Localization and fault detection in wireless sensor networks

Safdar Abbas Khan

UNIVERSITÉ PARIS-EST

Ecole Doctorale « Mathématiques et les sciences et techniques de l'information et de la communication » LISSI Laboratoire (EA-3956)

Thèse de doctorat

Spécialité : Informatique

Safdar Abbas KHAN

Localisation et détection d'erreurs dans les réseaux de capteurs sans fil

Soutenue publiquement le 16 décembre 2011 devant le jury composé de :

Rochdi MERZOUKI	Professeur à l'Université de Lille 1	Rapporteur
Yasser ALAYLI	Directeur du Laboratoire LISV, Versailles	Rapporteur
Yskandar HAMAM	Professor at Tshwane University of Technology, South Africa	Examinateur
Michel DIAZ	Directeur de recherche au LAAS de Toulouse	Examinateur
Johnson AGBINYA	Associate Professor at La Trobe University, Australia	Examinateur
Karim DJOUANI	Professeur à l'Université Paris Est Créteil	Directeur de thèse
Boubaker DAACHI	Maître de conférences à l'Université Paris Est Créteil	Co-directeur de thèse

Abstract

In this thesis we have covered three themes related to wireless sensor networks. The first one concerns the detection of measurement errors in sensor readings in a wireless sensor network. In order to identify a faulty node we have used soft computing techniques. A fuzzy inference system and a recurrent fuzzy inference system are used to model a node as far as its sensor measurement is concerned. The sensor measurement of a node is approximated by a function whose arguments are the real measurements of the neighboring sensors. The return of the function is the estimated value of the sensor measurement. The difference between the approximated value from the model and the actual measurement of the sensor is used as an indication for whether or not to declare a node as faulty.

Then we focus on the localization aspect of all the nodes in the network. Once the intermediate distances between the connected nodes have been calculated, the task remained to be accomplished is to find the position of all the nodes by using as minimum number of anchors as possible. Thus, we have proposed a localization method that uses exactly three anchor/beacon nodes. The motivation for the proposed localization scheme stemmed from the fact that a plane and hence all points on it are completely described by defining/knowing exactly three points. But how to integrate this idea in relation to localization in wireless sensor network is discussed in the second part, where we are able to attain the estimated position of all sensors by using only three anchors.

Finally we have focussed our attention on the power loss in a node signal due to voltage droop in the battery of the node. Since our proposed localization algorithm uses the strength in the signal from different nodes, paying attention to the received signal strength is crucial. When the battery of a node looses voltage, there is a decrease in the signal strength from that node. This decrease of strength can be interpreted as an increase in the distance between the respective nodes. In fact this is a misinterpretation of the RSS from localization point of view. Thus in the first part we propose a method to compensate for the apparent loss in signal power due

to voltage decrease and not due to increase in distance.

Contents

Chapter 1

General Introduction

In certain wireless sensor nodes localization strategies the intermediate distance between the nodes is obtained from the received signal strength (RSS). An error in RSS results in an incorrect estimation of distance between two nodes. As a consequence the approximated position of a node is far from the real position of the node. The obscurity in RSS due to voltage droop in the transmitter battery has not been addressed in the existing literature. So the problem is to overcome the inaccurate distance measurement resulting from erroneous RSS caused by energy loss in the transmitter battery.

Knowing the intermediate distance between the connected nodes is one of the initial steps in localization of nodes in a wireless sensor network. Reference points with known geographical coordinates are mandatory for the position estimation of the nodes. It means that the information of intermediate distances between the nodes is not sufficient for finding coordinates of the nodes. There is a need of landmarks with known locations such that the nodes will relatively localize themselves in relation to these known positions in addition to the intermediate distances amongst them and these landmarks. If some of the nodes are used as landmarks, then these nodes are termed as anchor nodes or simply anchors. If a node is not an anchor it is termed as tracked node.

A trivial solution to position estimation of a node is to know the distances between

the node and three anchors. It implies that the node has a connection to three anchors. With this approach the number of anchors exceeds the number of tracked nodes. If an anchor is equipped with a local/global positioning system, then increasing the number of anchors will increase the cost of the network. If an anchor node's position is preconfigured, then it implies a highly controlled topology of the network which is not applicable to a situation in which a WSN is randomly deployed. Thus in any case we need to decrease the number of anchor nodes. So the problem is to evolve a localization strategy that uses minimum possible number of anchors.

Once the nodes are deployed and their positions are calculated. The readings from the sensors are meaningful. That is we not only know a change in the physical quantity being measured but also the location where that change is detected. A question that arises is the reliance on the sensor measurement of a particular node. How can we be sure of the accuracy in sensor reading from a node? There is a possibility that a node has developed a faulty sensor. So the problem is to devise a strategy for the detection of faulty sensors in a wireless sensor network.

In this thesis we have addressed the above mentioned three issues. The literature survey shows that some of the research works were targeted to fulfill the inaccuracy in RSS due to attenuation in the signal. But no work was found that addressed the problem of inaccurate distance due to voltage droop in the battery of signal sending node.

Similarly the problem of localization has been carried out in multiple research works. In one of them the localization strategy is to divide the deployment area of the WSN in a regular grid and place an anchor at each vertex of the grid. Hence there is a need for more number of anchors. Then there are further techniques that have reduced the number of required anchors. These strategies require the anchors to be placed at the boundary of the network. Still the number of anchors is not the minimum.

The problem of fault detection has also been studied in various research works. Some of them have used a comparison of a sensor measurement with the neigh-

boring sensor measurements. If the sensor reading is similar to a certain number of neighboring sensor measurements then the particular sensor is declared as fault free. In one of the research work a recurrent neural network was used to have an approximation of a sensor measurement of a node. If the approximated value is quite different from the real sensor measurement, the node is declared to have a faulty sensor. A fault detection scheme using TSK fuzzy logic system has not been carried out.

Thus the contribution of the present thesis is three fold. We have proposed a method to overcome eventual localization errors arising from the decreasing energy in the batteries of the WSN nodes. We have developed a localization algorithm that uses exactly three nodes. Theoretically, this is the minimum number of reference points for localization in a plane. Finally we have presented a fault detection strategy that utilizes recurrent TSK fuzzy inference system. In this method, a sensor measurement of a node is approximated by a function whose arguments are the neighboring sensor measurements and the previous approximated value. If the difference between the approximated value and the real measurement is greater than the tolerated bound the sensor of that node is declared as faulty.

1.1 Wireless Sensor Networks

Recent technological advances have enabled the development of low-cost, low-power, and multifunctional sensor devices. These nodes are autonomous devices with integrated sensing, processing, and communication capabilities. A sensor is an electronic device that is capable of detecting environmental conditions such as temperature, sound, chemicals, or the presence of certain objects. Sensors are generally equipped with data processing and communication capabilities. The sensing circuitry measures parameters from the environment surrounding the sensor and transforms them into electric signals. Processing such signals reveals some properties of objects located and/or events happening in the vicinity of the sensor. The

Figure 1.1: MICAz sensor mote hardware (Image courtesy of Crossbow Technology [xbow, 2004d])

sensor sends such sensed data, usually via a radio transmitter, to a command center, either directly or through a data-collection station (a base station or a sink). To conserve the power, reports to the sink are normally sent via other sensors in a multihop fashion. Retransmitting sensors and the base station can perform fusion of the sensed data in order to filter out erroneous data and anomalies, and to draw conclusions from the reported data over a period of time. For example, in a reconnaissance-oriented network, sensor data indicates detection of a target, while fusion of multiple sensor reports can be used for tracking and identifying the detected target.

A wireless sensor network consists of a possibly large number of wireless devices able to take environmental measurements. Typical examples include temperature, light, sound, and humidity. These sensor readings are transmitted over a wireless channel to a running application that makes decisions based on these sensor readings. Many applications have been proposed for wireless sensor networks, and many of these applications have specific requirements that offer additional challenges to the application designer.

Figure 1.1 shows the latest-generation MICAz [xbow, 2004a, xbow, 2004c] sensor node. MICAz motes are equipped with an Atmel128L processor capable of maximum throughput of 8 millions of instructions per second when operating at 8

MHz. It also features an IEEE 802.15.4/Zigbee compliant RF transceiver, operating in the 2.4-2.4835-GHz globally compatible industrial scientific medical band, a direct spread-spectrum radio resistant to RF interference, and a 250-kbps data transfer rate. The MICAz runs on TinyOS [Hill et al., 2000] (v1.17 or later) and is compatible with existing sensor boards that are easily mounted onto the mote. A partial list of specifications given by the manufacturers of the MICAz mote is presented in table 1.1. Several advantages exist for instrumenting an area with a wireless sensor network [Agre and Clare, 2000]:

- Due to the dense deployment of a greater number of nodes, a higher level of fault tolerance is achievable in wireless sensor networks.
- Coverage of a large area is possible through the union of coverage of several small sensors.
- Coverage of a particular area and terrain can be shaped as needed to overcome any potential barriers or holes in the area under observation.
- It is possible to incrementally extend coverage of the observed area and density by deploying additional sensor nodes within the region of interest.
- An improvement in sensing quality is achieved by combining multiple, independent sensor readings. Local collaboration between nearby sensor nodes achieves a higher level of confidence in observed phenomena.
- Since nodes are deployed in close proximity to the sensed event, this overcomes any ambient environmental factors that might otherwise interfere with observation of the desired phenomenon.

1.2 Applications

Several applications have been envisioned for wireless sensor networks [Akyildiz et al., 2002b]. These range in scope from military applications to environment

Table 1.1: MICAz mote specification [xbow, 2004c]

Processor	Atmel ATMega128L @ 8 MHz
Program flash memory	128 kilobytes
Measurement serial flash	512 kilobytes
Configuration electrically erasable programmable read-only memory (EEPROM)	4 kilobytes
Serial communications	UART
Analog to digital converter	10 bit ADC
Other interfaces	Digital I/O, 12C, SPI
Processor current draw	8 mA in active mode < 1 μA in sleep mode

Frequency band	2400MHz to 2483,5MHz
Transmit (TX) data rate	250kbps
RF power	-24dBm to 0dBm
Receive sensitivity	-90dBm (min), -94dBm (typ)
Adjacent channel rejection	47 dB, +5-MHz channel spacing
	38 dB, -5-MHz channel spacing
Outdoor range	75m to 100m
Indoor range	20m to 30m
Radio current draw	19.7 mA in receive mode 11 mA (TX -10dBm) 14 mA (TX -5dBm) 17.4 mA (TX 0dBm) 20 μA in idle mode (voltage regulator on) 1 μA in sleep mode (voltage regulator off)

Battery	2 AA batteries
User interface	red, green, and yellow LED
Size	2.25×1.25×0.25 in (w/o battery pack)
Weight	0.7 oz (w/o batteries)
Expansion connector	51 pin

monitoring to biomedical applications.

1.2.1 Military applications

Wireless sensor networks can form a critical part of military command, control, communications, computing, intelligence, surveillance, reconnaissance, and targeting systems. Examples of military applications include monitoring of friendly and enemy forces; equipment and ammunition monitoring; targeting; and nuclear, biological, and chemical attack detection.

By equipping or embedding equipment and personnel with sensors, their condition can be monitored more closely. Vehicle-, weapon-, and troop-status information can be gathered and relayed back to a command center to determine the best course of action. Information from military units in separate regions can also be aggregated to give a global snapshot of all military assets.

By deploying wireless sensor networks in critical areas, enemy troop and vehicle movements can be tracked in detail. Sensor nodes can be programmed to send notifications whenever movement through a particular region is detected. Unlike other surveillance techniques, wireless sensor networks can be programmed to be completely passive until a particular phenomenon is detected. Detailed and timely intelligence about enemy movements can then be relayed, in a proactive manner, to a remote base station.

In fact, some routing protocols have been specifically designed with military applications in mind [Ye et al., 2002]. Consider the case where a troop of soldiers needs to move through a battlefield. If the area is populated by a wireless sensor network, the soldiers can request the location of enemy tanks, vehicles, and personnel detected by the sensor network (figure 1.2). The sensor nodes that detect the presence of a tank can collaborate to determine its position and direction, and disseminate this information throughout the network. The soldiers can use this information to strategically position themselves to minimize any possible casualties.

In chemical and biological warfare, close proximity to ground zero is needed for

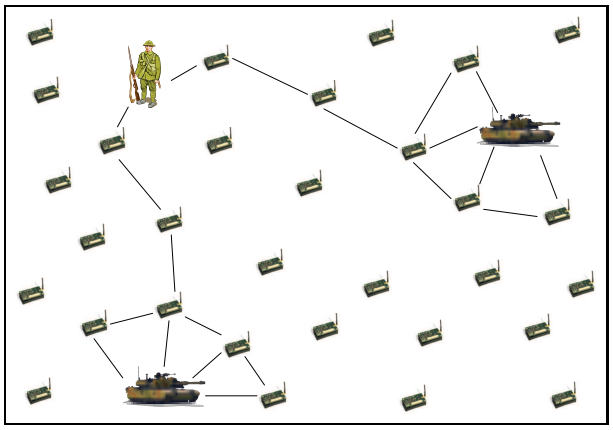


Figure 1.2: Enemy target localization and monitoring

timely and accurate detection of the agents involved. Sensor networks deployed in friendly regions can be used as early-warning systems to raise an alert whenever the presence of toxic substances is detected. Deployment in an area attacked by chemical or biological weapons can provide detailed analysis, such as concentration levels of the agents involved, without the risk of human exposure.

1.2.2 Environmental applications

By embedding a wireless sensor network within a natural environment, collection of long-term data on a previously unattainable scale and resolution becomes possible. Applications are able to obtain localized, detailed measurements that are otherwise more difficult to collect. As a result, several environmental applications have been proposed for wireless sensor networks [Agre and Clare, 2000, Akyildiz et al., 2002b]. Some of these include habitat monitoring, animal tracking, forest-fire detection, precision farming, and disaster relief applications.

Consider a scenario where a fire starts in a forest. A wireless sensor network deployed in the forest could immediately notify authorities before it begins to spread uncontrollably (figure 1.3). Accurate location information [Niculescu and Nath, 2001] about the fire can be quickly deduced. Consequently, this timely detection gives fire-fighters an unprecedented advantage, since they can arrive at the scene before the fire spreads uncontrollably.

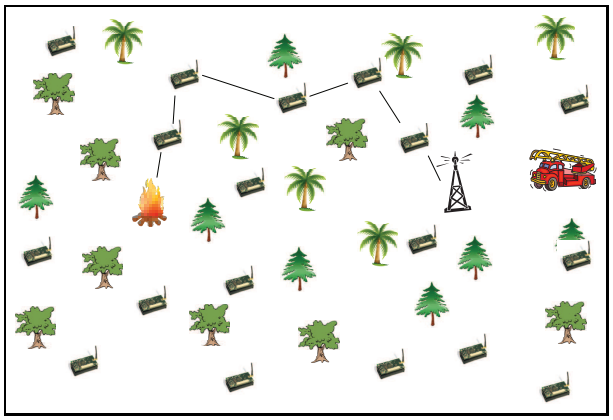


Figure 1.3: Forest-fire monitoring application

Precision farming [Sudduth, 1999] is another application area that can benefit from wireless sensor network technology. Precision farming requires analysis of spatial data to determine crop response to varying properties such as soil type [Locke et al., 2000]. The ability to embed sensor nodes in a field at strategic locations could give farmers detailed soil analysis to help maximize crop yield or possibly alert them when soil and crop conditions attain a predefined threshold. Since wireless sensor networks are designed to run unattended, active physical monitoring is not required.

Disaster relief efforts such as the ALERT flood-detection system [Bonnet et al., 2000] make use of remote field sensors to relay information to a central computer system in real time. Typically, an ALERT installation comprises several types of sensors, such as rainfall sensors, water-level sensors, and other weather sensors. Data from each set of sensors are gathered and relayed to a central base station.

1.2.3 Health Applications

Potential health applications abound for wireless sensor networks. Conceivably, hospital patients could be equipped with wireless sensor nodes that monitor the patients' vital signs and track their location. Patients could move about more freely while still being under constant supervision. In case of an accident – say, the patient trips and falls – the sensor could alert hospital workers as to the patients' location

and conditions. A doctor in close proximity, also equipped with a wireless sensor, could be automatically dispatched to respond to the emergency.

Glucose-level monitoring is a potential application suitable for wireless sensor networks [Schwiebert et al., 2001]. Individuals with diabetes require constant monitoring of blood sugar levels to lead healthy, productive lives. Embedding a glucose meter within a patient with diabetes could allow the patient to monitor trends in blood-sugar levels and also alert the patient whenever a sharp change in blood-sugar levels is detected. Information could be relayed from the monitor to a wristwatch display. It would then be possible to take corrective measures to normalize blood-sugar levels in a timely manner before they get to critical levels. This is of particular importance when the individual is asleep and may not be aware that their blood-sugar levels are abnormal.

1.2.4 PODS Project

Rare and endangered species of plants are threatened because they grow in limited locations. Evidently, these locations have special properties that sustain and support their growth. The PODS project [Biagioni, 2001, Biagioni and Bridges, 2002, PODS, 2000], located at Hawaii volcanoes National Park, consists of wireless sensor network deployed to perform long-term studies of these rare and endangered species of plants and their environment.

In Hawaii, the weather gradients are very sharp. In fact, regions of the island exist where rain forests and deserts are located less than 10 miles apart. Thus, it is not surprising that endangered species of plants are restricted to very small areas. Unfortunately, weather stations located throughout the island provide insufficient information for the areas where these endangered plants exist. Consequently, deploying a very dense wireless sensor network in the area of interest allows fine-grained temperature, humidity, rainfall, wind, and solar radiation information to be obtained by researchers.

These are just a few applications of wireless sensor networks. There are many

other applications in which wireless sensor networks are deployed and each one is designed according to the requirement of the application.

1.3 WSN Services

Most large-scale wireless sensor network applications share common characteristics. Services such as time synchronization, location discovery, data aggregation, data storage, topology management, and message routing are employed by these applications.

1.3.1 Time Synchronization

Time synchronization is an essential service in wireless sensor networks [Sivrikaya and Yener, 2004]. In order to properly coordinate their operations to achieve complex sensing tasks, sensor nodes must be synchronized. A globally synchronized clock allows sensor nodes to correctly time-stamp detected events. The proper chronology, duration, and time span between these events can then be determined. Incorrect time stamps, due to factors such as hardware clock drift, can cause the reported events relayed back to the base station to be assembled in incorrect chronological order.

Time synchronization is crucial for efficient maintenance of low-duty power cycles. Sensor nodes can conserve battery life by powering down. When properly synchronized, nodes are able to turn themselves on simultaneously. When powered up, sensor nodes can relay messages to the base station and subsequently power down again to conserve energy. Unsynchronized nodes result in increased delays while they wait for neighboring nodes to turn their radios on, and in the worst case, messages transmitted can be lost altogether. Various aspects in relation to time synchronization are discussed in [Elson et al., 2002, Sichitiu and Veerarittiphan, 2003, Mills, 1991, Mattern, 1989, Lamport, 1978, J. van Greunen and Rabaey, 2003, Ganeriwal et al., 2003, Fidge, 1988a, Fidge, 1988b, Elson and Estrin, 2001, Dai and

Han, 2004, Chandy and Lamport, 1985].

1.3.2 Location Discovery

Location discovery involves sensor nodes deriving their positional information, expressed as global coordinates or within an application-defined local coordinate system. The importance of location discovery is widely recognized [Savvides et al., 2002, Savvides et al., 2001, Niculescu and Nath, 2003a, Niculescu and Nath, 2003c, Niculescu and Nath, 2001, Meguerdichian et al., 2001]. It serves as a fundamental basis for additional wireless sensor network services where location awareness is required, such as message routing. Furthermore, in applications such as fire detection, it is generally not sufficient to determine *if* a fire is present, but more importantly, *where*. A brief review of location discovery solutions is discussed in chapter 2.

1.3.3 Data Aggregation

Data aggregation and query dissemination are important issues in wireless sensor networks [Heidemann et al., 2001]. Sensor nodes are typically energy constrained. Therefore, it is desirable to minimize the number of messages relayed, because radio transmissions can quickly consume battery power. A naive approach to reporting sensed phenomenon is one where all (raw) sensor reading are relayed to a base station for off-line analysis and processing. However, since sensor nodes within the same vicinity often detect the same, common phenomenon, it is likely some redundancy in sensor readings will occur [Krishnamachari et al., 2002]. Local collaboration allows nearby sensor nodes to filter and process sensor reading before transmitting them to a base station. Consequently, this process can reduce the number of messages relayed to the base station.

1.3.4 Data Storage

Data storage presents a unique challenge to developers. Event information collected by individual nodes must be stored at some location, either in situ or externally. In some cases, where an off-line storage area is not available, data must be stored within the wireless sensor network. Ratnasamy et al. [Ratnasamy et al., 2002, Ratnasamy et al., 2003] describe three data-storage paradigms employable in wireless sensor networks:

External Storage In this model, when a node detects an event, the corresponding data are relayed to some external storage located outside the network, such as a base station. The advantage of this approach is that queries posed to the network incur no energy expenditure since all data are already stored off-line.

Local Storage In this model, when a node detects an event, event information is stored locally at the node. The advantage of this approach is that no initial communication costs are incurred. Queries posed to the wireless sensor network are flooded to all nodes. The nodes with the desired information relay their data back to the base station for further processing.

Data-Centric storage In this model, event information is routed to a predefined location, specified by a geographic hash function (GHT), within the wireless sensor network. Queries are directed to the node that contains the relevant information, which relays the reply to the base station for further processing.

1.3.5 Topology Management and Message Routing

Wireless sensor networks can possibly contain hundreds or thousands of nodes. Routing protocols must be designed to achieve an acceptable degree of fault tolerance in the presence of sensor node failures, while minimizing energy consumption. Furthermore, since channel bandwidth is limited, routing protocols should be designed to allow for local collaboration to reduce bandwidth requirements.

Observations made in [Tilak et al., 2002] show that, although intuitively it appears a denser deployment of sensor nodes renders a more effective wireless sensor network, if the topology is not carefully managed, this can lead to a greater number of collisions and potentially congest the network. As a result, there is an increased amount of latency when reporting results and a reduction in the overall energy efficiency of the network. Furthermore, as the number of reported data measurements increases, the accuracy requirements of the application may be surpassed. This increase in the reporting rate by the deployed sensor nodes can actually harm the wireless sensor network performance, rather than prove beneficial.

Message-routing algorithms in ad hoc networks can be separated into two broad categories: greedy algorithms and flooding algorithms [Bose et al., 2001]. Greedy algorithms apply a greedy path-finding heuristic that may not guarantee a message reaches its intended receiver. One example of greedy routing, proposed by Finn in 1987, is forwarding to a neighbor that is closest to the destination. Additional steps are required to ensure the message is received by its intended recipient. Flooding algorithms employ a controlled packet duplication mechanism to ensure every node receives at least one copy of the message. For these algorithms to terminate, nodes in the sensor network must remember which messages have been previously received.

