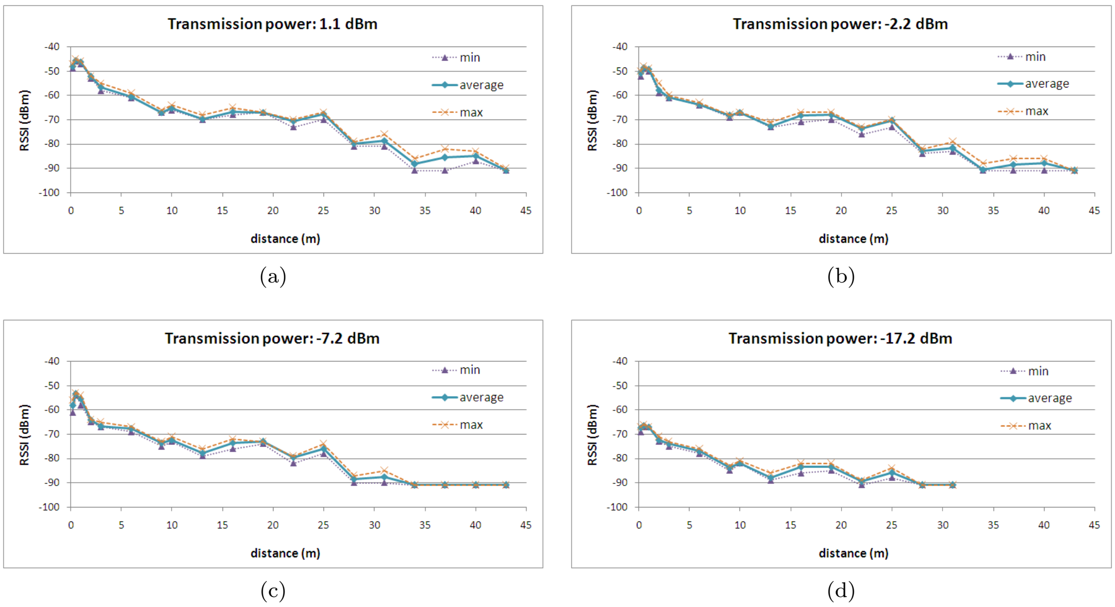



Figure 5.6: Static sender in an office environment using maximum transmission power of 3.0 dBm.




(a) (b) (c) (d)

Figure 5.7: Static sender in an office environment using various levels of transmission power.

## 5.2 System Operation

There are three phases in the system's operation: deployment, learning, and localization.

### 5.2.1 Deployment

First, the user loads a map of the current environment through the server's web interface. He or she then deploys nodes manually and registers their approximate positions (except for mobile nodes). To save deployment time it would be better if node positions are determined automatically, but to do this the WSN has to have two properties: global rigidity and a rather high node degree.

**Global rigidity** To make sure there is a unique solution for the mobile node's position we need a uniquely localizable sensor network. However, this is a fundamental problem in distance-based sensor network localization [28]. We can make a distinction between non-rigid, rigid, and globally rigid graphs. In Figure 5.8 we can see examples of such graphs. Non-rigid graphs can be continuously deformed to produce an infinite number of different realizations (such as in the rectangle case), while rigid graphs cannot [31]. However, in rigid graphs, there are two types of discontinuous deformations that can prevent a realization from being unique. The first is flip ambiguity, illustrated in the middle graph of Figure 5.8. Flip ambiguities occur for a graph in a $d$-dimensional space when the positions of all neighbors of some vertex span a $(d-1)$-dimensional subspace. In the example, the neighbors $n_1$ and $n_2$ create a mirror through which the vertex $v$ can be reflected. The second deformation is discontinuous flex ambiguity: the temporary removal of an edge or, in some cases, a set of edges allows the remaining part of the graph to be flexed to a different realization (Figure 5.9) [28]. A graph is globally rigid if it is rigid and has a unique realization.