1.4 Sensor Operating Systems

TinyOS is an open-source operating system designed for wireless embedded sensor networks [Hill et al., 2000, Tin, 2004a]. It features a component-based architecture that enables implementation of sensor network applications. TinyOS features a component library that includes network protocols, distributed services, sensor drivers, and data-acquisition tools. TinyOS features an event-driven execution model and enables fine-grained power management. It has been ported to several platforms with support for various sensor boards.

Currently, over 500 research groups and companies use TinyOS and the sensor

Table 1.2: TinyOS Research Projects

Project	Description
Calamari [Calamari, 2004]	Localization solutions for sensor networks
CotsBots [CotsBots, 2004]	Inexpensive and modular mobile robots built using off-the-shelf components to investigate distributed sensing and cooperation algorithms in large (> 50) robot networks
Firebug [FireBug, 2004]	Berkeley civil engineering project for the design and construction of a wildfire instrumentation system using networked sensors
TinyGALS [TinyGALS, 2004]	Globally asynchronous and locally synchronous model for programming event-driven embedded systems
galsC [GalsC,]	Language and compiler designed for use with the TinyGALS programming model
Mate [Mate, 2004]	Application-specific virtual machines for TinyOS networks
PicoRadio [PicoRadio, 2004]	Development of mesoscale low-cost transceivers for ubiquitous wireless data acquisition that minimizes power/energy dissipation
TinyDB [TinyDB,]	Query processing system for extracting information from a network of TinyOS sensors

motes developed by Crossbow [xbow, 2004b]. A partial list of research projects [Tin, 2004b] currently under way is presented in table 1.2.

1.5 Thesis Outline

After the brief introduction to wireless sensor networks, we shall now give an outline of our work in this thesis. Our work is divided in three parts. In the first part, we deal with the decrease in the strength of a signal from a node due to loss of battery power of the node. Each node in a wireless sensor network is capable of receiving and transmitting signals. So the transceiver of a node is using the battery energy for sending and receiving the signals. As time passes, the battery energy keeps on decreasing. So there is lesser and lesser energy available to the transmitter of the node to send signals. As a consequence, the strength of the signal too keeps decreasing.

Moreover as the distance between the transmitter and receiver increases, the power in the signal at the receiving end decreases. Thus a decrease in the received signal strength (RSS) from a particular node could have two explanations: It could either be due to the increase in distance between the transmitting node and the receiver node; or it could be due to the loss of battery at the transmitting node. In the applications where the distance is obtained by analyzing the RSS, the change in RSS due to energy drooping of the battery can cause erroneous results. For example the localization algorithm, (like many other localization techniques) that we have proposed, uses an RSS-distance model to calculate the distance between the concerned nodes.

Hence the change (decrease) in the RSS due to the change (decrease) in the battery voltage of the sending node would lead to misinterpretation in terms of increase in the distance between the nodes. Eventually it would result in an erroneous estimation about the node position. Thus in the first part of the thesis, we tackle the problem of avoiding the misinterpretation of increase in distance originating from the voltage droop in the transmitting node battery.

In the second part of the thesis, we have proposed a localization algorithm that uses minimum possible reference points to find the position of all the nodes in the wireless sensor network. As a reference point we are using anchor nodes. An anchor node is a node that is aware of its local/global geographical coordinates. We have demonstrated that three anchors is a necessary and sufficient condition for finding all the nodes in a wireless sensor network where the nodes form a point set triangulation. Many research works have been conducted in order to minimize the number of reference points and many of them require these reference points to be at the boundary of the network. In our proposed localization technique, we have no such condition. Any three randomly chosen nodes in the network can serve as anchors, irrespective of their location in the network. We have developed a heuristic technique to find out the initial layout of the nodes just by using the information of connectivity amongst them, that is, we find the topology of the network by using

only the distance matrix. Then by knowing the coordinates of any three nodes, we can estimate the coordinates of the rest of the nodes. The key point is to find the symmetry, orientation and position of the topology that is in accordance to the known coordinates of the three anchors.

The third part of the thesis deals with the detection of the faulty sensors in a wireless sensor network. After the deployment of a wireless sensor network, there is always a possibility that some of the nodes would develop a malfunctioning sensor. In order to rely on the sensor reading of a node, it is very important to have the information about its current health status, since it is very likely that the sensor is not giving accurate readings at all times. Thus we have developed a fault detection scheme to identify malfunctioning sensors. We achieve this goal by using a soft computing technique, that is, we model each sensor by fuzzy logic system. The sensor measurement of a node is approximated by the fuzzy logic system, whose input is the real sensor measurements of the neighboring nodes. If the difference between the approximated value and the real measurement of a node is greater than the accepted tolerance, the node is declared as faulty. We have also developed a recurrent model whose input also include the previously approximated values.

The thesis is organized as follows: In chapter 2 we discuss the general techniques used for the localization in wireless sensor networks. In chapter 5 we present the voltage drooping problem in relation to distance estimation amongst the nodes. Chapter 4 deals with the detailed description of the proposed localization algorithm. Chapter 3 is dedicated to the discussion of fault detection in wireless sensor networks. Finally chapter 6 presents the conclusion of our work and the perspective for future research.

Chapter 2

Location Estimation Methods

This chapter reviews three methods that can be used in an IEEE 802.15.4 network to determine the location of an object. The first one uses received signal strength (RSS) as a simple way of estimating the distance between nodes. The second approach takes advantage of the signal angle of arrival, if known, at two or more nodes to estimate location of the node that transmitted the signal. The last method measures the time difference of signal arrival at multiple nodes with known locations to estimate the location of the node of interest. Among these three methods, the RSS-based location estimation has received the most attention because of its minimum hardware requirements and the simplicity of its implementation.

2.1 Introduction

One of the applications of short-range wireless networking is determining the approximate physical location of objects at any given time. The real-time knowledge of the location of personnel, assets, and portable instruments can increase management efficiency. *Location estimation* refers to the process of obtaining location information on a node with respect to a set of known reference positions. The location estimation is also referred to as *positioning*, *locationing* , and *geolocationing*. The knowledge of the location of the nodes presents the opportunity of providing location-dependent

services. For example, a visitor in a museum can carry an audio/video device that provides relevant information to the visitor, depending on his or her location in the museum. The location of a node can also be used as part of the authentication process. In this way, the authenticity of a packet is determined not only by the information embedded in the packet but also by the location of the node that transmitted the packet.

Here we focus on the location-estimation methods that use short-range radio frequency (RF) signals. However, it is possible to use other types of signals such as ultrasound or infrared instead of an RF signal in a location-estimation algorithm; but RF-based positioning systems are also found to be more suitable for large-scale deployments.

The location-estimation systems developed using short-range wireless networking are sometimes referred to as *local positioning systems* (LPSs) to differentiate them from *global positioning systems* (GPSs). A GPS-enabled device determines its location by calculating its distance from three or more GPS satellites orbiting the Earth. Each GPS satellite continuously transmits a message containing the satellite location and the exact time. This message travels approximately with the speed of the light to reach the GPS receiver. The GPS receiver compares the exact time the message was received with the time the message was transmitted by the satellite to calculate the distance traveled. Knowing the distance to at least three satellites and the satellites positions, the receiver calculates its own position. The LPS, in contrast, does not use information provided by GPS satellites or any other long-range transmitter. An LPS uses the RF signals transmitted by local nodes with known positions or the mobile node itself to calculate the location of the mobile node relative to the known locations of other local nodes.

The choice of location-estimation algorithm depends on the application scenario. The location-estimation methods are compared based on their performance and complexity. The location accuracy, which is the distance between the actual location and the estimated location, is the most intuitive performance metric.

The location estimation usually involves two groups of nodes. The first group consists of nodes with known locations. These nodes, sometimes referred to as *anchor nodes*, are used as references for the location estimation. The location of the anchor nodes can be determined by the installer, or the anchor nodes may be equipped with GPS to determine their own locations.

The second group is the nodes with unknown locations, referred to as *tracked nodes*. The purpose of the location estimation is to determine the location of the tracked nodes with the help of the anchor nodes.

The basic idea of local positioning can be summarized as follows. A tracked node with unknown location emits a signal, which is received by the neighboring anchor nodes. The anchor nodes measure the *received signal strength* (RSS), the *time of arrival* (ToA), or the *angle of arrival* (AoA) of the received signal. These measured values are used as inputs to an algorithm that determines the approximate location of the tracked node. The algorithms normally use only one of these three inputs.

There are two types of processing approaches for position estimation of the nodes. These are *central* and *distributed* processing approaches. In central processing approach, a single node, referred to as the *central location processing node*, is dedicated to executing the location-estimation algorithm. All other nodes in the network only gather the location-related information such as RSS and send it to the central location processing node. The central location processing node calculates the estimated location of all the tracked nodes and communicates the calculated location back to each tracked node if requested. In distributed processing approach the task of the location-estimation is distributed among almost all the nodes in the network. In this way, there is no centralized location processing node and each node determines its own location by communicating only with nearby anchors nodes and potentially other tracked nodes.

2.2 RSS-Based Locationing Algorithms

The received signal strength (energy) can be measured for each received packet. The measured signal energy is quantized to form the *received signal strength indicator* (RSSI). The RSSI and the time at which the packet was received (timestamp) are available to MAC, network, and application layers for various types of analysis. The simplest method to determine the location of a tracked node is to request that the tracked node transmit a signal. Then the location of the reference node that reports the highest RSSI is considered the estimated location of the tracked node. The advantage of this method is that it can be implemented easily on low-cost, battery-powered nodes with small memory size and low processing capabilities. However, the location-estimation accuracy of this method can be inadequate for many applications. The only way to improve the accuracy of this method is to increase the number of anchor nodes, which is not a desired approach in low-cost applications. The following section presents another simple RSSI-based locationing method.

2.2.1 RSSI-Based Location Estimation Using Trilateration

Figure 2.1 shows a location-estimation scenario where there are three anchor nodes (1,2, and 3)and the fourth node is the tracked node. The goal is to determine the estimated two-dimensional location of the tracked node. Two-dimensional (2D) means only X and Y coordinates of the node position will be estimated. But the same concept can be extended to three-dimensional (3D) space as well. The location estimation begins with the tracked node transmitting a signal with a predefined output power. Assuming that all nodes have omnidirectional antennas, each one of the anchor node can estimate the distance r_i for $1 \leq i \leq 3$ between itself and the tracked node using the RSS-distance model [Seidel and Rappaport, 1992].

$$P_R = P_T - 10n\log(f) - 10n\log(r) + 30n - 32.44(\text{dBm})$$

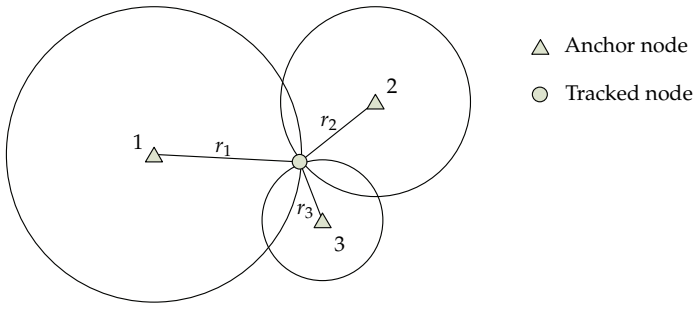


Figure 2.1: Location estimation using trilateration

where P_T is the transmitted power (in dBm) by the tracked node, P_R is the RSS at the anchor node location, f is the transmitted signal frequency in MHz, n is the path-loss exponent, and r is the distance in meters.

Anchor 1, for example, can estimate the distance (r_1) between its location and the location of the tracked node using RSS. From the single measurement done by anchor 1, the only conclusion that can be made is that the tracked node is located on the perimeter of a circle with radius r_1 and center at anchor 1. Using the Euclidean distance, we can write:

$$(X_1 - X)^2 + (Y_1 - Y)^2 = (r_1)^2$$

or

$$(X_1 - X)^2 + (Y_1 - Y)^2 - (r_1)^2 = 0$$

where (X_1, Y_1) and (X, Y) are coordinates for anchor 1 and the tracked node, respectively. Similar equations can be derived for anchor 2 coordinates (X_2, Y_2) and anchor 3 coordinates (X_3, Y_3). Therefore, to find the location of the tracked node we need to find (X, Y) that satisfies (2.1).

$$\begin{bmatrix} (X_1 - X)^2 + (Y_1 - Y)^2 \\ (X_2 - X)^2 + (Y_2 - Y)^2 \\ (X_3 - X)^2 + (Y_3 - Y)^2 \end{bmatrix} - \begin{bmatrix} (r_1)^2 \\ (r_2)^2 \\ (r_3)^2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \tag{2.1}$$

This method of determining the relative location of nodes using the geometry of intersection of three circles is referred to as trilateration.

This simple RSSI-based location estimation can also be used when there are more than three anchors involved. In this way the signal transmitted by the tracked node will be received by several nodes instead of only three nodes. The number of rows in (2.1) is proportional to the number of anchors participating in location estimation. Increasing the number of anchors may improve the location-estimation accuracy in some applications. It is also possible to engage only the nearby nodes in location estimation. The RSSI value of the packet received by each anchor node indicates the distance between the nodes. If an anchor node receives a packet from the tracked node as part of the location-estimation process, the anchor node only participates in the location estimation if the RSSI of the received packet is above a certain limit. By modifying the RSSI limit, one can increase or decrease the number of anchors nodes participating in the location estimation.

2.2.2 Location Estimation Based on Location Fingerprinting

Location estimation based on location fingerprinting is implemented in two phases. The first phase requires a site survey (offline training) to generate a database of measured RSSI values of the signals from the anchor nodes at certain locations. In the second phase (the real-time phase), each tracked node is capable of determining its own location by comparing the real-time measured RSSI of the signals received from the anchor nodes with the corresponding RSSI information available in its database [Seidel and Rappaport, 1992]. The basic concept of this method is shown in figure 2.2. The anchor nodes are numbered from 1 to 9. These anchors have overlapping coverage and form a *grid*. The physical distances between the nodes are not necessarily equal. During the first (training) phase, a receiver is placed at each predetermined location L_1 to L_6, and the RSSI of the received signal from anchors 1 to 9 are measured and stored in an array. For example, at location L_1, the array containing the received signal strength is the following:

$$ss_{L_1} = [ss_{L_1,1} \quad ss_{L_1,2} \quad \cdots \quad ss_{L_1,9}]$$

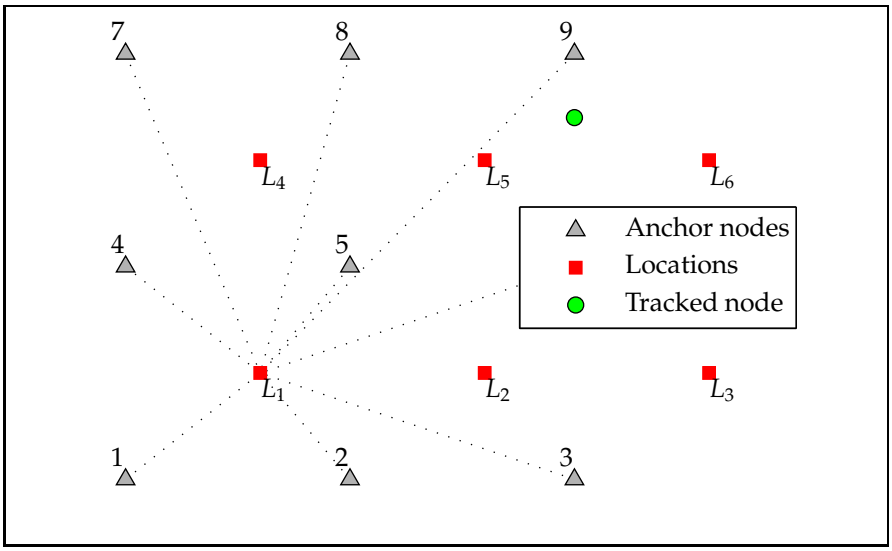


Figure 2.2: Location estimation based on fingerprinting

where $ss_{L_1,i}$ is the strength of the signal received from the anchor node i at location L_1. The database containing the signal strength information associated with all locations L_1 to L_6 is referred to as the *radio map*. In practice, the signal strength at known locations are measured multiple times, and the signal strength array contains the statistical average of the strength of the signals received from the anchors. The array of signal strength values at each location is known as the *fingerprint* (or *RF signature*) of that location.

After completion of the training phase, a tracked node as shown in figure 2.2 can determine its own location by going into receive mode and receiving the signals transmitted from each anchor node. The strength of each signal is calculated and stored in an array associated with the tracked-node current location:

$$ss_{\text{current}} = [ss_{\text{current},1} \quad ss_{\text{current},2} \quad \cdots \quad ss_{\text{current},9}]$$

where $ss_{\text{current},i}$ is the strength of the signal received from the anchor node i at the current location of the tracked node. The Euclidean distance can be used to determine the distance (difference) between the current signal strength array measured during the real-time phase and the signal strength array associated with each known loca-

tion. For example, the Euclidean distance between the ss_{L_1} and ss_{current} is calculated from

$$d\left(ss_{\text{current}}, ss_{L_1}\right) = \sqrt{\sum_{i=1}^{9}\left(ss_{\text{current}} - ss_{L_1,i}\right)}$$

where $d\left(ss_{\text{current}}, ss_{L_1}\right)$ is the distance between these arrays. This distance is not a physical distance and is only an indication of the similarity of ss_{L_1} and ss_{current} signal strength arrays.

The simplest method for determining the location of the tracked node in the real-time phase is the *single nearest-neighbor* technique. In this method the tracked node calculates the Euclidean distance between the real-time measured signal strength array ss_{current} and the signal strength array assiciated with the locations L_1 to L_6. The location of the tracked node is simply estimated to be equal to one of these six known locations, where ss_{L_j} has minimum distance to the ss_{current}, i.e.,

$$\text{estimated position} = L_j \text{ s.t. } d(ss_{\text{current}}, ss_{L_j}) = \min_{1\le k\le 6}\{d\left(ss_{\text{current}}, ss_{L_k}\right)\}$$

The advantage of the single nearest neighbor method is its simplicity, but it does not take advantage of the available *ss* arrays associated with the rest of the known locations to improve the location-estimation accuracy.

The *k-nearest neighbor* (KNN) method, shown in figure 2.3, can be used instead of the single nearest neighbor to improve the location-estimation accuracy. In this method, the tracked node identifies k known locations for which their *ss* array has the lowest distance to ss_{current}. In the KNN technique, the estimated location of the tracked node is the average of these k known locations:

$$\begin{aligned} X_E &= \frac{1}{k}\sum_{i=1}^{k} X_i \\ Y_E &= \frac{1}{k}\sum_{i=1}^{k} Y_i \end{aligned}$$

where (X_E, Y_E) is the estimated location of the tracked node and X_1, Y_1 to X_k, Y_k are

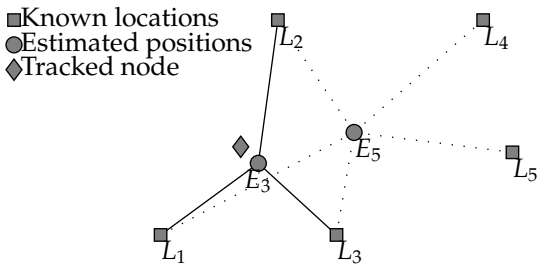


Figure 2.3: Effect of increasing k in KNN

the coordinates of the k-nearest neighbors.

For example, assuming that the location L_1 in figure 2.3 has the ss array with the smallest distance to ss_{current}, and L_2 and L_3 have the next two closest arrays:

$$d\left(ss_{\text{current}}, ss_{L_1}\right) < d\left(ss_{\text{current}}, ss_{L_2}\right) < d\left(ss_{\text{current}}, ss_{L_3}\right) \tag{2.2}$$

then in the nearest-neighbor method, L_1 will be the estimated location of the tracked node. In k-nearest neighbors with $k = 3$, the estimated location of the tracked node is E_3 in figure 2.3, which is closer to the actual location of the tracked node compared to the estimate provided by the nearest-neighbor method.

Increasing the value of k will not necessarily improve the location-estimation accuracy. For example, in figure 2.3 the estimated location when k is equal to 3 results in better estimation than $k = 5$. The reason is that by increasing the value of k, the further-away nodes are taken into account and may increase the estimation error.

The *weighted k nearest-neighbor* method can further improve the location-estimation accuracy of the KNN technique. In the KNN approach, all selected k neighbors (regardless of the distances of their associated ss arrays from the ss_{current} array) are treated equally in determining the estimated location. Ignoring the differences between these neighbors can be a source of error because the tracked node may be closer to some neighbors than others and this information will be lost in simple averaging of the location of all k-nearest neighbors. In the weighted k nearest neighbor

method, the distance of ss array associated with each k nearest neighbor from the ss_{current} is taken into account in estimating the location of the tracked node. That is the fingerprint with less distance to ss_{current} is assigned more weight than to the fingerprint with more distance. That is if w_i denotes the weight assigned to the fingerprint with location L_i, then from (2.2) we have:

$$w_1 > w_2 > w_3.$$

Hence the estimated position of the tracked node is given by

$$\begin{aligned} X_E &= \frac{1}{D}\sum_{i=1}^{k} X_i w_i \\ Y_E &= \frac{1}{D}\sum_{i=1}^{k} Y_i w_i \end{aligned}$$

where

$$D = \sum_{i=1}^{k} w_i$$

The basic concept of location estimation using fingerprinting can be seen as providing a database of known information to a system and expecting the system to learn how to relate the RSS information to a specific physical location. Therefore, the algorithms developed for other disciplines such as machine learning, neural networks, and pattern recognition can be used for fingerprinting-based location estimation as well.

2.2.3 Cooperative Location Estimation

In cooperative location estimation, not only are the distances from the tracked node to the anchor nodes measured but also the relative distances of the tracked nodes to each other are used as part of location estimation. Figure 2.4 highlights the difference between the basic trilateration method and cooperative technique. In trilateration method the location of the tracked node A is determined using range estimation

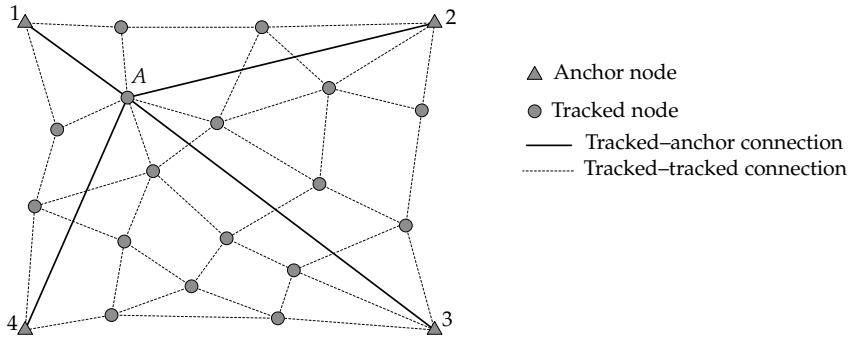


Figure 2.4: Cooperative location estimation

between the node A and the anchor nodes 1 to 4. The other tracked nodes with unknown locations do not participate in determining the location of the tracked node A. Every time a node needs to determine its own location using trilateration, only the tracked node itself and the nearby anchor nodes will participate in localization.