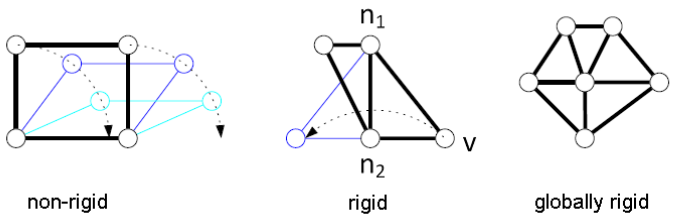



Figure 5.8: Rigidity of graphs

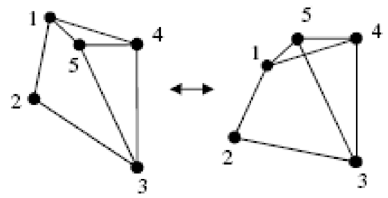



Figure 5.9: Discontinuous flex ambiguity

**Node degree** Theoretically a globally rigid graph can be found in a two-dimensional network if we have the absolute positions of at least three non-collinear sensors and the exact distances between nodes. However, noisy distance measurements complicate this. Moore et al. present a localization algorithm which is robust against such errors [31]. A general result of their simulations is that as noise goes to zero, nodes in large networks must have degree 10 or more on average to achieve 100% localization. We have 8 nodes at our disposal, which is not enough to reach a reasonable localization percentage using this algorithm. For example, when the mean-square error of the distance measurements is 10 cm (2.9% of the maximum ranging distance) and the average node degree is 10, then the localization rate of nodes is only 5%. Therefore, we have chosen to let the user register the location of the nodes. This has the advantage that if batteries of a node run out it can still be found later on.

### 5.2.2 Learning

In the learning phase the user helps the system to learn about the environment. The received signal strength depends for a good deal on the environment as we have seen in the experiments. Because the system will have to function in different locations it has to adapt to the location.

This is accomplished by letting the user perform measurements at various positions such that the relation between signal strength and distance can be learned for that specific location. Before each measurement the user registers the mobile node's position so that the distance between it and other nodes can be computed. The mean is computed for every set of measurements from a specific distance. Once all measurements are taken, non-linear regression is performed on each data set, creating a RSSI-distance relation for every sender-receiver combination. Two measurements per combination are demanded; otherwise, no graph can be fitted with some confidence. The data is fitted to the log-normal model using least sum of squares to find an optimal value for the attenuation constant. The reference signal strength value is a fixed value, obtained from 0.5 m distance. We assume that the user can place the node in such a location that there are no objects within that distance which have a considerable impact on this RSSI value. Moreover, not having to perform this reference measurement saves deployment time.

An advantage of the learning approach is that the system not only adapts itself to its surroundings, but also to the other parameters mentioned in section 5.1. Only antenna orientation may not be compensated for. The IRIS mote has an omnidirectional antenna, but we do not know if the user is between the sender and a receiver. We can only advise the user during learning to stand in the same direction as he would expect a visitor to do in order to obtain a good estimate.

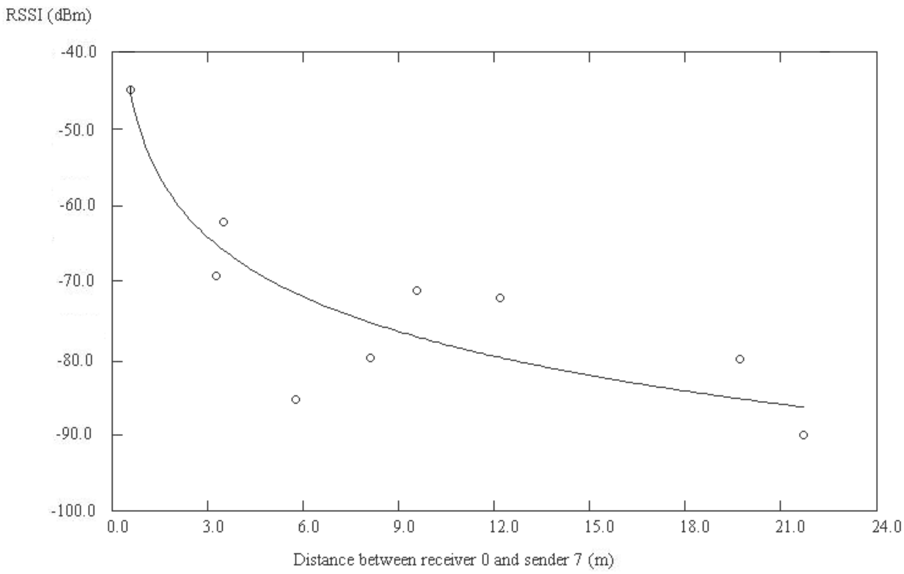



Figure 5.10: Automatic fitting of measured data to the log-normal shadowing model.