In the cooperative method the location of several tracked nodes can be determined concurrently using an iterative method. First, the RSSI measurements at the anchor nodes provide an estimate for the location of the tracked nodes participating in cooperative localization. Then each tracked node determines its approximate distance to the neighboring tracked nodes using the RSSI. The approximate distance between the tracked nodes is the additional information available in the cooperative method, which helps refine the location-estimation accuracy beyond the achievable accuracy in a basic trilateration method.

In a trilateration method, increasing the number of anchor nodes in a given area results in an improvement in location accuracy. But increasing the number of tracked nodes does not have any positive effect on accuracy of the trilateration technique. In the cooperative method, on the other hand, increasing either the number of anchor nodes or the number of tracked nodes can result in improvement in location-estimation accuracy.

2.2.4 Ad Hoc Positioning System

Niculescu and Nath [Niculescu and Nath, 2001] propose their ad hoc positioning system (APS), whereby nodes determine their location in reference to *landmarks* that are location aware. Landmarks can be other sensor nodes, base stations, or beacons that have positional information. Unlike GPS, where direct line of sight is required with a series of satellites in order to triangulate a location, landmark information is propagated through the wireless sensor network in a multihop fashion.

When an arbitrary node in the wireless sensor network has distance estimates to three or more landmarks, it computes its own position in the plane. The node utilizes the centroid of the landmarks as its location estimate. Nodes in direct communication with a landmark infer their distance from it based on the received signal strength of the landmark.

Through message propagation, nodes two hops away from a landmark estimate their distance based on the distance estimates of nodes located next to the landmark. The propagation schemes proposed by the authors eventually flood the entire network until all nodes are able to determine their coordinates.

2.3 Angle-of-Arrival Based Algorithms

At the expense of additional complexity and cost, it is possible to modify a node to become capable of determining the received signal *angle of arrival* (AoA). Figure 2.5 describes the basic concept of location estimation based on AoA. The anchor nodes transmit signals using omnidirectional antennas. The tracked node receives the signals from the nearby anchor nodes and can measure the received signal AoA. If the tracked node knows its own orientation, only two anchor nodes are required to determine the location of the tracked node. A node knows its orientation if it is aware of the North direction or a direction commonly known by the anchor nodes and the tracked node. Figure 2.5 shows a scenario in which the tracked node is unaware of its own orientation and therefore must receive the signal from at least

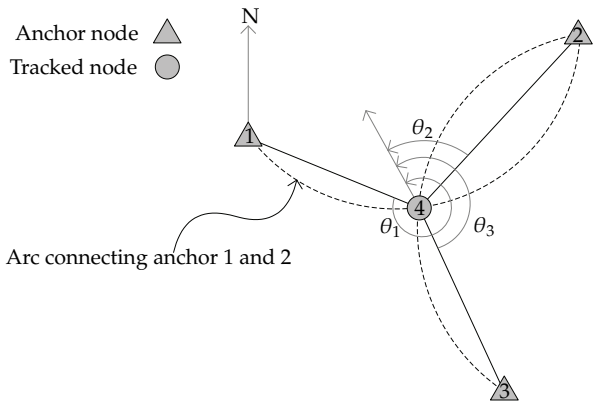


Figure 2.5: Location estimation using AoA

three anchors to be able to determine its own location. Although the tracked node does not know its orientation, it can calculate the angle between nodes 1, 4, and 2:

$$\angle 142 = 2\pi - (\theta_1 - \theta_2)$$

If the tracked node knows the AoA for only nodes 1 and 2, the location of the tracked node can be anywhere on an arc connecting nodes 1 and 2. By measuring the AoA of the signal received from anchor 3 as well, the tracked node can calculate the angle between nodes 2, 4, and 3:

$$\angle 243 = \theta_3 - \theta_2$$

Since the locations of the anchors are known, node 4 (the tracked node) can determine the arcs corresponding to its angle with nodes 1, 2, and 3. The intercept of the two arcs, shown in figure 2.5, is the location of node 4.

2.4 Time-Based Algorithms

The time-based and RSSI-based locationing algorithms have a common goal: determining the distance between the nodes based on the properties of the received signal. In the RSSI-based method, the received signal strength and the path-loss properties of the environment are used to estimate the distance. In time-based locationing

algorithms, the estimated propagation sped of the signal and the time it takes for the signal to travel from the transmitter to the receiver are used to determine the the distance between the nodes. GPS is an example of time-based locationing.

A time-based location estimation can be based on either the received signal *time of arrival* (ToA) or the *time difference of arrival* (TDoA). The ToA, shown in figure 2.6 required synchronization between the receiver and transmitter. The ToA is the

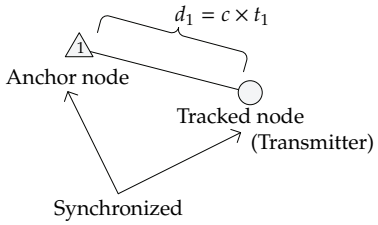


Figure 2.6: Estimating the distance using ToA

absolute value of the signal time of flight from the transmitter to the receiver. The distance from the tracked node to the anchor node (d_1) can be derived from the ToA (t_1) and the propagation speed (e.g., c = speed of light). The TDoA requires only synchronization of the receivers. The anchor nodes receive the signal transmitted by the tracked node, and the difference between the signal arrival times at these two anchor nodes can be used to calculate the Δd, which is the difference between the distance of d_1 and d_2. The TDoA requires participation of at least three anchor nodes to locate the position of the tracked node.

For the nodes shown in figure 2.7, we can formulate (2.3), where (X_1, Y_1), (X_2, Y_2),

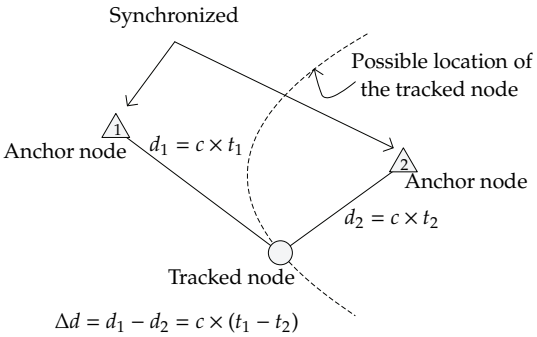


Figure 2.7: Determining the Δd based on TDoA

and X_E, Y_E are the coordinates of anchor node 1, anchor node 2 and the estimated

location of the tracked node, respectively.

$$\left.\begin{aligned} d_1^2 &= (X_1 - X_E)^2 + (Y_1 - Y_E)^2 \\ d_2^2 &= (X_2 - X_E)^2 + (Y_2 - Y_E)^2 \\ \Delta d &= \sqrt{(X_1 - X_E)^2 + (Y_1 - Y_E)^2} - \sqrt{(X_2 - X_E)^2 + (Y_2 - Y_E)^2} \end{aligned}\right\} \tag{2.3}$$

If only two anchor nodes participate in TDoA locationing, the only conclusion that can be made from (2.3) is that the tracked node is located on a hyperbolic curve, shown as a dashed line in figure 2.7. When a third anchor node is added, the estimated location of the tracked node will be intersection of the corresponding hyperbolic curves.

2.4.1 APS Using AoA

In [Niculescu and Nath, 2003a], Niculescu and Nath present two algorithms, DV-Bearing and DV-Radial, that allow sensor nodes to get a bearing and a radial in relation to a landmark using AoA to derive position information. The term "bearing" refers to an angle measurement with respect to another object. A "radial" refers to a reverse bearing which is simply the angle at which an object is seen from another location. The term "heading" refers to the sensor node's bearing with respect to true north and represents its absolute orientation.

AoA sensing requires sensor nodes to be equipped with an antenna array or several ultrasound receivers. This equipment is currently available in small package formats for wireless sensor network nodes such as the one developed for the Cricket Compass Project [Priyantha et al., 2001, Priyantha et al., 2000]. The theory of operation is based on TDoA and phase difference of arrival. If a node sends an RF signal and an ultrasound signal at about the same time, the receiving node can infer the distance between the sender and itself by measuring the time difference between the arrival of the RF signal and the ultrasound signal. To derive the angle of arrival of the signal, the receiving sensor node uses two ultrasound receivers placed at a

known distance from each other.

2.5 Conclusion

In this chapter we have given a brief overview of the basic localization methods that have been used in location based services in wireless sensor networks. Almost all the nodes in a wireless sensor network are capable of measuring RSS, therefore it is more economical approach for estimating inter-node distances than the approaches using angle of arrival or time based algorithms. We have also seen that in all the localization techniques there is a need for multiple number of anchor nodes except for some cooperative localization methods where the number of anchor nodes is minimum. Still number of anchors is not the minimum. In chapter 4 we shall show that the process of localization can be completed by using only three anchors.

Chapter 3

Detection of Measurement Errors

The goal of this chapter is to present a fault detection strategy for wireless sensor networks. Our scheme is based on modeling a sensor node by Takagi-Sugeno-Kang (TSK) fuzzy inference system (FIS), where a sensor measurement of a node is approximated by a function of the sensor measurements of the neighboring nodes. We have also modeled the nodes by recurrent TSK-FIS (RFIS), where the sensor measurement of a node is approximated as function of real measurements of the neighboring nodes and the previously approximated value of the node itself. Temporary errors in sensor measurements and/or communication are overcome by redundancy of data gathering. A node can develop a faulty sensor because the sensor chip is attached to the WSN mote. The sensor chip is exposed to the environment, thus wear and tear can arise, possibly resulting in an inaccurate measurement. A node with a faulty sensor is not completely discarded because it is useful for relaying the information amongst the other nodes. Each node has its own fuzzy model that is trained with input of neighboring sensors' measurements and an output of its actual measurement. A sensor is declared faulty if the difference between the outcome of the fuzzy model and the actual sensor measurement is greater than the prescribed amount depending on the physical quantity being measured. Simulations are performed using the fuzzy logic toolbox of MATLAB®. We also give a comparison of obtained results to those from a feed-forward artificial neural network, recurrent

neural network and the median [Ding et al., 2005] of measured values of the neighboring nodes.

3.1 Introduction

Wireless sensor networks are emerging as computing platforms for monitoring various environments including remote geographical regions, office buildings and industrial plants [Akyildiz et al., 2002d]. They consist of the following: a set of nodes that can communicate with each other; sensors that measure a desired physical quantity; and the system base station for data collection, processing, and connection to the wide area network. Modern wireless sensor nodes have microprocessors for local data processing, networking, and control purposes . WSNs have enabled numerous advanced monitoring and control applications in environmental, biomedical, and numerous other applications.

One of the motivations for WSN modeling stems from the need for intelligent fault detection in complex distributed sensory systems. Because sensor networks often operate in potentially hostile and harsh environments, most of the applications are mission critical. The sensors are often used to compute control actions [Di et al., 2000, Katsura et al., 2003, Lysheyski, 2002], where sensors faults can cause catastrophic events. For instance, the National Aeronautics and Space Administration was forced to abort the launch of the space shuttle *Discovery* due to a failure in one of the sensors in the sensor network of the shuttle's external tank (the failure was discovered through human inspection) [Moustapha and Selmic, 2008].

Sensors and actuators boarded on a WSN node are more prone to faults as compared to traditional integrated semiconductor chips. Feedback about the functionality status of nodes is mandatory for multisensor systems so that they could eventually recover and heal from possible faults. Components such as sensors and actuators have significantly higher fault rates than the traditional integrated semiconductor circuits-based systems. Multisensor systems need feedback information about the

health status of their nodes in order to recover and heal from eventual faults. This would enhance the reliability on the system. Due to malfunctions or noise the sensor reading are more or less uncertain in the sense that no sensor will render an accurate reading at all times. Because low-cost sensor nodes are often deployed in an uncontrolled or even harsh environment, they are vulnerable to have faults. It is thus desirable to detect, locate the faulty sensor nodes, and exclude them from the network during normal operation unless they can be used as communication node. Consequently we need to design a WSN that is capable of fault detection [Moustapha and Selmic, 2008, Zhirabok and Preobragenskaya, 1993, Pouliezos and Stavrakankis, 1994]. Efficiency in converting data to features while consistently accommodating the uncertainty inherent in the measurements form a key issue for diagnosing and dealing with sensor faults [Zhirabok and Preobragenskaya, 1993, Pouliezos and Stavrakankis, 1994].

The ancient method for fault tolerance is to equip a node with multiple sensors but doing so would not only increase the cost of a node and hence that of the network but would also lead in more complexity and power consumption. So recent works are centered around analytical redundancy [Leushen et al., 2002, S.C.Lee, 1994] in which the sensor measurements are processed analytically, and the mathematical models are compared with the physical measurements. Therefore, instead of using additional hardware we use analytical fault detection, and model each node of a WSN through Takagi-Sugeno-Kang (TSK) fuzzy inference system (FIS).

3.2 Related works

Fault detection and fault tolerance in wireless sensor networks have been investigated in many research works. In [Jaikaeo et al., 2001] diagnosis for sensor networks has been carried out with additional attention to the congestion avoidance at the central node. In [Koushanfar et al., 2003] Koushanfar et al. have proposed an online detection technique for faulty sensors, where nonparametric statistical methods are

used to identify the sensors that have the highest probability to be faulty. In [Chessa and Santi, 2001] the problem of fault identification in ad-hoc networks is addressed. The diagnostic model lies upon the comparison-based one-to-many communication paradigm. In [Ruiz et al., 2004] Ruiz et al. have developed a management architecture for detection of faults in event-driven WSNs. In [Ding et al., 2005] the identification of faulty sensors in reach of events is discussed. The proposed generic algorithms are localized and thus scalable for large networks, however those are limited due to uneven distribution of nodes . In [Moustapha and Selmic, 2008] a node is identified as faulty depending upon the comparison of the output from a modified recurrent neural network to real measurement. In [Krishnamachari and Iyengar, 2004] a solution to the fault-feature disambiguation problem in sensor networks is proposed in the form of Bayesian fault-recognition algorithms exploiting the notion that measurement errors due to faulty equipment are likely to be uncorrelated, while environmental conditions are spatially correlated. In [Luo et al., 2006] the fault correction problem for distributed event detection in a WSN is studied. This distributed fault-tolerant detection scheme achieves optimal results when the neighborhood size is chosen based on the given detection error bound such that better balance between detection accuracy and energy usage is obtained.

In [Chen et al., 2006] the authors have presented a localized fault detection algorithm to identify the faulty sensors. It uses local comparisons with a modified majority voting, where each sensor node makes a decision based on comparisons between its own sensing data and neighbors' data, while considering the confidence level of its neighbors. The scheme, however, is a little complex in the sense that information exchange between neighboring nodes has to occur twice to reach a local decision based on a threshold. In addition, it does not allow transient faults in sensor reading and internode communication, which could occur for most normal sensor nodes [Lee and Choi, 2008a]. Transient faults in sensing and communication have been investigated in [Lee and Choi, 2008b], where a simple distributed algorithm has been proposed to tolerate transient faults in the fault detection process. Some

other fault management schemes can be found in the survey written by Yu et al. [Yu et al., 2007].

The rest of the chapter is organized as follows: Section 3.3 presents the system model and the assumptions made. In section 3.4 we discuss how we are treating the problem of fault detection. Sections 3.5 and 3.6 respectively represent the fuzzy inference modeling and the neural network modeling for the sensor fault detection. In section 3.7 we discuss the implementation of the proposed approach. In section 3.8 we present and discuss the simulation results, and finally in section 3.9 we conclude this chapter.

3.3 WSN modeling

The WSN under consideration accommodates n number of localized stationary homogeneous nodes with unique identity number and same transmission range, which communicate via a packet radio network. The proposed algorithm assumes: all nodes are fault free during deployment and during the training of the fuzzy inference system. For each node an FIS is created. The communication algorithm ensures that: each sensor knows the identity of its neighbor, MAC protocol solves contention problem over logical link, the link level protocol provides one hop broadcast.

3.3.1 Communication model

The communication graph of a WSN is represented as a graph $G(V, E)$, where V represents the set of sensor nodes in the network and E represents the set of edges connecting sensor nodes. The Cartesian coordinates of the node A_i are represented by $(A_{i,1}, A_{i,2})$. Two nodes A_i and A_j are said to have an edge in the graph if the distance

$$d(i, j) = \sqrt{\left(A_{i,1} - A_{j,1}\right)^2 + \left(A_{i,2} - A_{j,2}\right)^2}$$

between them is less than r (transmission range). That is,

$$d(i,j) \leq r \Leftrightarrow \left(A_i, A_j\right) \in E.$$

For convenience we assume that G is undirected, which means that if $\left(A_i, A_j\right) \in E$ then $\left(A_j, A_i\right) \in E$. The communication graph can be a test graph in our fault detection if two nodes with an edge connecting them are compared. If some of the edges are not involved in the fault detection or ignored based on the previous test results, a test graph in our fault detection can be a subgraph of the communication graph. For simplicity, we assume that communication graph and test graph are the same. For the graph $G(V,E)$ and $A_i \in V$, the set of the neighbors of A_i, $N(A_i)$ is defined to be

$$N(A_i) := \left\{A_j \in V : (A_i, A_j) \in E\right\}$$

For two connected nodes $\left(A_i, A_j\right) \in E$ we define a set

$$D_{i,j} := N(A_j) - (N(A_i) \cup \{A_i\})$$

3.3.2 Fault model

The value measured by node A_i at k^{th} instant of time, t_k, is denoted by x_i^k. If the time instant is not explicitly required the sensor measurement shall simply be denoted by x_i. Nodes with permanent faulty sensors are to be identified but are not excluded from the network because they are useful in relaying data packets amongst the nodes. Nodes with transient errors in sensor reading are termed as fault-free.

For a homogeneous physical quantity the difference, between the measured value at a fault-free sensor with the measured values of its fault-free neighbors, is bounded. So, if A_i and A_j are neighbors then in case of possessing fault-free sensors the following condition is satisfied:

$$|x_i - x_j| \leq \delta$$

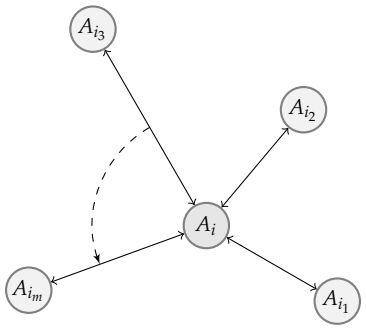


Figure 3.1: Neighbors of node A_i

where δ may vary depending on the application. If temperature is the physical quantity being measured, for example, then a sensor node and its neighbors are expected to have similar temperatures. Hence δ is exptected to be a small number. If the local binary decision at each node, instead of the sensed data, is transmitted to its neighbors, δ is set to 0.

3.4 Fault detection

Two nodes A_i and A_j are compared only if $(A_i, A_j) \in E$. Thus at time instant t_k if two such nodes have fully functional sensors then:

$$\left| x_i^k - x_j^k \right| \leq \delta_{i,j}^k \tag{3.1}$$

Suppose A_i has m neighbors i.e., $|N(A_i)| = m$. As shown in figure3.1, let these neighbors be denoted by

$$N(A_i) = \{A_{i_1}, A_{i_2}, ..., A_{i_m}\}.$$

So for this particular node we have

$$\left| x_i^k - x_{i_j}^k \right| \leq \delta_{i,i_j}^k, \text{ for } 1 \leq j \leq m.$$

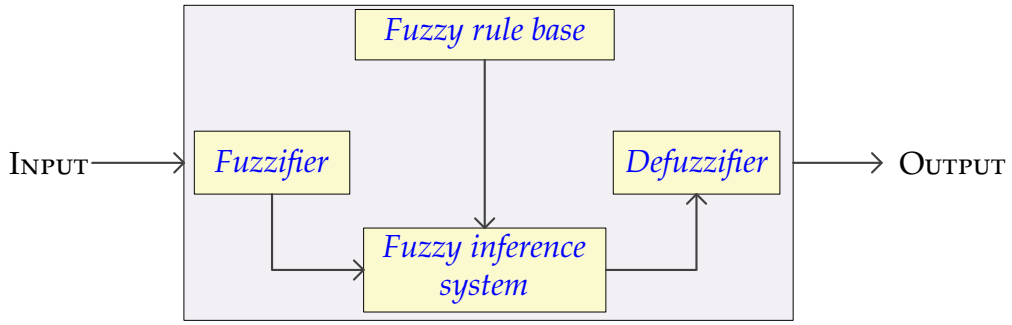


Figure 3.2: An example of a fuzzy logic system

Equivalently we can write it as

$$x_i^k = x_{i_j}^k + \epsilon_{i,i_j}^k, \text{ for } 1 \leq j \leq m$$

where ϵ_{i,i_j}^k is the difference between the i^{th} sensor measurement and that of its j^{th} neighbor at the instant t_k. Whence we get

$$mx_i^k = \sum_{j=1}^{m} \left(x_{i_j}^k + \epsilon_{i,i_j}^k \right)$$

or

$$x_i^k = \frac{1}{m} \sum_{j=1}^{m} \left(x_{i_j}^k + \epsilon_{i,i_j}^k \right) \tag{3.2}$$

Equation (3.2) represents a relation between the real sensor measurement of the node A_i and the sensor measurements of all of its neighbors. Which means the sensor measurement of A_i can be approximated by an m-variable function f of neighboring sensor measurements. That is

$$x_i^k \approx f\left(x_{i_1}^k, x_{i_2}^k, ..., x_{i_m}^k \right)$$

Hence for this node we create a TSK FIS which is trained with inputs as the sensor measurements of $N(A_i)$ nodes and output as the real sensor measurement of the node A_i.

3.5 TSK fuzzy treatment

The fuzzy logic system (FLS) [Takagi and Sugeno, 1985, Sugeno and Kang, 1986] is an inference system which mimics the human thinking and its basic configuration consists of a fuzzifier, some fuzzy IF-THEN rules, a fuzzy inference engine and a defuzzifier, as shown in figure 3.2. A fuzzy rule is written as the following statement:

$$R^l \ : \ \text{IF } x_1 \text{ is } B_1^l \text{ and } x_2 \text{ is } B_2^l \text{ and } \cdots x_n \text{ is } B_n^l \text{ THEN } y \text{ is } y^l$$

where $R^l (l = 1, 2, ..., M)$ denotes the l^{th} implication, $x_j (j = 1, 2, ..., n)$ are input variables of the FLS, y^l is a singleton, B_j^l is the fuzzy membership function which can represent the uncertainty in the reasoning. When we use the product inference, center-average and singleton fuzzifier, the output of the fuzzy system for an input $\mathbf{x} = (x_1, x_2, ..., x_n)^T$ can be expressed as

$$y = \frac{\sum_{i=1}^{n} \alpha_i y^i}{\sum_{i=1}^{n} \alpha_i}$$

where α_i implies the overall truth value of the premise of the i^{th} implication, and is computed as

$$\alpha_i = \prod_{l=1}^{M} A_l^i(x_i)$$

We are also using a recurrent FIS in which the added input is the previously approximated value, as shown in figure 3.3.

3.6 Neural network treatment

The sensor measurement of a node is also approximated as a function of neighboring nodes by using MLP neural network as shown in figure 3.4. The input vector to input layer, $\boldsymbol{\alpha} = [\alpha_1 , \alpha_2 , \alpha_3 , \alpha_4 , \alpha_5]$ has components $[x_1^k , x_2^k , x_7^k , x_{11}^k , x_{12}^k]$, which are the sensed values at the neighbors of node A_6, as would be discussed latter in this chapter. The 5×6 matrix $\mathbf{U} = [\beta_{i,j}]$ represents the input-to-hidden layer weights. The activation function of each of the hidden layer neuron is denoted by σ_i for

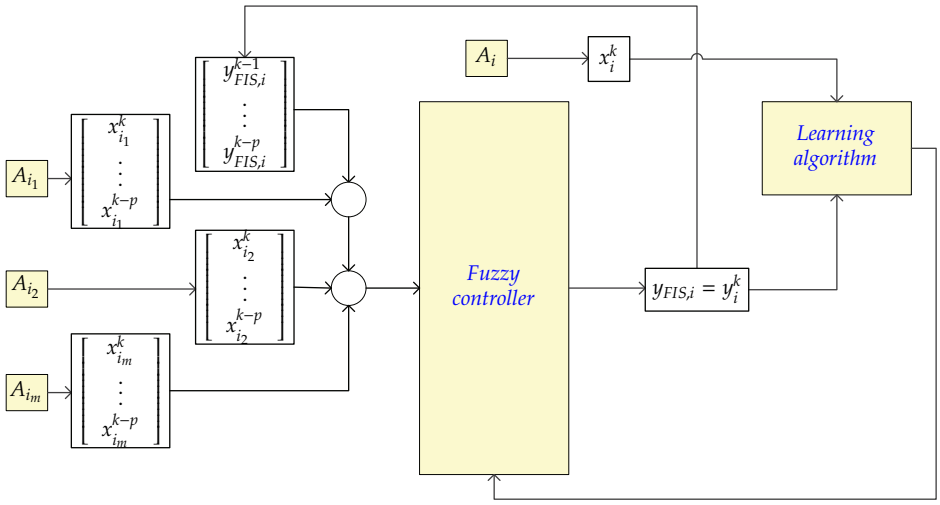


Figure 3.3: Recurrent fuzzy controller for fault detection in a sensor

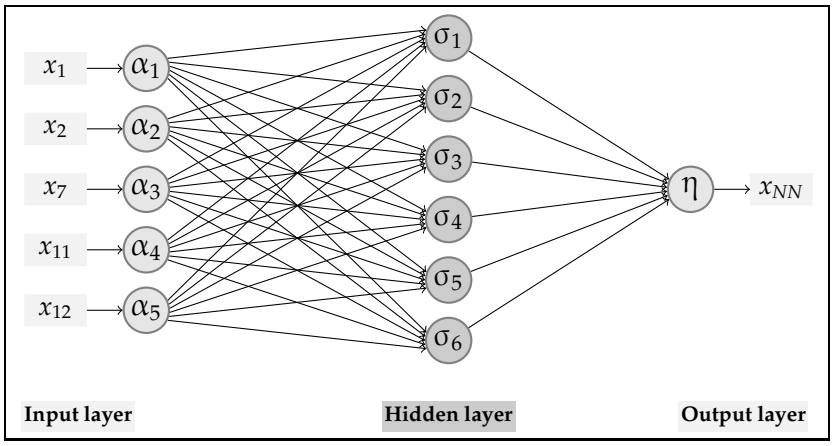


Figure 3.4: Three layered neural network for node A_6 with five input variables and one output variable

$i = 1, 2, ..., 6$. Each of the σ_i is the logsigmoid function. These activation functions are represented by a vector $\boldsymbol{\sigma}^T = (\sigma_1\ ,\ \sigma_2\ ,\ \cdots\ ,\sigma_6)$. The vector $\mathbf{w}^T = (w_1, w_2, ..., w_6)$ represents the hidden-to-output layer weights. The activation function of the output layer is denoted by η and is the linear identity function. The single scalar output x_{NN} is the sensor measurement approximated by the neural network:

$$x_{NN} \approx \eta \left\{ \sum_{j=1}^{6} w_j \sigma_j \left(\sum_{i=1}^{5} \alpha_i \beta_{i,j} \right) \right\}$$

In matrix form which is written as:

$$x_{NN}(\boldsymbol{\alpha}) = \eta\left\{\mathbf{w}^T\sigma\left(\mathbf{U}^T\boldsymbol{\alpha}\right)\right\}$$

3.7 Implementation of the proposed fault detection approach

Suppose we want to measure the health status of the node A_i. So for this node we train an initial FKS FIS with input $\mathbf{x}_{FIS} = (x_{i_1}, x_{i_2}, ..., x_{i_m})^T$ and output $y_{FIS} = x_i$. The type of membership function is gaussian. The number of membership functions for each component of input vector depends upon the range of temperature being measured. For a larger range the number of membership functions is greater than the number of membership function for a shorter range. If number of membership functions is kept constant and the temperature range is increased then the size of fuzzy set will increase but the number of rules will remain the same. And hence inference will loose fine tuning. Here we are using five membership functions for each neighboring sensed value x_{i_j} for $j = 1, 2, ..., m$. So the fuzzy rules for node A_i are given by

$$R^l \ : \ \text{IF } x_{i_1}^k \text{ is } F_1^l \text{ and} x_{i_2}^k \text{ is } F_2^l \ \cdots \text{ and } x_{i_m}^k \text{ is } F_m^l \text{ THEN } y_{FIS}^l \ = \ x_i^k$$

for $l = 1, 2, ..., M$, where M is the total number of rules (in present case $M = 5^m$). The plot of membership functions of the variable x_{i_3} (where $i = 6$) obtained through fuzzy tool box of MATLAB® is shown in figure 3.18. After training FIS we apply it through simulation on a WSN scenario. So at an instant t_k the output of a fuzzy controller is $y_{FIS} = y_i^k$ as shown in figure 3.6. Then we compare this output value with the actual sensed measurement at node A_i and if

$$\left|y_i^k - x_i^k\right| \geq TOLERENCE \tag{3.3}$$

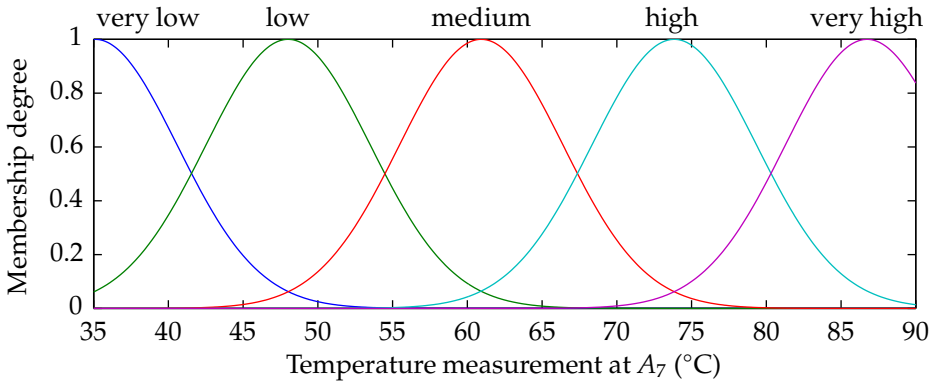


Figure 3.5: Plot of membership functions for the variable x_{i_3}, where $i = 6$

holds true then the sensor of the node A_i is identified as faulty. Note the difference between (3.1) and (3.3). The condition (3.1) states that the difference between the measurements of two neighboring fully functional senors is bounded. While in (3.3) the absolute value difference between the actual measurement and the FIS approximation is compared with the tolerance permitted by the WSN. Now we talk about the members of $N(A_i)$ that can participate in finding health status of the node A_i.

A node $A_{i_j} \in N(A_i)$ shall participate in the fault identification of the node A_i if the condition (3.5) is satisfied, in which the node A_{i_j} shall tally its own status with that of the elements of D_{i,i_j}. So there is a possibility that one or more elements of $N(A_i)$ shall not be involved in A_i's fault identification. If $|N(A_i)| = m$ and l of these nodes are not participating then there are

$$\begin{pmatrix} m \\ l \end{pmatrix} \equiv \frac{m!}{l!(m-l)!}$$

combinations for the participating neighboring nodes with l varying from 1 to $m-1$. The total number of possible combinations is

$$\sum_{l=1}^{m-1} \begin{pmatrix} m \\ l \end{pmatrix} = 2^m - 2$$

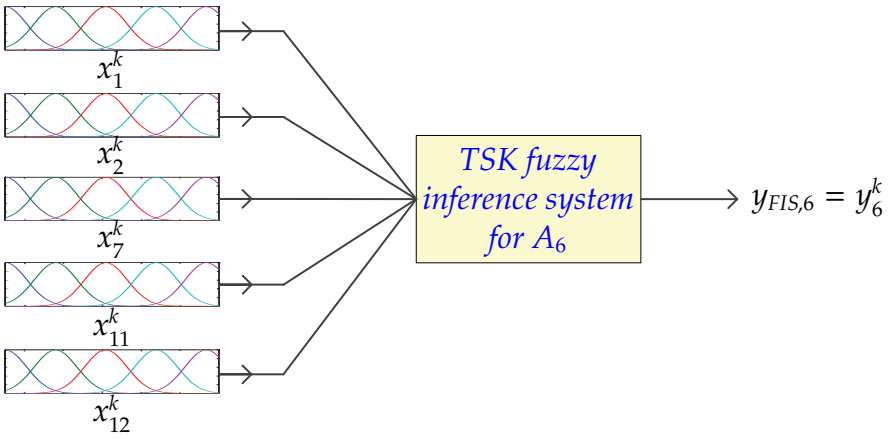


Figure 3.6: Approximated value for A_6 by the fuzzy controller

where each combination corresponds to an FIS. Now we describe the condition (3.5). For the node A_{i_j}, we have

$$D_{i,i_j} = N(A_{i_j}) - (N(A_i) \cup \{A_i\})$$

Let $|D_{i,i_j}| = \zeta$ and these nodes be denoted by $u_1, u_2, ..., u_\zeta$. The sensor measurement of the node A_{i_j} is compared with the sensor measurements of the nodes $u_1, u_2, ..., u_\zeta$. To tackle the transient faults we shall have this comparison for multiple times$(t_1, t_2, ..., t_k)$. Let us denote $x_{i_j}^q$ by $T(A_{i_j}, t_q)$ where $q = 1, 2, ..., k$. So on the same pattern we shall have sensor measurements of these ζ nodes as $T(u_\gamma, t_q)$ for $\gamma = 1, 2, ..., \zeta$ and $q = 1, 2, ..., k$. Let us define a function

$$g(x_{i_j}^q, T(u_\gamma, t_q)) = \begin{cases} 1 & \text{if } A_{i_j} \text{ and } u_\gamma \text{ satisfy conditon (3.1)} \\ 0 & \text{otherwise} \end{cases} \tag{3.4}$$

So the results from function (3.4) are stored in an $\zeta \times k$ matrix $H = [h_{\gamma,q}]$ where

$$h_{\gamma,q} = g(x_{i_j}^q, T(u_\gamma, t_q))$$

A label C_{i_j,u_γ} is attached to A_{i_j} with

$$C_{i_j,u_\gamma} = \begin{cases} 1 & \text{if } \sum_{q=1}^{k} h_{\gamma,q} \geq (k-\mu) \\ 0 & \text{otherwise} \end{cases}$$

where μ depends upon the number of instances the data is gathered. Now, if

$$\sum_{\gamma=1}^{\zeta} C_{i_j,u_\gamma} \geq \lambda \tag{3.5}$$

Where λ is selected as a threshold for this condition, on whose fulfilment the node A_{i_j} participates in the fuzzy fault identification of the node A_i.

3.8 Simulation results

We have simulated a sensor network with 15 sensor nodes as shown in figure 3.7 and one sensor per node. Each node has at least three one hop neighbors. The quantity being measured is the temperature. The temperature of all nodes is gathered for a period of 80 hours equally divided into 100 instances. For the simulation purpose the temperature T at a point (x, y) and at time t is given by

$$T(x, y, t) = \sqrt{x^2 + y^2} + L\cos(\phi + 2\pi f t) + \sin(\frac{5}{2}t) + 60$$

where $L = 25$, $f = 0.025$ and $\phi = \pi$. The reason for choosing this particular heuristic function is that with this expression the temperature varies from 34.15°C to 88.88°C. The temperature changes smoothly and there are no sudden jumps or discontinuities. The differences in the data output are small enough to guarantee and justify the theoretical approach described in section 3.4. Each sensor is modeled using an FIS as described in previous sections. An FIS has inputs consisting of the sensor measurements of the neighboring nodes. Each input variable to FIS has five membership functions of type gaussian. An FIS is generated by using the grid

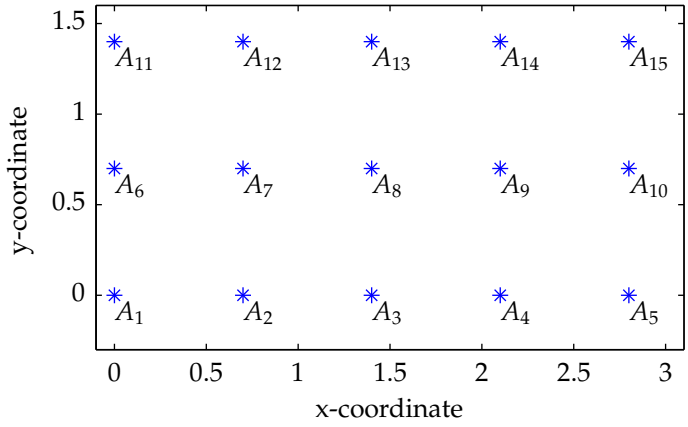


Figure 3.7: A WSN scenario with 15 nodes

partition and is trained by using hybrid method. We have used MATLAB® as a simulation software. Here we consider and discuss the status of the node A_6 with

$$N(A_6) = \{A_1\,,\, A_2\,,\, A_7\,,\, A_{11}\,,\, A_{12}\} \tag{3.6}$$

The initial FIS is trained with input of sensor measurements of all the five neighboring (in order) nodes. The k^{th} sample input vector to FIS has the components:

$$x_1^k \quad x_2^k \quad x_7^k \quad x_{11}^k \quad x_{12}^k$$

where k is varied from 1 to 100, that is, the FIS is trained with the temperature values of neighborhood nodes for the entire period of 80 hours. Similarly the neural network is also trained from these data spanned over eighty hours.

Figure 3.8 shows a comparison of the actual measurement of node A_6 with the FIS model, NN model, and the median of the real sensor measurements from $N(A_6)$. The advantage of FIS model over the median method is that it always takes into account the individual measurement from each of the neighbor nodes. An individual erroneous sensor measurement extends its error to the combined input when we take the median. The estimation for the sensed measurement of A_6 by FIS outperforms the approximated values both from NN and median models. Since the temperature

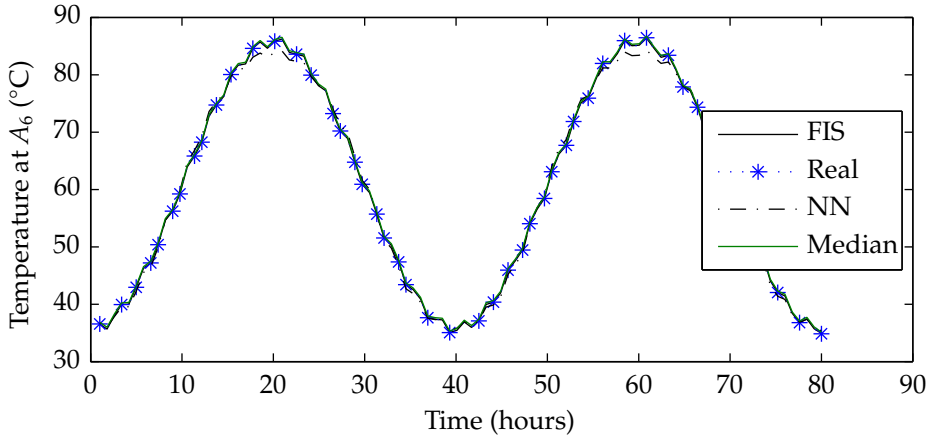


Figure 3.8: Real sensor measurement of node A_6 and its models using TSK FIS, NN, and median

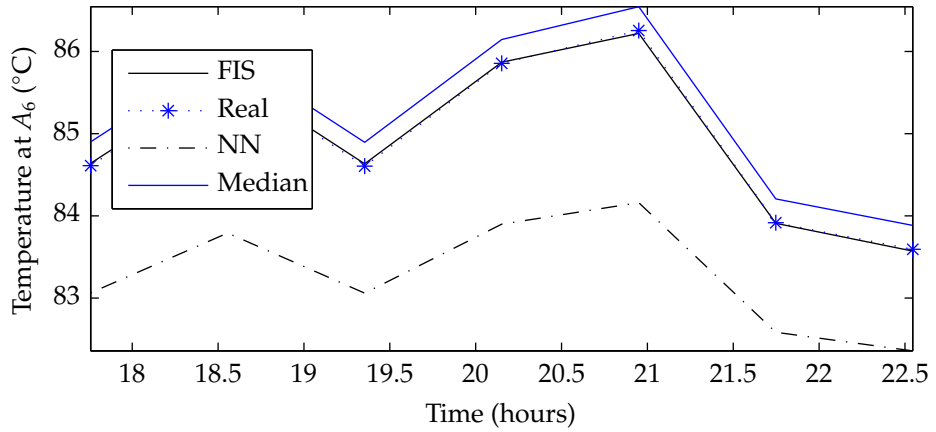


Figure 3.9: A magnified portion from the figure 3.8

data in figure 3.8 is condensed and the approximated values from different models are not clearly distinguishable so a portion has been zoomed in and is shown in figure 3.9.

The absolute value of the difference between the approximations by different models and the real measurement is shown in figure 3.10. Since the FIS model closely approximates the real value and the difference between the two is very small therefore, we are using a logarithmic scale on the temperature measurement axis.

In order to detect a fault in the sensor of node A_6 we introduced an increasing

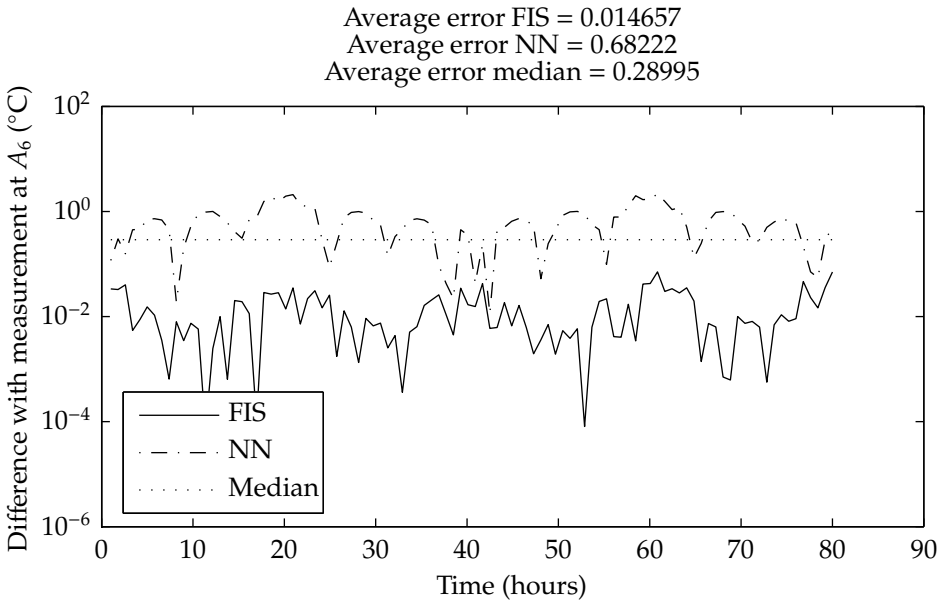


Figure 3.10: Absolute difference of FLS, NN, and median model value with real sensor measurement

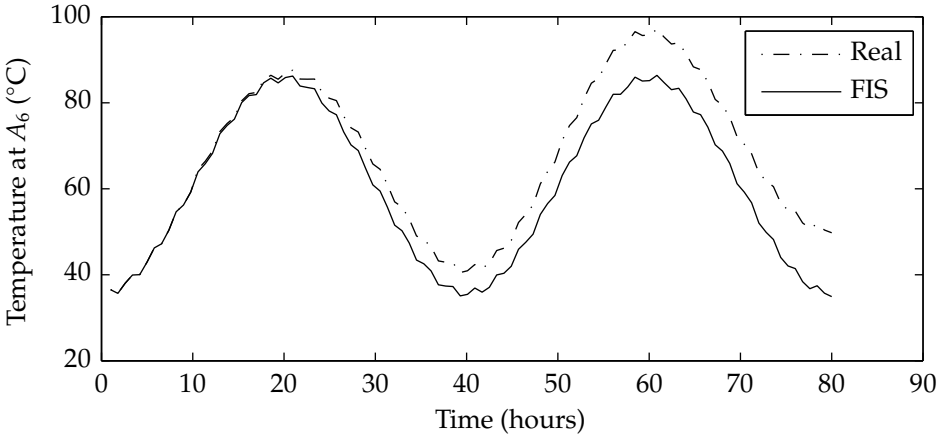


Figure 3.11: Sensor measurement and FIS values for the entire period of 80 hours

deviation, as a function of time, in its temperature measurement:

$$\epsilon(t) = \left[\sin\frac{t}{4} + \frac{t-10}{5}\right] H(t-10)$$

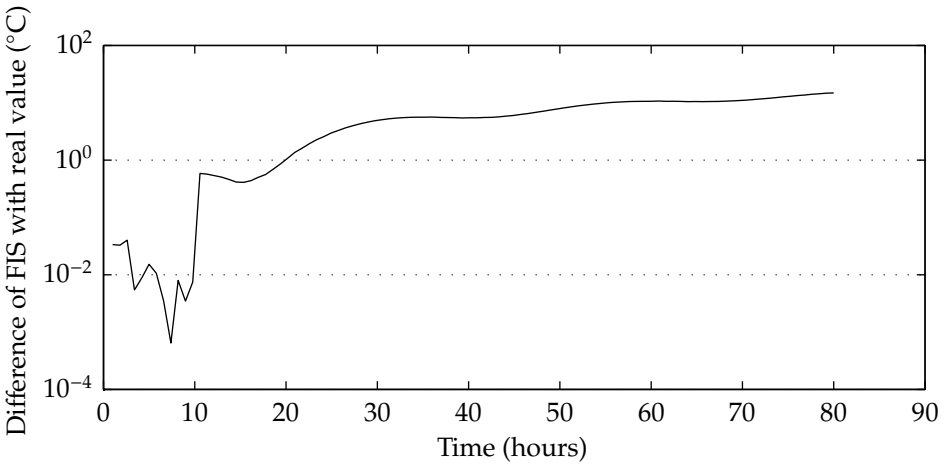


Figure 3.12: Difference between the actual and FIS estimated values

where t is in hours and $H : \mathbb{R} \to \{0, 1\}$ is the unit step function:

$$H(x) = \begin{cases} 1 & , \quad x \geq 0 \\ 0 & , \quad x < 0 \end{cases}$$

Then we plotted the gradually deviating real measurement of node A_6 and the approximated measurements by the FIS for the entire period of 80 hours. The results are shown in figure 3.11. The temperature measurement for the first 10 hours behaves normally but after that there arise a gradually increasing difference between the real value at A_6 and the value estimated by the FIS. The absolute value of the difference between the two measurements is shown in figure 3.12. Once again the difference between the two measurements for the first 18 hours is so small that it is better to scale the temperature measurement axis logarithmically. From $t = 20$ onwards the real measurement starts differing from the FIS estimated value by more than 1°C. Also from the figure 3.12 one can decide when to identify the node as faulty depending upon the tolerance allowed by the application.