### 5.2.3 Localization

An active mobile architecture is employed (Figure 3.1a). This setup has the advantage of being more accurate than the passive mobile architecture because a transmitted signal by the mobile node will be received simultaneously by the static nodes. We collect the RSSI values at the server and select the three motes which have the highest ones because these are considered to be the most reliable. Then we compute the distances between the motes and the mobile node. We adapt Equation 2.2 (multiplying it by 0.5) because the reference distance is 0.5 m instead of 1 m. Next, we estimate the position of the sending mote by selecting the point for localization that gives the minimum total error between the estimated position and distances estimated from measurements. We take the same approach as An et al. [1] by using the Weighted Minimum Mean Square Error (W-MMSE) to minimize this error. Assume that there are $N$ static nodes taken into account and $d_i$ is the estimated distance between the mobile node and a static node $i$ ($i \in \{1, 2, \ldots, N\}$)

located at $(x_i, y_i)$, then we can define the error estimation function as:

$$MMSE = \sqrt{\sum_{i=1}^{n} w_i \cdot error_i^2} \tag{5.1}$$

where $w_i = \frac{1}{d_i}$, $error_i = |d_i - \sqrt{(x_i - x_e)^2 + (y_i - y_e)^2}|$ and $(x_e, y_e)$ is the estimated position in two-dimensional coordinates of the mobile node.

## 5.3 System Validation

This section validates whether the built system meets explicit and implicit requirements. The explicit requirements are described in section 4.1, while the implicit ones were, as by definition, not written down.

### 5.3.1 Explicit Requirements

The primary goal, showing the positions of deployed sensor nodes on a map, has been achieved. The secondary goal has not been reached due to time constraints. To point the user to Logica's stand an LCD display could be connected to a sensor board (e.g., the MDA300 board [10]), which in turn is attached to a sensor node. The sensor node should be programmed to interface with the display. Moreover, the server should inform the mobile node about the distance and direction to the stand.

**Environment** The system was evaluated in two environments: the open workspace environment of Figure 5.1 (wing 4B) and a similar environment with a size of 31 m × 14 m (wing 4C). The environments are similar to the intended environment as they are quite spacious and feature metal bookcases, pillars, and walls. During evaluation people were sitting at their desks or walking around. In wing 4B, five static nodes were deployed in the area, while one static node and the base station were deployed outside the area to send the results to the server. In wing 4C, five static nodes and the base station were set up.

**Accuracy** Whether the accuracy requirements are met depends on the environment. I determined the difference between the estimated and the real location for wing 4B and 4C at 20 and 12 different positions, respectively. In both environments eight positions were used in the learning phase. The mobile node was turned on approximately six seconds during each position estimation. Both C-MMSE and W-MMSE were evaluated. C-MMSE is obtained from Equation 5.1 by setting all weights to 1. W-MMSE was found to perform better than C-MMSE as was also found by An et al. [1] (see section 3.3); the following results assume W-MMSE is used.

The mean and maximum error for wing 4B were 6.4 m and 19.5 m, respectively, and 4.1 m and 9.0 m for wing 4C. The accuracy requirement is not met in the former case, but is in the latter. The mean error is somewhat misleading, however, because large position errors influence this measure rather heavily. The cumulative distribution function (CDF) of the position error, illustrated in Figure 5.11, gives a better insight into how accuracy and precision are related. For example, 50% of the position estimates are accurate up to 3.7 meters on wing 4B. If the position error is small it is hard to identify the cause. We therefore look at large errors to determine why signal strength values are much larger or smaller than expected. Analysis of position errors larger than 6 meters on wing 4B shows there are a number of causes: the signal can be either strengthened or weakened due to multipath effects, or walls or pillars lower the measured signal strength value significantly. The expectation is that these error sources are also the cause of position estimation errors smaller than 6 meters, so dealing with these error sources will most likely improve the overall accuracy. Suggestions for improving the accuracy are given in chapter 6.

The obtained accuracy may seem low compared to other systems if we look at Table 3.1, but this is partially because in other systems much time is invested in optimizing the results for the specific environment. However, we are constrained in deployment time. To make a fair comparison we estimate how much training data can be gathered in the ten minutes available. In our setup eight measurements can be performed. For MoteTrack and RADAR there is also accuracy data available assuming eight measurements can be collected. Such data is not available for OPT; fourteen measurements were used to obtain the relation between distance and RSSI. For Trajectory Matching we assume the user can walk on every path in the environment twice. Ferret and Cricket are not considered in this overview as a large number of nodes were used in a relatively small area to evaluate these systems. Table 5.1 compares the systems' accuracies. It shows our system performs well compared to most other systems while using less nodes per square meter than most systems.