3.8.1 Transient fault tolerance

Now we discuss the fault tolerance of the proposed approach. By fault tolerance we mean an intermittent perturbation in the sensor measurement of a node that shall be

ignored by our scheme. The results for the estimated value for node A_6 are discussed here to elucidate the fault tolerance aspect in the presented method. On its turn every member of $N(A_6)$, as mentioned in (3.6), is made to show an irregular behavior. The transient error and hence the disturbed sensor reading, $\tilde{x}_j^k$, of neighboring nodes at an instance t_k is as follows:

$$\tilde{x}_j^k = x_j^k + EB \sin\left(\frac{t_k}{4}\right) \tag{3.7}$$

for $j = 1, 2, 7, 11, 12$, where EB is the bound on the introduced perturbation. The number of neighboring nodes with transient fault is varied from 1 to m, for the present example $m = 5$. Then these perturbed values are used as an input to FIS and obtained output value is compared with the real observed value of the sensor measurement, x_6^k in this case. The results for different values for EB are shown in tables 3.1 and 3.2, and in figures 3.13 and 3.14. From table 3.1 we can infer that even if 50% of the neighbors are manifesting a disturbed behavior than usual, the difference between the real sensed measurement and the FIS estimated value is acceptably small.

Table 3.1: Transient fault with absolute value less than 1

Nodes with transient faults	**Min. diff. (°C)**	**Max. diff. (°C)**	**Average diff. (°C)**
0	6.8143×10^{-5}	0.070821	0.014657
1	0.00054166	0.323710	0.125320
2	0.00472610	0.578390	0.249150
3	0.00956560	0.774180	0.374370
4	0.00914440	0.911180	0.501320
5	0.00872320	1.013200	0.630710

As shown in figure 3.15 the measurement of node A_1 is perturbed and the rest of the $N(A_6)$ sensor measurements show the usual behavior. Still the difference between the sensor measurement of node A_6 and its estimated value is very less as is shown in figure 3.16 which is a magnified portion of figure 3.15.

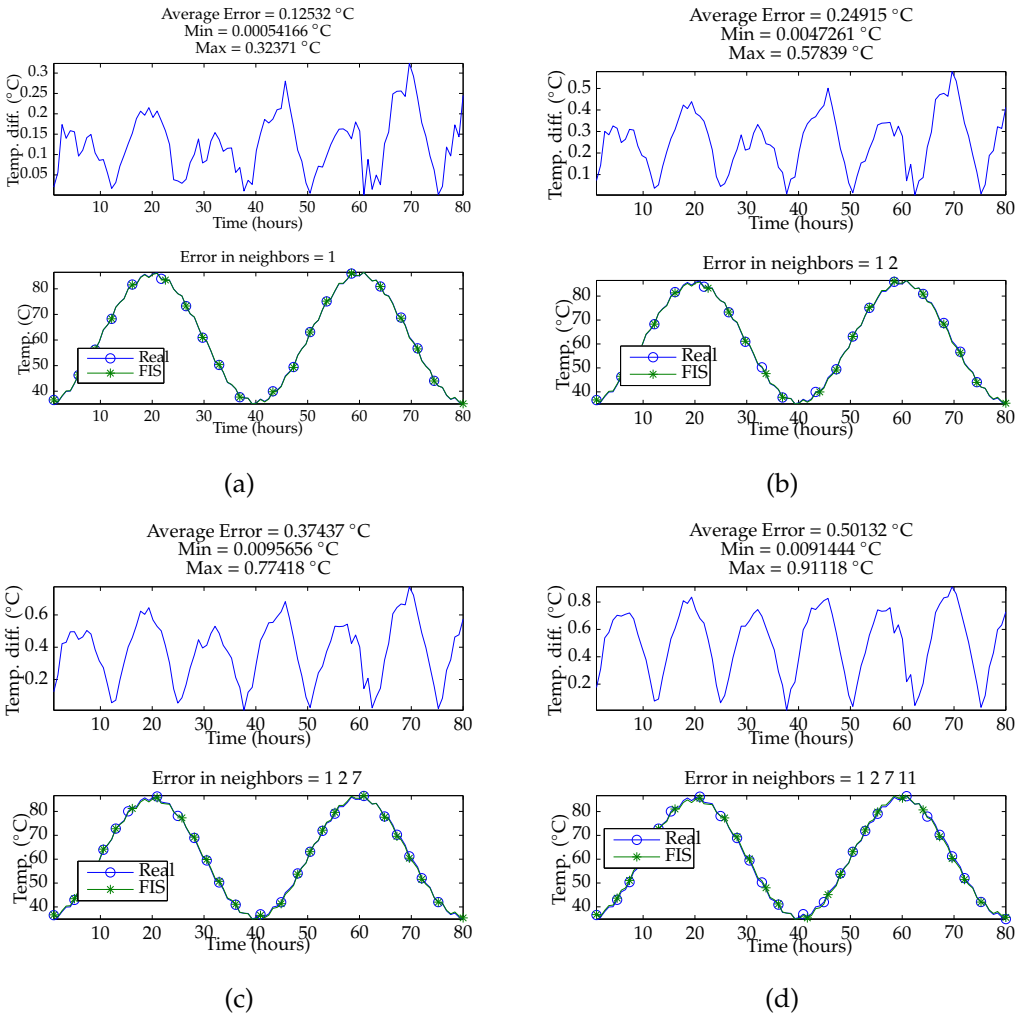


Figure 3.13: Transient faults in neighboring nodes with absolute value less than 1

3.8.2 Recurrent FIS Treatment

We have also conducted our approach with recurrent fuzzy inference system (RFIS). Also we have done a comparison with recurrent neural network (RNN) and median of the neighboring node sensor measurements. The RFIS is demonstrated in figure 3.3. And the RNN is demonstrated in figure 3.17. The RFIS is trained with input $\mathbf{x}^k = (x^k_{i_1}, x^k_{i_2}, ..., x^k_{i_m}, y^{k-1}_i)^T$ and output $y^k_i = x^k_i$. we use three membership functions for each neighboring sensed value x_{i_j} for $j = 1, 2, ..., m$. So the fuzzy rules for node A_i

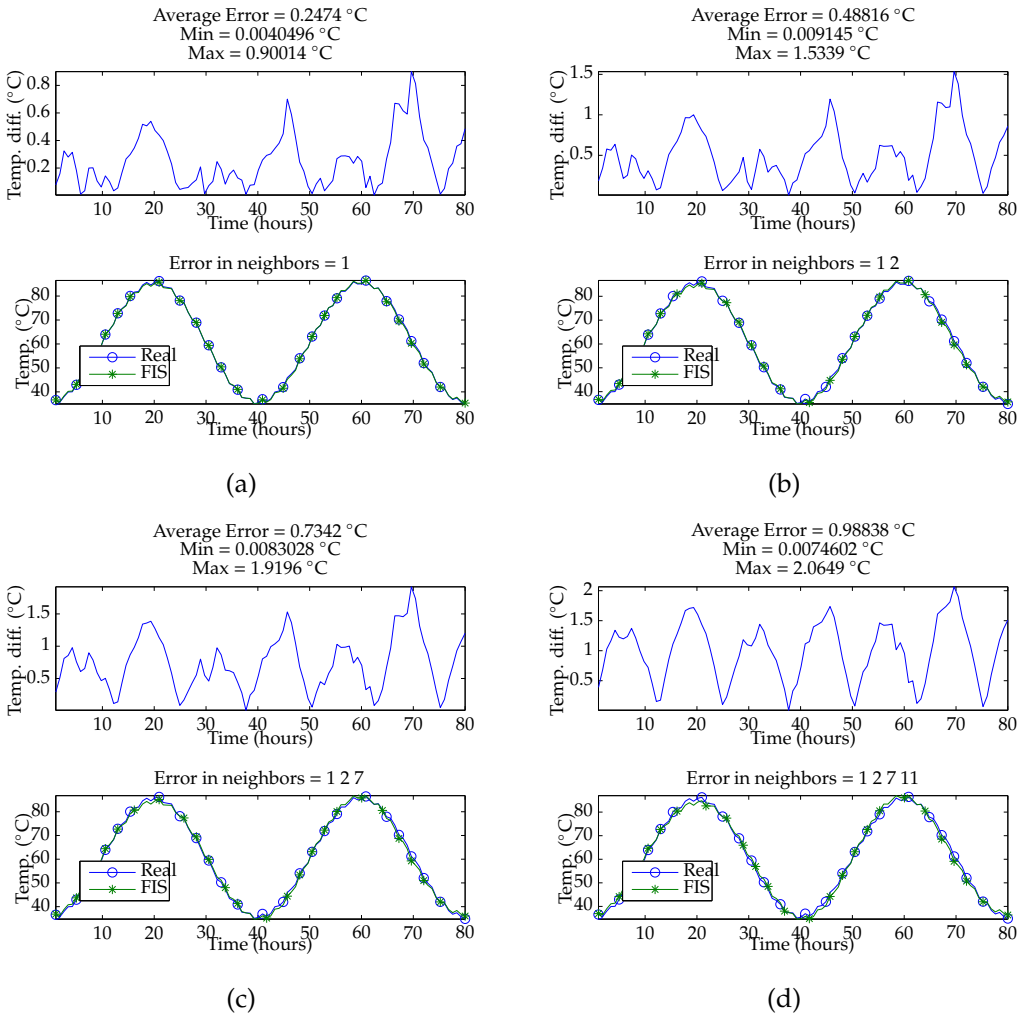


Figure 3.14: Transient faults in neighboring nodes with absolute value less than 2

are given by

$$R^l \ : \quad \text{IF } x^k_{i_1} \text{ is } F^l_1 \ \cdots \text{ and } x^k_{i_m} \text{ is } F^l_m \text{ and } y^{k-1}_{FIS} \text{ is } F^l_{m+1}$$
$$\text{THEN } y^l_{FIS} = x^k_i$$

for $l = 1, 2, ..., M$, where M is the total number of rules (in present case $M = 3^{m+1}$). The plot of membership functions of the variable x_{i_3} (where $i = 6$) obtained through fuzzy tool box of MATLAB® is shown in figure 3.18. Since, for the sake of example we have

Table 3.2: Transient fault with absolute value less than 2

Nodes with transient faults	Min. diff. (°C)	Max. diff. (°C)	Average diff. (°C)
1	0.0040496	0.90014	0.24740
2	0.0091450	1.53390	0.48816
3	0.0083028	1.91960	0.73420
4	0.0074602	2.06490	0.98838
5	0.0066175	2.01350	1.25680

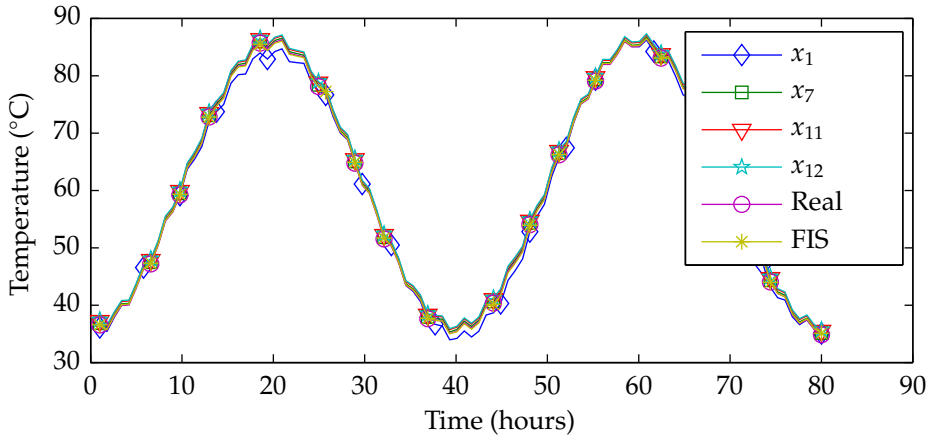


Figure 3.15: Sensor measurements of neighbors and estimated value of the node itself

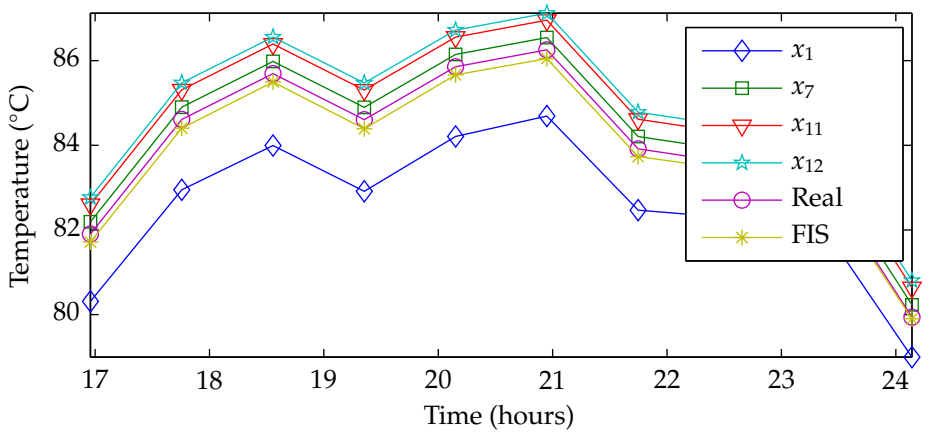


Figure 3.16: A zoomed in portion of figure 3.15

chosen node A_6, therefore, the k^{th} input to RFIS sample vector has the components:

$$x_1^k \quad x_2^k \quad x_7^k \quad x_{11}^k \quad x_{12}^k \quad y_6^{k-1}$$

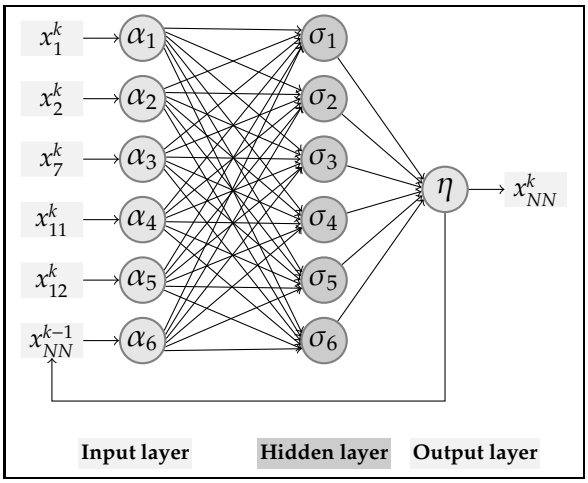


Figure 3.17: Three layered RNN for node A_6 with five input variables and one output variable

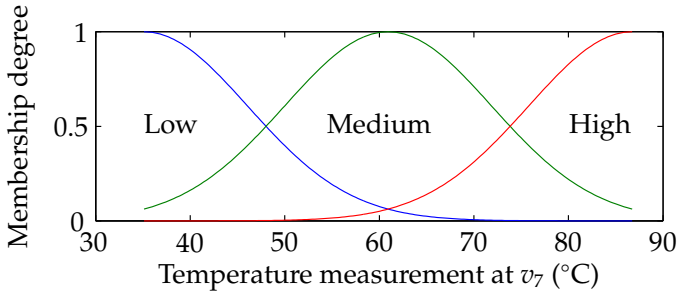


Figure 3.18: Plot of membership functions for the variable x_{i_3}, where $i = 6$

Figure 3.19 shows a comparison of the real measurement of node A_6 with the RFIS model, RNN model, and the median of the real sensor measurements from $N(A_6)$. Since the temperature data in figure 3.19 is condensed and the approximated values from different models are not clearly distinguishable so a portion has been zoomed-in and is shown in figure 3.20. The absolute value of the difference between approximations by different models and the real measurement is shown in figure 3.21. Figure 3.22 shows the RFIS approximated values and the real sensor measurement of node A_6 with increasing deviation introduced. For the recurrent technique, the results for different values for EB, in (3.7), are shown in tables 3.3 and 3.4. From the tables we can see that RFIS is performing better than FIS. Once again, like earlier, from the figure 3.22 we can decide when to declare the node as faulty depending on

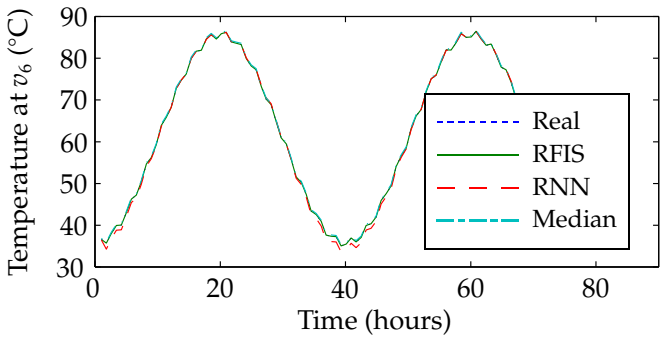


Figure 3.19: Real sensor measurement of node v_6 and its models using recurrent TSK FIS, RNN, and median method

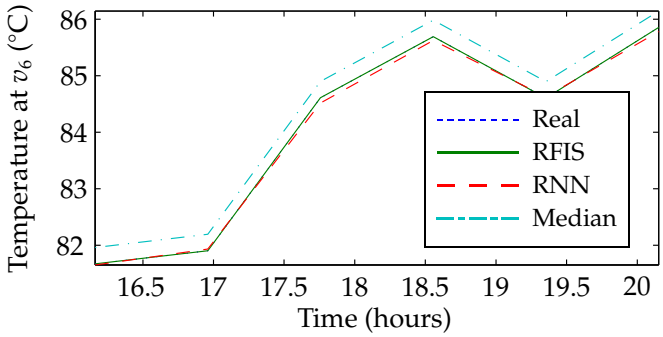


Figure 3.20: A portion magnified from the figure 3.19

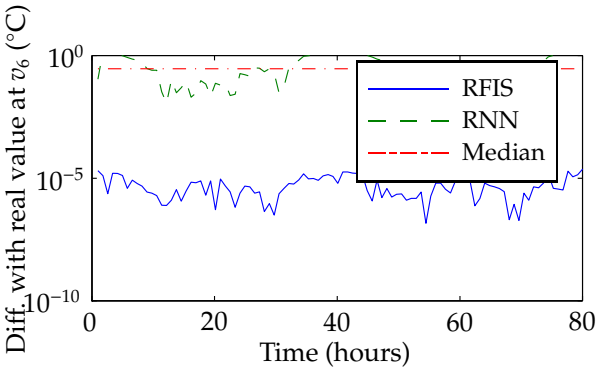


Figure 3.21: Absolute difference of RFIS, RNN, and average model value with real sensor measurement

the desired difference between the real and RFIS approximated value.

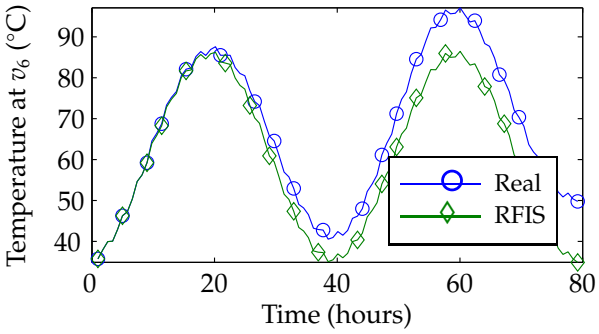


Figure 3.22: Sensor measurement and the RFIS values for the entire period of 80 hours

Table 3.3: Transient fault with absolute value less than 1

Nodes with transient faults	Min. diff. (°C)	Max. diff. (°C)	Average diff. (°C)
0	3.996×10^{-7}	3.473×10^{-4}	6.522×10^{-5}
1	0.00049349	0.25241218	0.13505085
2	0.00090606	0.47156798	0.26456387
3	0.00130369	0.66836709	0.39195162
4	0.00168033	0.83957573	0.51623019
5	0.00204862	0.99968624	0.63941108

Table 3.4: Transient fault with absolute value less than 2

Nodes with transient faults	Min. diff. (°C)	Max. diff. (°C)	Average diff. (°C)
1	0.00094208	0.56351787	0.26891727
2	0.00176691	1.02979571	0.52720860
3	0.00256196	1.42193454	0.78187899
4	0.00331539	1.73497709	1.03106361
5	0.00405246	1.99931670	1.27880158

3.9 Conclusion

This chapter describes a distributed, sensor fault identification scheme for a wireless sensor network. Each node of the sensor network is modeled by a fuzzy inference system which approximates the measurement of that node as a function of the real measurements of the neighboring nodes. The difference between the actual value detected at a node and the estimated value given by its corresponding FIS model

is used to decide whether or not to declare the node as faulty. Since the scheme is distributed and that the computations are performed at the base station the suggested method is less energy consuming. Simulation results show the efficiency of proposed scheme and that the fuzzy inference model outperforms the results given by artificial neural network and that of median of the one-hop neighbor measurements. Once we know the id of a faulty node, it is indispensable to find its geographic location. In the next chapter we discuss our proposed localization scheme.

Chapter 4

Localization

In this chapter we shall show that three randomly chosen nodes as anchors in a wireless sensor network are sufficient to localize all of the nodes, where the nodes are in point set triangulation. The claim is supported in the form of a theorem. We start the process of localization from the information of connectivity between the nodes and the distance matrix. From the distance matrix we find the topology of the network through a heuristic approach. Finally we introduce three nodes as anchors and from the real exact positions of anchors we localize all the sensors. We conducted our proposed localization algorithm in different WSN scenarios by performing simulations in MATLAB®. The obtained results show a substantial improvement in the position estimation of sensors.

4.1 Introduction

A Wireless Sensor Network (WSN) consists of spatially distributed autonomous sensors to cooperatively monitor physical or environmental conditions, such as temperature, sound, vibration, pressure, or motion. The development of wireless sensor networks was motivated by military applications such as battlefield surveillance. It is now used in areas including industrial process monitoring and control, machine health monitoring, environment and habitat monitoring, health-care applications,

home automation, and traffic control etc. In addition to one or more sensing devices, each node in a sensor network is typically equipped with a radio transceiver or other wireless communications device, a small microcontroller, and an energy source, usually a battery [Townsend and Arms, 2004]. In [Khan et al., 2010a] we have dealt with the nodes battery voltages in regard to localization.

In location based services such as battlefield surveillance or wild fire control, the information sensed by a node is of less importance unless the position of the source node is known. The process of finding the geographical position of such a node is called *localization*. From localization point of view there are two types of nodes in a WSN: anchor nodes and the tracked nodes. Anchor nodes are the nodes whose geographical position is known, say for example they are equipped with global positioning system (GPS) or their position is pre-configured before their deployment. A tracked node is a node whose position is not known at the time of its deployment. A sensor node can detect a change in the physical quantity for which it is meant to be and can transfer this information to other nodes. If a sensor node does not know its position then the information sent does not contain the location of the geographical region in which that change took place. Hence localization is indispensable for location based services [Akyildiz et al., 2002c]. In literature, sensor nodes are simply referred to as the sensors and the anchor nodes as the anchors. In this text the term *node* shall represent either of the two nodes, a tracked node or an anchor node.

The motivation for the present work stems from answering the question: Whether or not it is possible to localize (in 2D) all the nodes in a WSN with exactly three anchors. The answer is yes as well as no. For 'yes' certain conditions are required. From graph theory point of view it is 'yes' if we know the pairwise distances between all nodes. In case, where the pairwise distances between the critical nodes are not known the answer is 'no'. Moreover, the answer is also 'yes' when the nodes, along with the edges information, form a point set triangulation; a result proven in one of the subsequent sections. It means, first of all there is no orphan node

(whose node-degree = 1) in the network and that the network is not divisible in to two subnetworks that are connected either by single link or have just one node in common. The assumptions made in our work are more or less the same as implicitly and, or explicitly stated in other works, cited in the next section. However, our approach does require that the neighbors of any node form a cycle. The value of the node degree of an arbitrary node is at least three, unless it is at the boundary of the network. In that case it is connected to at least two other nodes. With these assumptions it is always possible to find the point set triangulation of the entire set of nodes in the WSN. And hence planar coordinates of any three nodes will render the coordinates of all the other nodes.

4.2 Related Works

A lot of work has been done for the localization in a WSN when each of the sensor is in direct contact with some anchor [Wong et al., 2005], [Cassioli, 2009]. Furthermore the localization process has also been studied when the number of anchors is significantly smaller than that of sensors [Dakkak et al., 2011]. It involves the WSNs in which most of the senors are interconnected with the other sensors and only a few are in direct contact with the anchors and usually the anchors are deployed at the boundary of the WSN [Bischoff et al., 2006, Hamam et al., 2009]. In [Regalia and Wang, 2010], [Alfakih et al., 1999] the distance-based node localization is discussed when there is a need for reconstructing the distance matrix.

In [Patil et al., 2005], authors have proposed to localize all the sensors using three anchors. Although, in our approach too, a WSN requires exactly three anchors but unlike [Patil et al., 2005] there is no condition of the three anchors being in transmission range of each other and having at least one sensor in the common transmission range of all the three anchors. In our approach any three nodes can serve as anchors irrespective of their position in the WSN. Our work does not require constraint for any sensor to be in the common transmission range of the anchors.

The process of localization using multidimensional scaling is proposed in [Shang et al., 2004], however we use a greedy algorithmic approach for this task. We shall start the process of localization from the information given by distance matrix. A distance matrix is a symmetric sparse matrix whose $(i, j)^{th}$ component is the distance between the i^{th} and j^{th} node. The distance matrix is sparse because the inter-connectivity of the nodes is not too dense to the limit that an arbitrary node is connected to most of the nodes. Rather a node is connected to only a few of the nodes and it is out side the transmission range of most of the nodes. Moreover the distance between two nodes is measured through "RSSI – distance" model where the distance is obtained from the signal strength information. If a node is outside the transmission range of another node then the corresponding element in the distance matrix is zero.

Here is how we are going to tackle the aforementioned localization problem. First of all we model the process of finding the topology of the sensor nodes with the help of the distance matrix. Then we accumulate the temporary Cartesian coordinates of all nodes in a two-column matrix. Then we choose three nodes as anchors and using their real positions we shall find the estimated positions of all the nodes.

The rest of the chapter is as follows. In section 4.3 we present the formulation of the problem. In section 4.4 we present our proposed solution to find the estimated position of all nodes. We demonstrate the proposed approach with the help of a working example in section 4.5. In section 4.6 we present the simulations performed in MATLAB® and their results and finally in section 4.7 we conclude this chapter.

4.3 Formulation Of The Problem

4.3.1 System Model

All of the n nodes in the WSN are homogeneous and have same circular transmission range. Two nodes are termed connected if they are in the transmission range of each other. Each node knows its own ID and the IDs of its one hop neighbors. This WSN

is represented as a graph $G(V, E)$, where

$$V = \{A_1, A_2, \ldots, A_n\}$$

is the set of nodes and E is the set of ordered pairs of nodes (A_i, A_j) that are connected. Furthermore, this graph is undirected, i.e., if $(A_i, A_j) \in E$ then also $(A_j, A_j) \in E$. So there is a symmetric binary relation $(\sim)$ defined over V as

$$A_i \sim A_j \iff (A_i, A_j) \in E$$

If a pair $(A_i, A_j) \notin E$ it is denoted by $A_i \nsim A_j$.

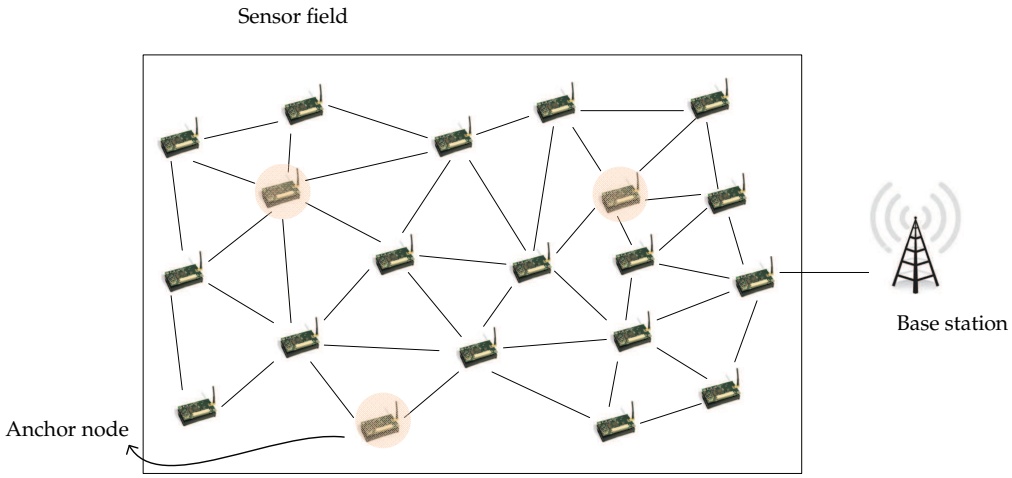


Figure 4.1: A wireless sensor network scenario

4.3.2 Communication Model

Consider a WSN scenario as shown in figure 4.1, where the circled highlighted motes are the anchor nodes. When the wireless sensor network is deployed, all of the nodes communicate with their directly connected neighbors. The nodes that are connected to each other transmit and receive ping signals. The information in the signals travels by relaying and reaches the sink. The sink is in direct connection with the base station. Whenever there is a locomotion in any node it transmits signals to

be reached at the base station. Now at the base station we have the information of the connected nodes and the strengths in their respective signals. Thus by using the RSSI and distance model we find the distances between the connected nodes and hence the distance matrix. The RSS and distance model is given by [Smith, 1998]:

$$P_r = kd^{-\alpha}$$

where P_r is the strength of the received signal, k is a constant which takes into account carrier frequency and transmitted power, d is the distance between the connected nodes and α is the attenuation exponent. From this expression the distance between the nodes is obtained as:

$$d = \left(\frac{P_r}{k}\right)^{-\frac{1}{\alpha}}$$

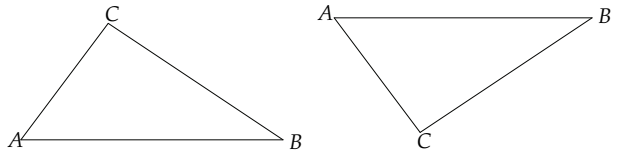
The information regarding the distance and received signal strength from the base station leads us to form the distance matrix for the WSN.

In the absence of anchors or any reference point for that matter it is impossible to localize the network. However, with the help of the distance matrix alone it is possible to get the initial layout of the WSN. In other words it is possible to find the topological structure of the connectivity of the nodes. But if the network contains some nodes that are only connected to one node then the topology obtained from the distance matrix is not unique. So a node must be connected to at least three other nodes and that the neighbors of any node form a cycle. This leads to the unique topology of the network. Then the final part of the problem is to find the estimated positions of all the nodes, which is accomplished by introducing three anchors. Now we prove that three anchors are sufficient to completely localize the nodes in WSN.

Theorem 1. *Given the point set triangulation of points V of $G(V, E)$. If we know the position coordinates of any three points of V, we know the position coordinates of all the points of V.*

Proof. Given three lengths a , b , c or ($\overline{BC}$, $\overline{CA}$, $\overline{AB}$) then up to position and orientation in the Cartesian plane, two triangles correspond (figure4.2). Both these triangles are mirror images of each other. Thus up to position, orientation, and symmetry

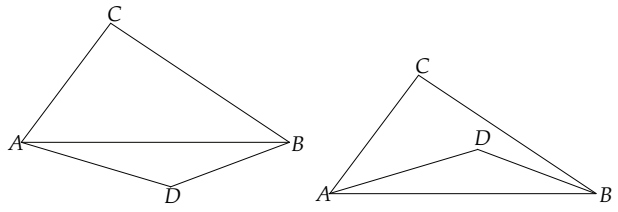
the three lengths correspond to a unique triangle say the one in figure4.2(a). Given



(a) (b)

Figure 4.2: Triangle up to position and orientation

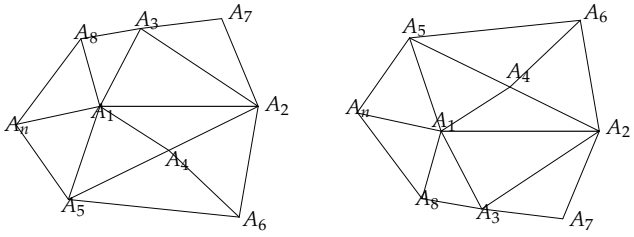
two more lengths $\overline{AD}$ and $\overline{BD}$ then figure4.2(a) shall render figure4.3(a) and not figure4.3(b) because it is not a triangulation of the four points. Continuing in the



(a) Permitted (b) Not Permitted

Figure 4.3: Addition of a fourth point

same manner with n points, $V = \{A_1, A_2, \ldots, A_n\}$ and corresponding lengths from the set $E = \{(A_i, A_j) : A_i \sim A_j\}$, there is a unique configuration (figure4.4(a)) up to position, orientation, and symmetry. Given the Cartesian coordinates of only one



(a) (b)

Figure 4.4: Configuration of n points in triangulation

node say A_1, then the graph figure4.4(a) shall have infinite number or orientations and flips with the point (A_{1x}, A_{1y}) as pivot. With Cartesian coordinates of two nodes say A_1 and A_2 the graph can have just one flip (figure4.4(b)) around the line segment (A_1, A_2). But with the Cartesian coordinates of just one more node the graph has

a unique position, orientation, and symmetry. And hence we know the Cartesian coordinates of all nodes. □

4.4 Proposed Approach

Let us denote the nodes by $A_1, A_2, A_3, \cdots, A_n$. The distance matrix is denoted by $d_{n\times n} = [d(i,j)]$. If $(A_i, A_j) \notin E$, then $d(i,j) = 0$, otherwise, $d(i,j)$ is obtained from the RSSI-distance model. In the distance matrix $d(i,i) = 0$ for $i = 1, 2, 3, \cdots, n$. Two nodes A_i and A_j are said to be connected if $d(i,j) \neq 0$ and this fact is denoted by $A_i \sim A_j$. Two nodes A_k and A_l are not connected if and only if $d(i,j) = 0$ and this fact shall be denoted by $A_k \nsim A_l$. At any instant the position of the i^{th} node shall be denoted by $X_i = [x_i \quad y_i]$. The positions of all the nodes are presented by a two column matrix

$$X = [\mathbf{x} \quad \mathbf{y}]_{n\times 2}$$

where $\mathbf{x}$ and $\mathbf{y}$ are the column vectors:

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix} \quad \text{and} \quad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_n \end{bmatrix}$$

Now we proceed in two steps. In step one we find the topology of the nodes irrespective of its symmetry, orientation, and position in the Cartesian plane. We accomplish this step without knowing the positions of any of the nodes and just with the help of the distance matrix. Then by treating this topology as a single entity in step two we find the exact symmetry, orientation and position in $\mathbb{R}^2$ with the help of real positions of any three nodes.

4.4.1 Finding the Topology

Under the assumption that each node is connected to every other node then there are ${}^nC_2 = \frac{n!}{2!(n-2)!} = \frac{n(n-1)}{2}$ connection pairs:

$$\begin{array}{lll} A_1 \sim A_2 & A_1 \sim A_3 & \cdots \quad A_1 \sim A_n \\ & A_2 \sim A_3 & \cdots \quad A_2 \sim A_n \\ & & \vdots \quad \vdots \quad \vdots \\ & & A_{n-1} \sim A_n \end{array}$$

But we have no such assumption. Anyhow the total number of possible connections, $m := {}^nC_2$, between the nodes shall be denoted by pairs of indices:

$$\begin{array}{lll} \mu_1 := (1,2) & \mu_2 := (1,3) & \cdots \quad \mu_{n-1} := (1,n) \\ & \mu_n := (2,3) & \cdots \quad \mu_{2n-3} := (2,n) \\ & & \vdots \quad \vdots \quad \vdots \\ & & \mu_m := (n-1,n) \end{array}$$

That is the pair $\mu_1 := (1\,,\,2)$ denotes the possibility that $A_1 \sim A_2$, the pair $\mu_2 := (1\,,\,3)$ denotes that of $A_1 \sim A_3$ and so on. Note that for any pair $\mu_j = (k,l)$ for $j = 1,2,3,\ldots,m$, the first element of the pair shall be denoted by $\mu_j(1) = k$ and the second element by $\mu_j(2) = l$. Let us orderly enumerate the actual connections (and rename them by ρ_i) between the nodes as given by the distance matrix and exclude all other possibilities. For that purpose the pair μ_j for $j = 1,2,3,\ldots,m$, shall be tagged false if $A_{\mu_j(1)} \nsim A_{\mu_j(2)}$, otherwise the pair is tagged true. We perform the following routine:

1: $i = 0$
2: **for** $j = 1$ **to** m **do**
3: **if** $A_{\mu_j(1)} \sim A_{\mu_j(2)}$ **then**
4: $i++$
5: $\rho_i \leftarrow \mu_j$
6: **end if**
7: **end for**

8: $s \leftarrow i$

So the actually connected pairs are: ρ_i for $1 \leq i \leq s$, where $s \leq m$. That is $A_{\rho_i(1)} \sim A_{\rho_i(2)}$ for $i = 1, 2, \ldots, s$.

Before we proceed further we declare a function that shall be used latter. Suppose at any intermediate instance the coordinates X_i and X_j of the pair $A_i \sim A_j$ have been found out and the coordinates $X_k = [x_k \quad y_k]$ of a third node A_k are yet to be found with the condition that $A_k \sim A_i$ and $A_k \sim A_j$. So X_k is one of the points of intersection of the circles:

$$\begin{aligned} (x - x_i)^2 + (y - y_i)^2 &= (d(i,k))^2 \\ (x - x_j)^2 + (y - y_j)^2 &= (d(j,k))^2 \end{aligned} \tag{4.1}$$

The two points of the solution of system (4.1) are P_1 and P_2 and hence the two possible positions $X_k^{(1)}$, $X_k^{(2)}$ for A_k are $X_k^{(1)} = P_1$ and $X_k^{(2)} = P_2$. Thus we define a function with three arguments: the coordinates X_i and X_j of the pair $A_i \sim A_j$ and the index k of the node such that $A_k \sim A_i$ and $A_k \sim A_j$; and the return of the function are the two possible values for X_k. This function is denoted by f and is explained in algorithm 4.1. Note that the distance matrix $d_{n\times n} = [d(i,j)]$ is a global variable which is available to all the routines and algorithms.

Algorithm 4.1 Possible position for the node connected to two linked nodes

Input: X_i, X_j, k
Output: $X_k^{(1)}, X_k^{(2)}$

1: $\begin{bmatrix} P_1 \\ P_2 \end{bmatrix} \leftarrow \begin{cases} (x - x_i)^2 + (y - y_i)^2 &= (d(i,k))^2 \\ (x - x_j)^2 + (y - y_j)^2 &= (d(j,k))^2 \end{cases}$

2: **return** $\begin{bmatrix} X_k^{(1)} \\ X_k^{(2)} \end{bmatrix} \leftarrow \begin{bmatrix} P_1 \\ P_2 \end{bmatrix}$

For an intermediate instance where the coordinates $X_{\rho_i(1)}$ and $X_{\rho_i(2)}$ of the nodes pair represented by ρ_i i.e., $A_{\rho_i(1)} \sim A_{\rho_i(2)}$ have been found out, we find the the coordinates of the nodes connected to both of the nodes $A_{\rho_i(1)}$ and $A_{\rho_i(2)}$ as follows. Let us denote by N_{ρ_i} the ordered set of indices of the nodes that are common to both the nodes of the pair ρ_i. If $\left|N_{\rho_i}\right| = q$ then the common nodes of the pair represented

by ρ_i are:

$$A_{N_{\rho_i}(1)}\ ,\ A_{N_{\rho_i}(2)}\ ,\ \ldots\ ,\ A_{N_{\rho_i}(q)}$$

and if their possible positions are denoted by

$$Z_1^{(1\text{ or }2)}, Z_2^{(1\text{ or }2)}, \ldots, Z_q^{(1\text{ or }2)}$$

we can obtain them by using algorithm 4.1 and performing the following routine:

1: **for** $j = 1$ **to** q **do**
2: $\begin{bmatrix} Z_j^{(1)} \\ Z_j^{(2)} \end{bmatrix} \leftarrow f(X_{\rho_1(1)}\ ,\ X_{\rho_1(2)}\ ,\ N_{\rho_i}(j))$
3: **end for**

Since each of Z_j has two possible positions $Z_j^{(1)}$ and $Z_j^{(2)}$. Therefore the total number of possible configurations for Z_1, Z_2, ..., Z_q are 2^q. The superscripts for each configuration form a $q-$ tuple, where any element of the tuple is either 1 or 2. Let us denote the tuples by σ_k for $k = 1, 2, 3, \ldots, 2^q$. Thus the possible configurations are:

$$Z_1^{\sigma_k(1)}\ ,\ Z_2^{\sigma_k(2)}\ ,\ \cdots, Z_q^{\sigma_k(q)} \text{ where } 1 \leq k \leq 2^q.$$

Out of these 2^q configurations only one or two correspond to the actual positions of nodes $A_{N_{\rho_i}(1)}$, $A_{N_{\rho_i}(2)}$, ..., $A_{N_{\rho_i}(q)}$. The case where two configurations correspond shall be dealt with latter in the current section. In order to filter that particular configuration we shall measure the distances amongst the temporary node positions of each configuration and compare it against the respective distances as given by the distance matrix. For a configuration σ_k we shall store the pair-wise distances amongst the q points:

$$\begin{array}{ccccc} \left\|Z_1^{\sigma_k(1)} - Z_2^{\sigma_k(2)}\right\|, & \left\|Z_1^{\sigma_k(1)} - Z_3^{\sigma_k(3)}\right\|, & \cdots, & \left\|Z_1^{\sigma_k(1)} - Z_q^{\sigma_k(q)}\right\| \\ & \left\|Z_2^{\sigma_k(2)} - Z_3^{\sigma_k(3)}\right\|, & \cdots, & \left\|Z_2^{\sigma_k(2)} - Z_q^{\sigma_k(q)}\right\| \\ & & & \vdots \quad \vdots \quad \vdots \\ & & & \left\|Z_{q-1}^{\sigma_k(q-1)} - Z_q^{\sigma_k(q)}\right\| \end{array} \tag{4.2}$$

in a row matrix $D^{(k)}$. Also we shall store the respective pair-wise distances from the distance matrix

$$\begin{array}{ccccc} d(N_{\rho_i}(1), N_{\rho_i}(2)), & d(N_{\rho_i}(1), N_{\rho_i}(3)), & \cdots, & d(N_{\rho_i}(1), N_{\rho_i}(q)) \\ & d(N_{\rho_i}(2), N_{\rho_i}(3)), & \cdots, & d(N_{\rho_i}(2), N_{\rho_i}(q)) \\ & & & \vdots \quad \vdots \quad \vdots \\ & & & d(N_{\rho_i}(q-1), N_{\rho_i}(q)) \end{array} \tag{4.3}$$

in a row matrix dd. Note that each of (4.2) and (4.3) has $r := {}^qC_2 = \frac{q(q-1)}{2}$ elements. That is, there are r pair-wise distances both in $D^{(k)}$ and dd. We denote such a pair by ω_t where $1 \leq t \leq r$. We compare the elements of $D^{(k)}$ with the elements of dd for $k = 1, 2, \ldots, 2^q$. That is we perform the following iteration:

1: **for** $k = 1$ **to** 2^q **do**
2: $\quad \epsilon_k = \left\|D^{(k)} - dd\right\|$
3: **end for**

The index k_0 of $\min\{\epsilon_k : k = 1, 2, \ldots, 2^q\}$ corresponds to the configuration that is in accordance with the given distance matrix. Thus out of all 2^q possible configurations, the desired one is:

$$Z_1^{\sigma_{k_0}(1)}, Z_2^{\sigma_{k_0}(2)}, \cdots, Z_q^{\sigma_{k_0}(q)}$$

Hence the coordinates of the common elements of the pair ρ_i are given by:

1: **for** $j = 1$ **to** q **do**
2: $\quad X_{N_{\rho_i}(j)} \leftarrow Z_j^{\sigma_{k_0}(j)}$
3: **end for**

Thus we declare a function g as shown in algorithm 4.2. with input as the index pair denoted by ρ_i (such that $X_{\rho_i(1)}$ and $X_{\rho_i(2)}$ have been determined); and the output as the coordinates of the nodes connected to both of $A_{\rho_i(1)}$ and $A_{\rho_i(2)}$.