**Mobility** As we have seen in section 5.1.1, the signal strength fluctuates unpredictably in case the node is moving. Therefore, the node needs to be stationary in order to obtain a position estimate. This estimate is typically updated within 6-10 seconds in the user interface. The update speed depends on the number of hops between the node located farthest away from the base station, and the connection speed of the client which is connected to the server.

**Deployment** Deployment takes between 8-10 minutes, depending on the number of nodes to be deployed. If for some reason the link quality between nodes is bad, typically 5 more minutes are needed to place the nodes in such a position that the link quality is good to ensure messages can be sent quickly to the base station.

**Availability** The commercially available IRIS mote has been bought.

**Cost** Eight nodes were bought and the available budget was not exceeded.

| Localization system | Accuracy (m) | | | Anchor node density ($m^2$ per node) |
|---|---|---|---|---|
| | 25% | 50% | 80% | |
| Our system | 2.9 | 3.7 | 7.2 | 109 |
| MoteTrack [26] | 4.3 | 7.2 | 14.9 | 87 |
| RADAR [3] | 3.1 | 6.3 | — | 326 |
| Online Person Tracking [1] | 3.3 | 3.9 | 5.4 | 48 |
| Trajectory Matching [23] | 0.7 | 1.5 | 2.0 | 52 |

Table 5.1: Given that limited deployment time is available, our system performs similar to or better than most other approaches.

### 5.3.2 Implicit Requirements

The most important implicit requirement is that the demonstration is stable. Stable can be defined as the ability of the system to cope with errors or deal with unexpected situations. Efforts have been taken to establish this, such as:

1. Reporting of error messages to the user in case there is a failure. For example, if too few measurements are done during learning, this is shown in the user interface.

2. Establishing reliable communication links. Messages between mote network and server are buffered and retransmitted if needed, aggregated if possible, and dropped if unnecessary.

3. Showing the status of the sensor nodes. Static nodes send a "heart beat" at a certain interval to indicate that they are still up and running. The user interface shows each node colored

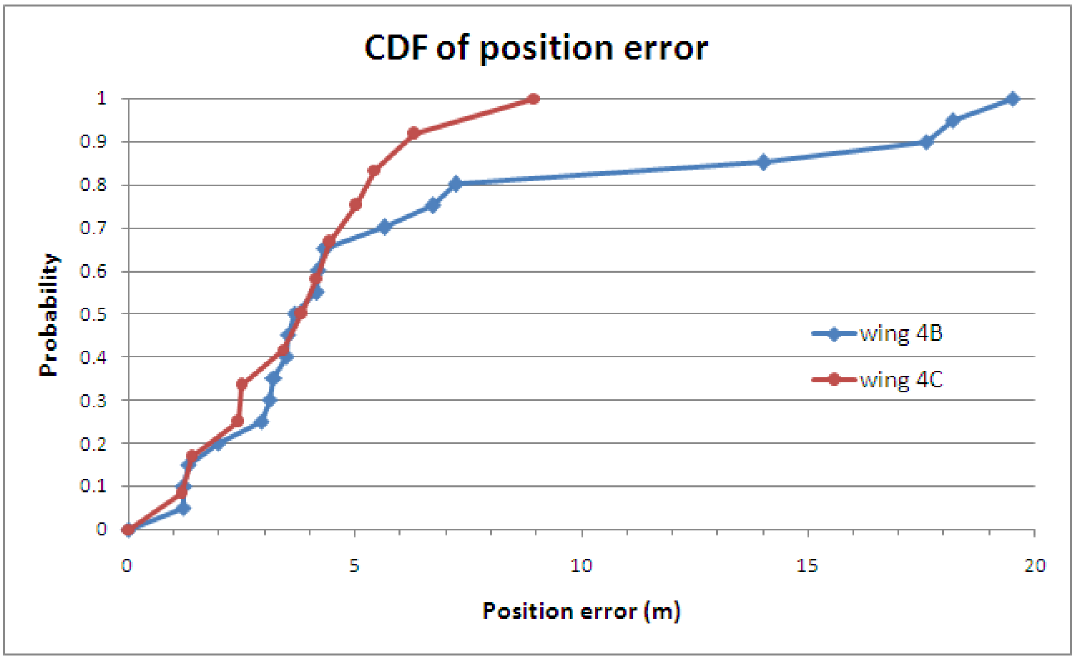



Figure 5.11: The cumulative distribution function (CDF) shows the probability of achieving a given accuracy. For example for wing 4B, the position estimate is within 7.2 meters of the real position 80% of the time.

green, orange or red, depending on how long ago the last heart beat was received. This is useful because sometimes a random node stops sending messages for unknown reasons; restarting the node fixes this problem (temporarily). The heart beat has also proven handy in case other people turn off sensor nodes ("It did not seem to be working."). Even security personnel was alerted once because a sensor node looked quite suspicious!

# Chapter 6

# Conclusion

We have discussed the most relevant techniques for indoor localization in Wireless Sensor Networks, being lateration, angulation, and scene analysis. Furthermore, a comparison was made between several localization systems. We can conclude that the choice for a certain system or technique depends on the intended application. The application requirements therefore determine for a great deal which hardware and software setup is feasible.

A prototype of a system for determining the position of a moving device has been developed. With some help of the user, the system can adapt itself to its environment within limited time by learning the relation between distance and signal strength. The primary goal, showing the relative positions of deployed sensor nodes on a map which is to be displayed on a PDA, has been achieved. The system's main advantage over most other existing systems is its short deployment time in a new environment while still achieving a reasonable accuracy. Furthermore, the developed system is not limited to indoor environments. It can be adapted easily to other environments by performing automatic function fitting on a radio propagation model suited for the specific environment (e.g., outdoor).

The requirements of the application were validated. Improvements can be made with respect to accuracy and mobility. The accuracy requirements were met in a small open workspace environment. Multipath effects and obstructions such as walls and pillars were found to be causes of position errors. To be able to reach the desired accuracy in a large space, a relatively simple solution would be to increase the number of anchor nodes. However, this would also increase the cost of the system. Another way to improve accuracy would be to use a wall attenuation model, which systems such as RADAR and OPT employ. Such a model compensates for walls between sender and receiver. Our model could be extended with a wall model by adapting the fitting function. Measurements made during learning and localization should then be corrected for the number of walls between sender and receiver. Furthermore, adding wall information to maps should be made possible in the user interface, or this information should be inferred from a map by the system to save deployment time. This extension has not been built into the system due to time limitations.

To better support a moving user, the mobile node could be attached to an accelerometer to detect if it is moving. If it is, an approach similar to the Trajectory Matching (TM) system can be used to locate the mobile user. In case the node is stationary, our system is used for positioning because TM can only localize a moving sensor node.

# Bibliography


[1] Xueli An, Jing Wang, R. Venkatesha Prasad, and I. G. M. M. Niemegeers. OPT: online person tracking system for context-awareness in wireless personal network. In *REALMAN '06: Proceedings of the 2nd International Workshop on Multi-hop Ad Hoc Networks: from Theory to Reality*, pages 47–54, New York, NY, USA, 2006. ACM.

[2] Atmel. AT86RF230 – Low power 2.4 GHz radio transceiver for ZigBee and IEEE 802.15.4 applications. http://www.atmel.com/dyn/products/product_card.asp?part_id=3941, December 2007.

[3] Paramvir Bahl and Venkata N. Padmanabhan. RADAR: An in-building RF-based user location and tracking system. pages 775–784, Tel-Aviv, Israel, March 2000. IEEE Infocom.

[4] Can Basaran. A hybrid localization algorithm for wireless sensor networks. Master's thesis, Yeditepe University, Turkey, July 2007.

[5] T. Bokareva, W. Hu, S. Kanhere, B. Ristic, N. Gordon, T. Bessel, M. Rutten, and S. Jha. Wireless sensor networks for battlefield surveillance. Proceedings of Land Warfare Conference 2006, October 2006.

[6] Srdjan Čapkun, Maher Hamdi, and Jean-Pierre Hubaux. GPS-free positioning in mobile ad hoc networks. *Cluster Computing*, 5(2):157–167, 2002.

[7] Youssef Chraibi. Localization in wireless sensor networks. Master's thesis, Royal Institute of Technology, Sweden, November 2005.