Algorithm 4.2 Tentative coordinates of the nodes connected to both the nodes of a pair ρ_i with known $X_{\rho_i(1)}$ and $X_{\rho_i(2)}$

Input: ρ_i
Output: Coordinates of all nodes connected to $A_{\rho_i(1)}$ and $A_{\rho_i(2)}$

$N_{\rho_i} = \left\{ l : A_l \sim A_{\rho_i(1)} \text{ and } A_l \sim A_{\rho_i(2)} \right\}$
$q \leftarrow \left| N_{\rho_i} \right|$
for $j = 1$ **to** q **do**
  $\begin{bmatrix} Z_j^{(1)} \\ Z_j^{(2)} \end{bmatrix} \leftarrow f(X_{\rho_i(1)}, X_{\rho_i(2)}, N_{\rho_i}(j))$
end for
$r \leftarrow {}^{q}C_2$
for $t = 1$ **to** r **do**
  $dd(t) = d(N_{\rho_i}(\omega_t(1)), N_{\rho_i}(\omega_t(2)))$
end for
for $k = 1$ **to** 2^q **do**
  for $t = 1$ **to** r **do**
    $D^{(k)}(t) \leftarrow \left\| Z_{w_t(1)}^{\sigma_k(w_t(1))} - Z_{w_t(2)}^{\sigma_k(w_t(2))} \right\|$
  end for
  $\epsilon_k \leftarrow \left\| D^{(k)} - dd \right\|$
end for
return k_0 s.t. $\epsilon_{k_0} := \min\{\epsilon_k : k = 1, 2, \ldots, 2^q\}$
for $j = 1$ **to** q **do**
  $X_{N_{\rho_i}(j)} \leftarrow Z_j^{\sigma_{k_0}(j)}$
end for
return $\begin{bmatrix} X_{N_{\rho_i}(1)} \\ X_{N_{\rho_i}(2)} \\ \vdots \\ X_{N_{\rho_i}(q)} \end{bmatrix}$

Now we combine all the functions and routines constructed so far and find the two column matrix X giving the temporary coordinates of the nodes $A_1, A_2, \ldots, A_n$. Note that $\rho_1(1) = 1$, so without any harm and loss of generality we start the process of localization by putting $X_1 = [0 \quad 0]$ and $X_{\rho_1(2)} = [d(1, \rho_1(2)) \quad 0]$. That is we place the first element of the first valid pair at the origin and its second element on the

positive $x-$axis. We then mark the indices 1 and $\rho_1(2)$ as plotted and this fact is termed equivalent to ρ_1 as marked plotted. The process shall continue till the time all the indices, from 1 till n, are marked as plotted. Since $X_{\rho_1(1)}$ and $X_{\rho_1(2)}$ have been determined so we can apply function g as described by algorithm 4.2 on ρ_1. It is here that we obtain two configurations which correspond to the distance matrix. We can choose anyone of them because one is the mirror image of the other with the line of symmetry as the line segment joining $X_{\rho_1(1)}$ and $X_{\rho_1(2)}$. As we shall be flipping the topology, if required, so it does not matter which one is chosen at this step. Thus in algorithm 4.3 we summarize the process of finding *initial* coordinate matrix $X_{n\times 2}$ whose point plot is equivalent to the topology of the original WSN.

4.4.2 Symmetry, Orientation and Position of the Topology

After having executed algorithm 4.3 the plot from the coordinates of matrix X is a topological equivalent of the WSN topology. From this point onwards the topology obtained from X is treated as a single entity. Now we tackle the process of finding the symmetry, orientation and position of this topology in $\mathbb{R}^2$. We flip, rotate and translate the topology wherever required. For that purpose any arbitrarily chosen three nodes from $A_1, A_2, \ldots, A_n$ can serve as anchors. Let these nodes be A_α, A_β, and A_γ and their real positions be denoted by Y_α, Y_β, and Y_γ respectively. From X the respective estimated position of these three nodes are X_α, X_β, and X_γ. First of all we find whether or not the topology obtained from algorithm 4.3 requires a flip. Meaning that if we plot the real and estimated positions of these points whether or not they agree in circular direction. The topology obtained so far needs a flip if they have different circular direction as shown in figure 4.5. Consider the four vectors:

$$\mathbf{u}_1 = Y_\beta - Y_\alpha \qquad \mathbf{v}_1 = X_\beta - X_\alpha$$

$$\mathbf{u}_2 = Y_\gamma - Y_\alpha \qquad \mathbf{v}_2 = X_\gamma - X_\alpha$$

Algorithm 4.3 Initial estimated coordinates of all of the nodes

Input: Distance matrix $d = [d(i, j)]$ for $1 \leq i, j \leq n$
Output: Two column matrix X

1: $m \leftarrow {}^{n}C_2$ and $i \leftarrow 0$
2: **for** $j = 1$ **to** m **do**
3: **if** $A_{\mu_j(1)} \sim A_{\mu_j(2)}$ **then**
4: $i++$
5: $\rho_i \leftarrow \mu_j$
6: **end if**
7: **end for**
8: $s \leftarrow i$
9: $\begin{bmatrix} X_{\rho_1(1)} \\ X_{\rho_1(2)} \end{bmatrix} \leftarrow \begin{bmatrix} 0 & 0 \\ d\left(\rho_1(1)\,,\ \rho_1(2)\right) & 0 \end{bmatrix}$
10: Mark ρ_1 as plotted
11: **for** $i = 1$ **to** s **do**
12: **if** ρ_i is marked plotted **then**
13: $q \leftarrow \left|N_{\rho_i}\right|$
14: $\begin{bmatrix} X_{N_{\rho_i}(1)} \\ X_{N_{\rho_i}(2)} \\ \vdots \\ X_{N_{\rho_i}(q)} \end{bmatrix} \leftarrow g\left(\rho_i\right)$
15: **end if**
16: Mark N_{ρ_i} as plotted
17: **end for**
18: **return** $X \leftarrow \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_n \end{bmatrix}$

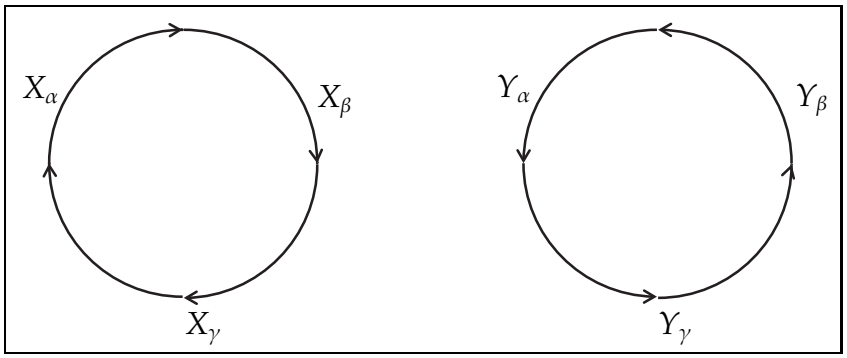


Figure 4.5: The topology obtained from algorithm 4.3 requires a flip if the circular orders of the estimated and real positions are different.

and take the cross product of the vectors as:

$$\mathbf{u} = \mathbf{u}_1 \times \mathbf{u}_2 \quad \text{and} \quad \mathbf{v} = \mathbf{v}_1 \times \mathbf{v}_2$$

If the condition,

$$\text{sgn}(u_z) = -\text{sgn}(v_z)$$

is satisfied, i.e., the third components of the vectors $\mathbf{u}$ and $\mathbf{v}$ have opposite signs, then the estimated topology obtained so far needs a flip. Implying that the obtained topology is the mirror image of the original topology. We accomplish the flip by changing the signs of all the elements in first column of X. After this update, we find the rotation needed in order for X to match the orientation of the original topology.

What is the amount of rotation and what is the pivot around which the topology shall be rotated, is described as follows: Translate the matrix X by amount $[-x_\alpha \quad -y_\alpha]$. That is the topology X is translated in such a way that α^{th} node is at the origin. By treating the combination of the three points Y_α, Y_β, and Y_γ as a single entity, translate this system of three points by amount $-Y_\alpha$. Such that the updated coordinates of the point Y_α are $[0 \quad 0]$. Now there are two cases depending upon the collinearity of the vectors $\mathbf{u}_1$ and $\mathbf{v}_1$. In the case they are collinear then the topology does not require a rotation. In the case where these are non-collinear as shown in figure 4.6, then there is an angle, say θ between them. We find the value of θ by using $\mathbf{u}_1 \cdot \mathbf{v}_1 = |\mathbf{u}_1||\mathbf{v}_1| \cos\theta$.

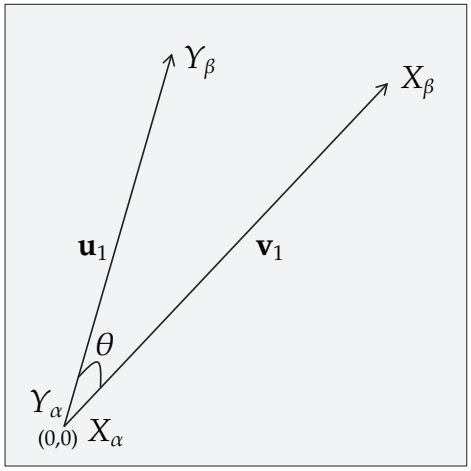


Figure 4.6: The case where the vectors $\mathbf{u}_1$ and $\mathbf{v}_1$ are non-collinear

Then we rotate the vector $\mathbf{v}_1$ by an angle of amount θ in the direction of $\mathbf{u}_1$ such that both the vectors become collinear. Note that the entire topology as a single entity is rotated with X_α as pivot. This rotation is accomplished (or X is updated) as follows:

1: **for** $i = 1$ **to** n **do**
2: $$\begin{bmatrix} x_i \\ y_i \end{bmatrix} \leftarrow \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix}$$
3: **end for**

The point X_α remains invariant under this rotation transformation. After rotation this topology is in the same orientation as the original topology. Then we translate the point X_α to the original position of the point Y_α. And thus resulting in the superposition of $X_\alpha, X_\beta, X_\gamma$ over $Y_\alpha, Y_\beta, Y_\gamma$. Hence the topology has a unique symmetry, orientation, and position in $\mathbb{R}^2$. Algorithm 4.4 depicts the process of finding symmetry, orientation and position of the topology.

Algorithm 4.4 Finding symmetry, orientation and position of the topology in $\mathbb{R}^2$ from the initial coordinate matrix X

Input: Estimated coordinate matrix X from algorithm 4.3 and three real positions Y_α, Y_β, and Y_γ of nodes A_α, A_β, and A_γ

Output: Updated two column matrix X giving the final estimated positions of all the nodes

1: Calculate the vectors:

$$\begin{array}{lllllll} \mathbf{u}_1 & \leftarrow & Y_\beta - Y_\alpha & & \mathbf{v}_1 & \leftarrow & X_\beta - X_\alpha \\ \mathbf{u}_2 & \leftarrow & Y_\gamma - Y_\alpha & & \mathbf{v}_2 & \leftarrow & X_\gamma - X_\alpha \\ \mathbf{u} & \leftarrow & \mathbf{u}_1 \times \mathbf{u}_2 & & \mathbf{v} & \leftarrow & \mathbf{v}_1 \times \mathbf{v}_2 \end{array}$$

2: **if** $\text{sgn}(u_z) = -\text{sgn}(v_z)$ **then**

3: $X \leftarrow \begin{bmatrix} -x_1 & y_1 \\ -x_2 & y_2 \\ \vdots & \vdots \\ -x_n & y_n \end{bmatrix}$

4: **end if**

5: $X \leftarrow \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ \vdots & \vdots \\ x_n & y_n \end{bmatrix} + \begin{bmatrix} -x_\alpha & -y_\alpha \\ -x_\alpha & -y_\alpha \\ \vdots & \vdots \\ -x_\alpha & -y_\alpha \end{bmatrix}$ %Translate X by $[-x_\alpha \ -y_\alpha]$

6: Translate the system of three points Y_α, Y_β, and Y_γ such that $Y_\alpha \leftarrow [0 \quad 0]$

7: Solve $\mathbf{u}_1 \cdot \mathbf{v}_1 = |\mathbf{u}_1||\mathbf{v}_1| \cos\theta$, for θ

8: $R \leftarrow \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$ %Rotation matrix

9: **for** $i = 1$ **to** n **do**

10: $X_i^T \leftarrow R X_i^T$

11: **end for**

12: Translate X s.t. X_α is equal to original Y_α

13: **return** $X \leftarrow \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_n \end{bmatrix}$

4.5 Working Example

Consider a 6×6 distance matrix:

$$d = \begin{bmatrix} 0 & 0.2834 & 0.9665 & 0.6934 & 0.2188 & 0.9967 \\ 0.2834 & 0 & 0.8804 & 0.4181 & 0.4757 & 0.9126 \\ 0.9665 & 0.8804 & 0 & 0.7864 & 0.9462 & 0.0322 \\ 0.6934 & 0.4181 & 0.7864 & 0 & 0.8548 & 0.8159 \\ 0.2188 & 0.4757 & 0.9462 & 0.8548 & 0 & 0.9731 \\ 0.9967 & 0.9126 & 0.0322 & 0.8159 & 0.9731 & 0 \end{bmatrix}$$

Here all the 15 pairs from $\rho_1 = (1, 2)$ till $\rho_{15} = (5, 6)$ are true. So we localize these six nodes as follows:

1. $X_1 = [0 \quad 0]$ and $X_2 = [0.2834 \quad 0]$

2. $N_{\rho_1} = \{3, 4, 5, 6\}$

3. Since $\left|N_{\rho_1}\right| = 4$, there are $2^4 = 16$ possible configurations for the placement of $A_3, \ldots A_6$.

4. If $Z_1, \ldots, Z_4$ are temporary coordinates then the configuration that corresponds to the distance matrix is:

$$\begin{bmatrix} Z_1 \\ Z_2 \\ Z_3 \\ Z_4 \end{bmatrix} = \begin{bmatrix} 0.4220 & 0.8694 \\ 0.6817 & 0.1271 \\ -0.1731 & 0.1338 \\ 0.4249 & 0.9015 \end{bmatrix}$$

5. So the initial estimated coordinates for the nodes indexed by N_{ρ_1} are: $X_3 = Z_1, X_4 = Z_2, X_5 = Z_3, X_6 = Z_4$

6. Now all of the six indices are marked as plotted. Therefore by the end of the execution of algorithm 4.3, the initial coordinate matrix becomes:

$$X = \begin{bmatrix} 0 & 0 \\ 0.2834 & 0 \\ 0.4220 & 0.8694 \\ 0.6817 & 0.1271 \\ -0.1731 & 0.1338 \\ 0.4249 & 0.9015 \end{bmatrix}$$

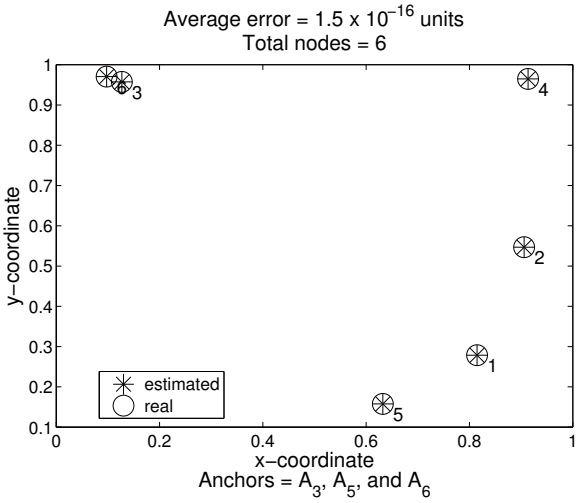


Figure 4.7: Comparison of original and estimated positions with six nodes for the working example

7. The three nodes with exact positions are A_3, A_5, A_6. Their exact positions are:

$$\begin{bmatrix} Y_3 \\ Y_5 \\ Y_6 \end{bmatrix} = \begin{bmatrix} 0.1270 & 0.9575 \\ 0.6324 & 0.1576 \\ 0.0975 & 0.9706 \end{bmatrix}$$

8. By calculating the respective vectors (line 1 of algorithm 4.4) we find that a flip to X is not required.

9. Now translating the whole topology such that X_3 is at the origin we get:

$$X = \begin{bmatrix} -0.4220 & -0.8694 \\ -0.1386 & -0.8694 \\ 0 & 0 \\ 0.2597 & -0.7423 \\ -0.5951 & -0.7356 \\ 0.0029 & 0.0321 \end{bmatrix}$$

10. The angle of rotation is found out to be, $\theta = 1.2437$ radians.

11. After rotation the updated coordinate matrix becomes:

$$X = \begin{bmatrix} 0.6877 & -0.6790 \\ 0.7788 & -0.4106 \\ 0 & 0 \\ 0.7864 & 0.0074 \\ 0.5054 & -0.7999 \\ -0.0294 & 0.0131 \end{bmatrix}$$

12. Now we translate X as a single entity such that X_3 becomes equal to the given Y_3. Thus the estimated positions of the nodes are given by the coordinate matrix:

$$X = \begin{bmatrix} 0.8147 & 0.2785 \\ 0.9058 & 0.5469 \\ 0.1270 & 0.9575 \\ 0.9134 & 0.9649 \\ 0.6324 & 0.1576 \\ 0.0975 & 0.9706 \end{bmatrix}$$

Figure 4.7 shows the comparison of the estimated positions with the real positions of the six nodes in the above working example.

4.6 Simulation Results

With the discussed approach, various simulations were performed in MATLAB® in order to test the validity of the proposed algorithm. We created scenarios by placing nodes in a rectangular region of $\mathbb{R}^2$, found their exact positions, and calculated the distances between them. The nodes with distances less than the transmission range were termed as connected and accordingly we found the distance matrix. After that we chose randomly three nodes as the anchor nodes. Then we put this only information of the distance matrix and the three positions of the anchor nodes into our algorithm, which gave us the coordinates of every other node. An output of three simulations is shown in figures 4.8, 4.9 and 4.10. For comparison we have drawn the real and the estimated positions of the nodes. Our results are better than the centroid approach of [Bulusu et al., 2000] where the average error is 1.7519 units.

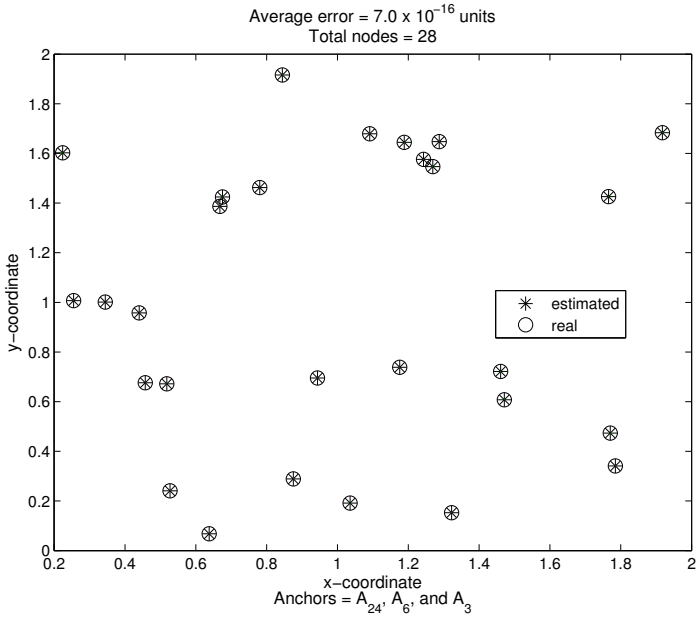


Figure 4.8: Comparison of original and estimated positions with 28 nodes

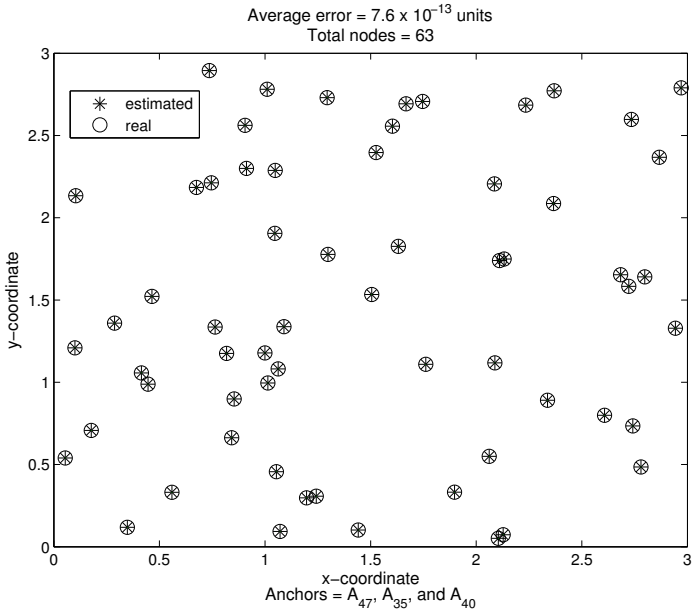


Figure 4.9: Comparison of original and estimated positions with 63 nodes

4.6.1 Computational complexity

Since the presented localization algorithm is centralized, each node has to send message to the base station. The message consists of the node id, id's of the connected neighbors, and the RSS from those neighbors. The information about the neighbors

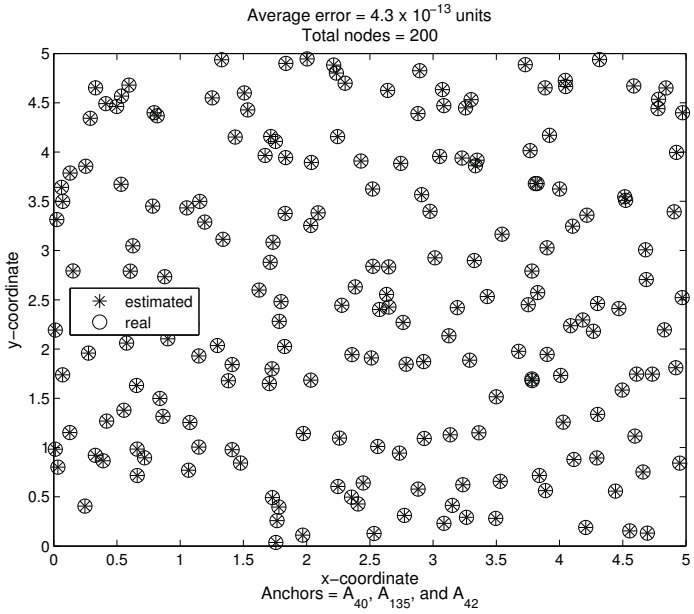


Figure 4.10: Comparison of original and estimated positions 200 nodes

of the nodes takes 8 bytes and the information about the RSS from each neighbor takes 2 bytes. So each node sends a data packet of size 16 bytes. Since there are n nodes in total so the total traffic cost is $16 \times n$. So the order of the computation complexity is $O(n)$.

4.6.2 Time complexity

The time taken by the proposed algorithm for localization of all the nodes is observed and is shown in figure 4.11. We have also plotted the curve fitting of the observed data. That curve that fits the observation is an exponential curve

$$t = ae^{bn}$$

where t and n are respectively the time in seconds and the number of nodes. The values of the constants are $a = 41.72$ and $b = 0.006143$.