[8] Crossbow. Application note, Avoiding RF interference between WiFi and ZigBee. http://www.xbow.com/Support/Support_pdf_files/ZigBeeandWiFiInterference.pdf.

[9] Crossbow. IRIS data sheet. http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/IRIS_Datasheet.pdf.

[10] Crossbow. MDA300 data acquisition board. http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/MDA300CA_Datasheet.pdf.

[11] David M. Doolin and Nicholas Sitar. Wireless sensors for wildfire monitoring. In Masayoshi Tomizuka, editor, *Smart Structures and Materials 2005: Sensors and Smart Structures Technologies for Civil, Mechanical, and Aerospace Systems*, volume 5765, pages 477–484. SPIE, 2005.

[12] E. Elnahrawy, Xiaoyan Li, and R. P. Martin. The limits of localization using signal strength: a comparative study. In *Proceedings of the First Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks (SECON '04)*, pages 406–414, October 2004.

[13] Rodrigo Fonseca, Omprakash Gnawali, Kyle Jamieson, Sukun Kim, Philip Levis, and Alec Woo. TinyOS Extension Proposal 123: The Collection Tree Protocol (CTP). http://www.tinyos.net/tinyos-2.x/doc/html/tep123.html, February 2007. Draft-Version 1.8.

[14] Rodrigo Fonseca, Omprakash Gnawali, Kyle Jamieson, and Philip Levis. TinyOS Extension Proposal 119: Collection. http://www.tinyos.net/tinyos-2.x/doc/html/tep119.html, February 2006.

[15] Rodrigo Fonseca, Omprakash Gnawali, Kyle Jamieson, and Philip Levis. Four-bit wireless link estimation. Technical Report SING-07-00, UC Berkeley, Univ. of Southern California, MIT CSAIL, Stanford Univ., 2007.

[16] Erik Haas. *Design, Evaluation and Implementation of a Multi-Carrier Transmission System for Aeronautical Communications*. Herbert Utz Verlag, Munich, Germany, 2003.

[17] Carl Hartung, Richard Han, Carl Seielstad, and Saxon Holbrook. FireWxNet: a multi-tiered portable wireless system for monitoring weather conditions in wildland fire environments. In *MobiSys '06: Proceedings of the 4th international conference on Mobile systems, applications and services*, pages 28–41, New York, NY, USA, 2006. ACM.

[18] Tian He, Sudha Krishnamurthy, Liqian Luo, Ting Yan, Lin Gu, Radu Stoleru, Gang Zhou, Qing Cao, Pascal Vicaire, John A. Stankovic, Tarek F. Abdelzaher, Jonathan Hui, and Bruce Krogh. VigilNet: An integrated sensor network system for energy-efficient surveillance. *ACM Transactions on Sensor Networks (TOSN)*, 2(1):1–38, 2006.

[19] Jeffrey Hightower and Gaetano Borriello. A survey and taxonomy of location systems for ubiquitous computing. Technical Report UW-CSE 01-08-03, University of Washington, Computer Science and Engineering, Seattle, WA, USA, August 2001.

[20] Ville A. Kaseva, Mikko Kohvakka, Mauri Kuorilehto, Marko Hnnikinen, and Timo D. Hmlinen. A wireless sensor network for RF-based indoor localization. In *EURASIP Journal on Advances in Signal Processing*, volume 2008, 2008.

[21] Stuart Kininmonth, Scott Bainbridge, Ian Atkinson, Eric Gill, Laure Barral, and Romain Vidaud. Sensor networking the Great Barrier Reef. *Spatial Sciences Queensland Journal*, pages 34–38, Spring 2004.

[22] Koen Langendoen and Niels Reijers. Distributed localization in wireless sensor networks: a quantitative comparison. *Computer Networks*, 43(4):499–518, 2003.

[23] HyungJune Lee, Martin Wicke, Branislav Kusy, and Leonidas Guibas. Localization of mobile users using trajectory matching. In *MELT '08: Proceedings of the first ACM International Workshop on Mobile Entity Localization and Tracking in GPS-less Environments*, pages 123–128, New York, NY, USA, 2008. ACM.

[24] Yanjun Li, Zhi Wang, and Yeqiong Song. Wireless sensor networks for wildfire monitoring. The Sixth World Congress on Intelligent Control and Automation (WCICA 2006), June 2006.

[25] M.E.M. Lijding, H.P. Benz, N. Meratnia, and P.J.M. Havinga. Smart Signs: Showing the way in smart surroundings. Technical Report TR-CTIT-06-20, Centre for Telematics and Information Technology, University of Twente, Enschede, 2006.

[26] Konrad Lorincz and Matt Welsh. Motetrack: a robust, decentralized approach to RF-based location tracking. *Personal Ubiquitous Computing*, 11(6):489–503, 2007.

[27] Alan Mainwaring, David Culler, Joseph Polastre, Robert Szewczyk, and John Anderson. Wireless sensor networks for habitat monitoring. In *WSNA '02: Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications*, pages 88–97, New York, NY, USA, 2002. ACM.

[28] Guoqiang Mao, Barış Fidan, and Brian D.O. Anderson. Wireless sensor network localization techniques. *Computer Networks*, 51(10):2529–2553, July 2007.

[29] Miklós Maróti, Branislav Kusy, Gyula Simon, and Ákos Lédeczi. The flooding time synchronization protocol. In *SenSys '04: Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems*, pages 39–49, New York, NY, USA, 2004. ACM.

[30] MIT Computer Science and Artificial Intelligence Lab, Cambridge, MA, USA. *Cricket v2 User Manual*, January 2005. http://cricket.csail.mit.edu/.

[31] David Moore, John Leonard, Daniela Rus, and Seth Teller. Robust distributed network localization with noisy range measurements. In *SenSys '04: Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems*, pages 50–61, New York, NY, USA, 2004. ACM.

[32] Nissanka B. Priyantha, Hari Balakrishnan, Erik D. Demaine, and Seth Teller. Mobile-assisted localization in wireless sensor networks. *Proceedings of IEEE INFOCOM '05*, 1:172–183, March 2005.

[33] Nissanka B. Priyantha, Anit Chakraborty, and Hari Balakrishnan. The Cricket location-support system. In *MobiCom '00: Proceedings of the 6th Annual International Conference on Mobile Computing and Networking*, pages 32–43, New York, NY, USA, 2000. ACM.

[34] Sven Rienstra and Serhat Gülçiçek. Wireless sensor networks. Bachelor's thesis, Saxion Hogeschool Enschede, Netherlands, June 2008.

[35] Andreas Savvides, Mani Srivastava, Lewis Girod, and Deborah Estrin. Localization in sensor networks. In *Wireless Sensor Networks*, chapter 15, pages 327–349. Kluwer Academic Publishers, Norwell, MA, USA, 2004.

[36] Spread Spectrum Scene. An introduction to indoor radio propagation. http://sss-mag.com/indoor.html, June 2001.

[37] Adam Smith, Hari Balakrishnan, Michel Goraczko, and Nissanka Bodhi Priyantha. Tracking Moving Devices with the Cricket Location System. In *2nd International Conference on Mobile Systems, Applications and Services (Mobisys 2004)*, Boston, MA, USA, June 2004.

[38] William Stallings. *Data and Computer Communications*. Prentice Hall, $8^{th}$ edition, 2007.

[39] Tsenka Stoyanova, Fotis Kerasiotis, Aggeliki Prayati, and George Papadopoulos. Evaluation of impact factors on RSS accuracy for localization and tracking applications. In *MobiWac '07: Proceedings of the 5th ACM International Workshop on Mobility Management and Wireless Access*, pages 9–16, New York, NY, USA, 2007. ACM.

[40] M. Terwilliger, A. Gupta, V. Bhuse, Z. Kamal, and M. Salahuddin. A localization system using wireless network sensors: a comparison of two techniques. In *Proceedings of the 1st Workshop on Positioning, Navigation and Communication (WPNC '04)*, Hanover, Germany, March 2004.

[41] Mark Terwilliger. *Localization in wireless sensor networks.* PhD thesis, Western Michigan University, Kalamazoo, MI, USA, April 2006.

[42] Brett Warneke, Matt Last, Brian Liebowitz, and Kristofer S. J. Pister. Smart Dust: Communicating with a cubic-millimeter computer. *Computer*, 34(1):44–51, 2001.

[43] Adel Amin Abdel Azim Youssef. *SALAM: a scalable anchor-free localization algorithm for wireless sensor networks.* PhD thesis, University of Maryland, College Park, MD, USA, 2006.

[44] Pei Zheng and Lionel Ni. *Smart Phone and Next Generation Mobile Computing.* Morgan Kaufmann, 2005.