4.6.3 Scalability with bounded error in measurements

When a bounded error is introduced in the distances between the interconnected nodes the results for average errors in position estimation vs number of nodes are shown in figure 4.12. Note that the bound for the error introduced is 0.1 times the

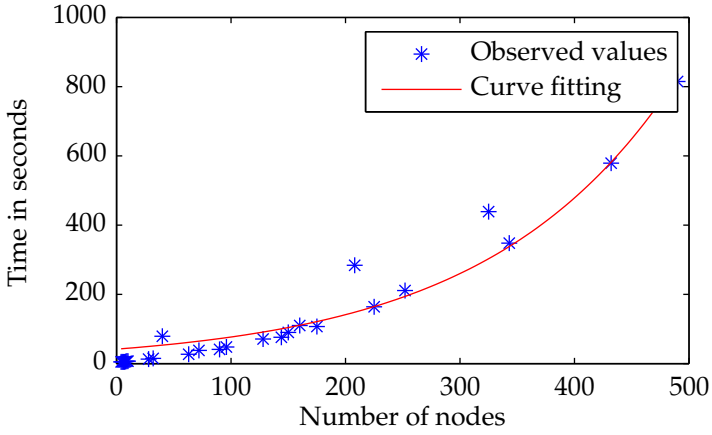


Figure 4.11: Time complexity of the proposed algorithm

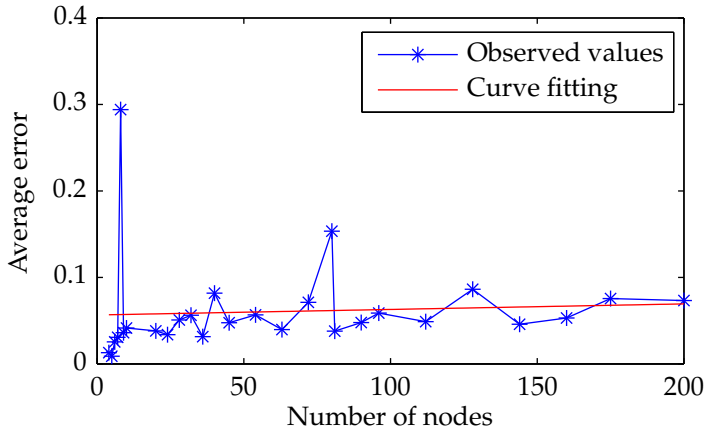


Figure 4.12: Average error in position estimation when bounded error is introduced in distance measurements

radius of connectivity. The linear polynomial curve fitting of the observed data is given by:

$$\text{error} = 6.408 \times 10^{-5} n + 0.05658$$

In figure 4.13 we have shown a result of the position estimation of 160 nodes.

4.7 Conclusion

In this chapter we have proposed a process of finding the estimated positions of the sensor nodes in a wireless sensor network. After performing the simulations and the validation of our algorithm in the appropriate scenarios we have seen that

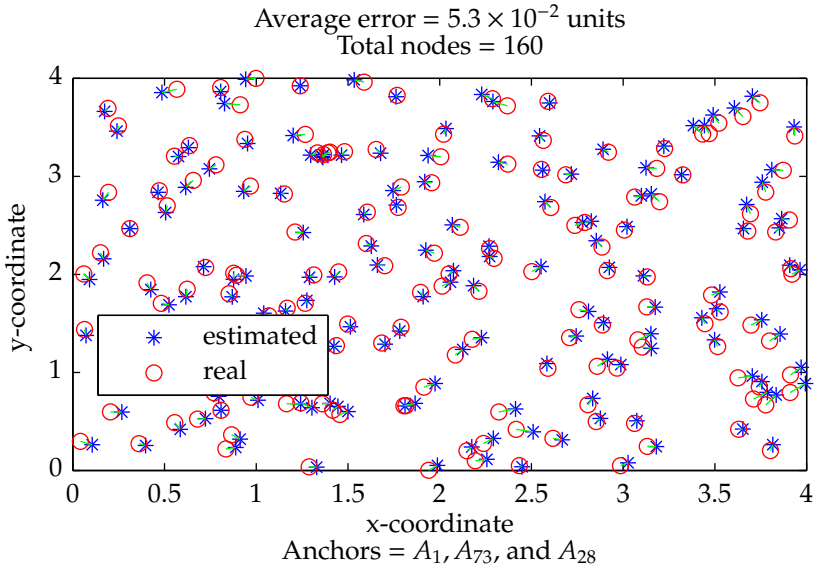


Figure 4.13: Estimated positions with error introduced in the distance measurements

this is an efficient method of localization that uses the information from the distance matrix. For the localization purpose, our method requires the position of exactly three nodes. In other words using the proposed localization algorithm we need only three anchor nodes. We also conclude that three anchors is the necessary and sufficient condition for such types of networks in which a node is connected to at least three other nodes, and that the one-hop neighbors of anode form a ring. After performing the simulations we have seen that position estimation of the nodes has sufficiently increased. In the future perspective we aim to improve this algorithm for all types of wireless sensor networks with less number of constraints about the connectivity amongst the nodes. As the distance information amongst the connected nodes is vital for the localization, therefore, it is crucial to pay attention to the sources that create errors in distance estimations. One such source is the droop of battery voltage. This aspect is discussed in the next chapter.

Chapter 5

Signal Strength Loss Compensation

Received Signal Strength Indication (RSSI) plays a vital role in the range-free localization of sensor nodes in a wireless sensor network and a good amount of research has been made in this regard. One important factor is the battery voltage of the nodes (i.e., the MICAz sensors) which is not taken into account in the existing literature. As battery voltage level performs an indispensable role for the position estimation of sensor nodes through anchor nodes therefore, in this chapter, we take into a account this crucial factor and propose an algorithm that overcomes the problem of decaying battery. We show the results, in terms of more precise localization of sensor nodes through simulation. This portion of the work is presented in [Khan et al., 2010b] and extended in [Khan et al., 2011] where we include the use of neural network to overcome the localization errors generated due to gradual battery voltage drooping.

5.1 Introduction

Wireless Sensor Networks (WSNs) have become important in the fields of military defence and environmental sciences. Their applications are also found in home and ubiquitous environments. A wireless sensor network is a network of distributed nodes that monitor the physical changes like temperature, pressure, vibrations, and motion/breaches at the desired locality, to name a few, [Akyildiz et al., 2002a, Akyildiz et al., 2002e, Bergamo and Mazzini, 2002]. The nodes are fully equipped to measure these changes. After an observation is made it travels in the form of data packets from node to node till it reaches its destination. For this purpose the sensors are also equipped with transceiver antennas. To perform these activities the nodes use

Figure 5.1: MICAz mote for which the idea is proposed. These motes loose their battery voltage with the passage of time.

energy provided by the attached battery.

One of the many important issues in the WSNs is the localization [Yun et al., 2009, Kim and Kwon, 2005, He et al., 2003]. There are certain applications of the WSNs, e.g., environmental monitoring like forest fire observation, and like intrusion, in which the location of the information source is very important. The received data is meaningless unless the position of the event occurrence is known. There are two types of the localization schemes: range-based and range-free, [Yun et al., 2009, Kim and Kwon, 2005, He et al., 2003]. From localization point of view there are two types of the nodes in a WSN: anchor nodes and tracked nodes. Anchors are equipped to know their position/location, either by GPS or by pre-configuration [Bahl and Padmanabhan, 2000, Hightower et al., 2000, Rappaport et al., 1996]. But for sensors, neither they are equipped with GPS nor their locations are pre-configured. In range-based schemes sensors are localized with the known positions of anchors by measuring the angle of arrival (AoA), time of arrival (ToA), or time difference of arrival (TDoA) [Niculescu and Nath, 2003b, Cong and Zhuang, 2002]. For this scheme to be applicable we need to employ certain devices in order to measure one or all of the three quantities. In range-free schemes sensors are localized with the help of known positions of anchors but without the use of AoA, ToA, or TDoA. Most of the time the distance between a sensor and an anchor is calculated using RSSI.

Figure 5.1 shows a MICAz sensor that is used in our experiments. In [Crossbow_Technology_Incorporation,] it is shown that with the passage of time these sensors start losing battery voltage.

The drop in the battery voltage of MICAz sensor is measured over a period of 200 hours and is graphically presented in figure 5.2. This drop in battery voltage

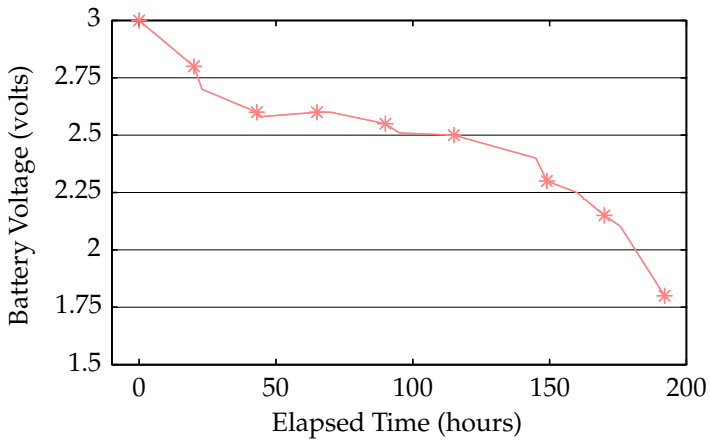


Figure 5.2: Plot of the battery voltage of MICAz sensor against time over the period of two hundred hours

may lead to an erroneous sensor location.

A range-free geometric approach towards the computation of the sensor nodes location is given in [Bahl and Padmanabhan, 2000, Yun et al., 2007]; where the estimated position of a sensor is the centroid of positions of connected anchors. If $\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3, \cdots, \mathbf{r}_n$ are the positions of connected anchors to a particular sensor, then the estimated position of that sensor is given by (5.1).

$$\mathbf{r}_{est} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{r}_i \tag{5.1}$$

In [Hightower et al., 2000] an improved version of this geometric approach is given which utilizes the weighted average method. For a particular sensor, a weight, called *edge weight*, is assigned to a connected anchor according to its proximity. Suppose $w_1, w_2, w_3, \cdots, w_n$ are the weights assigned to connected anchors in reference to a sensor then the estimated position of the sensor is given by (5.2).

$$\mathbf{r}_{est} = \frac{\sum_{i=1}^{n} w_i \mathbf{r}_i}{\sum_{i=1}^{n} w_i} \tag{5.2}$$

There are two more techniques given in [Yun et al., 2009] for sensor localization. Fuzzy logic system (FLS) and genetic algorithm are used in one of them and the other is carried out by the implementation of neural networks.These techniques produce better results as compared to those presented in [Bahl and Padmanabhan, 2000] and [Hightower et al., 2000].

In this chapter, the energy considerations [Anastasi et al., 2009] are also taken into account while calculating the edge weights of the anchors. The edge weights calculated from erroneous RSSI are less informative due to drooping battery voltage leading to misinterpretation about exact sensor location. So we need to introduce a compensation term that will recoup the edge weight. An algorithm is given to find that compensation term and to calculate the enhanced location of the sensors.

We also present a solution to this problem through the use of neural networks. Although a soft computing technique for localization is used by [Yun et al., 2007, Yun et al., 2009] but they have not considered the voltage decrease in node batteries. When a neural network is trained with real data, it can learn the pattern between the input variables and the output variables. In our case the input variables are the battery voltage, the time elapsed since the node is in working mode and the RSSI observed. The single output variable is the real distance. Note that the distance obtained by the RSSI – distance model presented in section 5.3 could be erroneous. Therefore it is better to deal the relationship between the observed RSSI, voltage, time elapsed and the real distance through a neural network.

The rest of the chapter is arranged as follows: In section 5.2 we present our idea and we formulate the problem. In section 5.3 we present our solution and devise an algorithm for the better performance regarding the localization of the senors. We analyze and implement our solution by introducing the new parameter of battery voltage decay in section 5.4. The treatment of the problem by using neural network is given in section 5.5. Finally section 5.6 gives the conclusion, demonstrating that adhering systematic attention to the battery voltage decay results in elevated performance of sensor localization through anchor nodes in a WSN.

5.2 Problem Formulation

In WSN anchor nodes are fully aware of their position in the geographical region and we need to find the sensor node positions. Sensor positions are calculated based on the anchor positions. The transmission range of an anchor is assumed to be spherical which is reduced to a circular region in two dimensional space. By transmission range we mean an area where a node can detect an other node. A sensor node assigns edge weights to anchor nodes by measuring strength in their respective ping signals. Thus we get an initial estimate of sensor position by (5.2). The greater the RSSI observed from an anchor, the greater the edge weight is assigned by the

sensor. That is the edge weight assigned has a positive correlation with the RSSI. The battery voltage level is gradually drooped due to its usage by the signal emitter of the anchor node. This battery voltage decay results in the reduced power supply to the emitter. The emitter then sends signals with reduced strength and declined RSSI. This leads to a misinterpretation about the distance between the sensor and the anchor. The implication is obvious that the error in distance measure is increased with decaying battery. In this chapter we are concerned with the calculation of the edge weight which is invariant to the battery voltage level. Thus we need to find out the compensation term that is added to the measured edge weight so that the resulting edge weight is reported as constant by a sensor for a particular anchor irrespective of the battery voltage.

Suppose the relationship between the edge weight w and the battery voltage V, for a particular anchor-sensor node pair is presented by (5.3).

$$w = f(V) \tag{5.3}$$

If V_0 is the maximum voltage, then the maximum edge weight is $w_0 = f(V_0)$. As a matter of fact we should have $f(V) = w_0$ for all battery voltage levels V. Thus with the decay in battery voltage the difference, denoted by $\chi(V) = |f(V_0) - f(V)|$, between the real edge weight and the observed edge weight increases. Thus $\chi(V)$ is the compensation term that we shall add to the observed edge weight to get the real edge weight. We denote by $g(\cdot, \cdot)$ a function of two variables; the observed edge weight w_{ob} and the battery voltage V of the anchor node, returning the real edge weight.

$$g(w_{ob}, V) = w_{ob} + \chi(V) \tag{5.4}$$

First of all we try to find out the relationship between the time elapsed t and the battery voltage V of the MICAz mote. The degree two and degree three polynomial approximations of battery voltage as a function of time elapsed are respectively given by (5.5) and (5.6), and are graphically shown in figure 5.3. Table 5.1 shows the difference between polynomially approximated values and the observed battery voltage at various time instances measured in hours. The quadratic and cubic polynomial approximations are providing us with results close to the observed data. We shall use the degree two approximation as a trade-off between better performance

and increasing computation complexity.

$$V(t) = a_{22}t^2 + a_{21}t + a_{20} \tag{5.5}$$

$$V(t) = a_{33}t^3 + a_{32}t^2 + a_{31}t + a_{30} \tag{5.6}$$

where

$$\begin{aligned} a_{33} &= -5.444 \times 10^{-7} \\ a_{32} &= 1.426 \times 10^{-4} \quad & a_{22} &= -1.194 \times 10^{-5} \\ a_{31} &= -1.342 \times 10^{-2} \quad & a_{21} &= -1.914 \times 10^{-3} \\ a_{30} &= 2.983 \quad & a_{20} &= 2.817 \end{aligned}$$

An anchor node with low battery voltage will send signal with low RSSI. If this

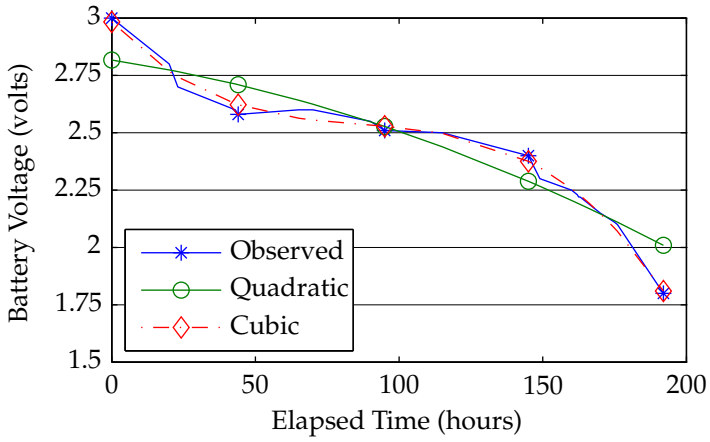


Figure 5.3: The degree two and degree three polynomial approximations for the battery voltage against the time elapsed in hours

fact is avoided then sensor node shall conclude that the anchor node is at a farther distance than it really is. Now we shall compensate for this misinterpretation by the sensor node and we start our solution formulation in the next section.

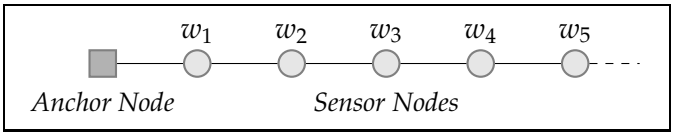
5.3 Presentation of the solution

Consider one dimensional case with one anchor node at the origin and just one sensor node. By making sure that the battery of the anchor node is fully charged we calculate the edge weights by placing the sensor node at different positions as is shown in figure5.4(a).

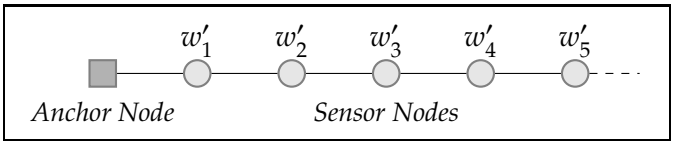
With the fully charged battery if the measured edge weights are w_1 , w_2 , w_3 , $\cdots$,

Table 5.1: Difference between polynomial approximated values and the observed battery voltage

Time	Linear Error	Quadratic Error	Cubic Error
020	0.08141	0.02605	0.03271
023	0.18012	0.06666	0.04315
043	0.27153	0.11262	0.02632
065	0.26208	0.04214	0.03632
090	0.30134	0.00197	0.01660
115	0.34060	0.06101	0.00238
149	0.52600	0.03327	0.04844
162	0.59042	0.03642	0.00682
163	0.59999	0.03221	0.00662
166	0.62870	0.01974	0.00452
170	0.66698	0.00344	0.00189
176	0.71441	0.01028	0.02969
192	1.00754	0.20935	0.00996



(a) *Fully charged battery*



(b) *Partially charged battery*

Figure 5.4: Weights measured by the sensor node at different positions at different voltage levels of the battery of the anchor node

then we have the relationship $w_1 > w_2 > w_3 > \cdots$. At a later stage the weights measured by the sensor node are observed as shown in figure5.4(b). When the battery voltage level has drooped, once again we have the inequalities $w'_1 > w'_2 > w'_3 > \cdots$, but the signal strength is decreased. Thus we can have a situation in which $w'_i < w_i$, (for $i = 1, 2, 3, 4,$ or 5) whereby giving the misinterpretation about the location of the sensor as we know that the distance is constant. Although the location of the sensor is same but the edge weight is lower than its preceding case. The sensor shall conclude that it is farther from the anchor than it really is. This is because the RSSI is weak and low edge weight is calculated.

Now let us find the relationship between the battery voltage level and observed

edge weight for a fixed distance between sensor-anchor pair. In an experiment the RSSI at different sensor-anchor distances was measured and the results are shown in figure 5.5. The cubic polynomial approximation from the observed data for distance

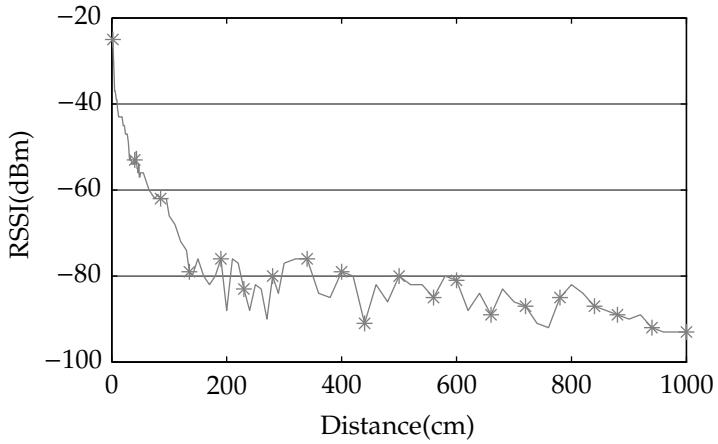


Figure 5.5: RSSI measured against distance for MICAz sensor

d in centimeters as a function of RSSI measured in dBm is given by (5.7).

$$d = a_3\text{RSSI}^3 + a_2\text{RSSI}^2 + a_1\text{RSSI} + a_0 \tag{5.7}$$

where

$$\begin{aligned} a_3 &= -0.007791 & a_1 &= -47.98 \\ a_2 &= -1.058 & a_0 &= -701.6 \end{aligned}$$

The relation of RSSI in dBm to the power P in mW is shown in (5.8).

$$\text{RSSI} = 10\log_{10}(P) \tag{5.8}$$

Edge weight is calculated from the RSSI which is the measure of the power of the signal. By the well known Watt's law the power, P, is the product of voltage, V, and current, I, i.e., $P = VI$ and by Ohm's law we have $V = IR$. Combining these two laws we get $P = V^2/R$. If the resistance remains constant, then we see that the power is directly proportional to the square of the voltage. By letting $R = 1\Omega$ we observe that with the decrease in the voltage of the battery of MICAz mote, there is decrease in the power of the sent signal as is shown in table 5.2. Thus for a constant distance between the sensor-anchor pair the relationship between the power of the signal and the battery voltage is $P = V^2$. The observed edge weight decreases as the battery voltage droops. As a mater of fact we need the edge weight to remain constant for

Table 5.2: Power in the MICAz mote signal at different battery voltages

Voltage (volts)	**Power** (watts)	**Voltage** (volts)	**Power** (watts)
1.54	2.37	2.51	6.30
2.25	5.06	2.55	6.50
1.85	3.42	2.59	6.71
2.15	4.62	2.60	6.76
2.05	4.20	2.60	6.76
2.29	5.24	2.62	6.86
2.43	5.90	2.66	7.08
2.47	6.10	2.80	7.84
2.49	6.20	2.94	8.64
2.50	6.25	3.00	9.00

a fixed sensor-anchor pair distance.

As shown in figure 5.5, at the distance of 10 cm, the RSSI of the signal is -41dBm and the voltage is 3 volts. For this voltage the power in the signal using $P = 10^{(RSSI/10)}$ is 7.94×10^{-5}mW. For $V = 3$ we have $P = 7.94 \times 10^{-5}$ and we know that for $V = 0$, $P = 0$. As P is proportional to the square of V, it means that 9 corresponds to 7.94×10^{-5}. Thus power of the sent signal as a function of battery voltage is given by (5.9) where $\alpha = 8.82 \times 10^{-6}$. Figure 5.6 shows the plot of power in the signal in relation to the battery voltage.

$$P(V) = \alpha V^2 \tag{5.9}$$

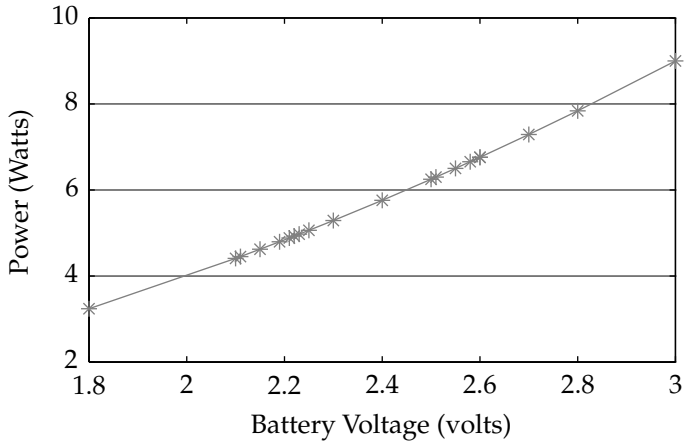


Figure 5.6: The plot of power in signal as a function of battery voltage

As edge weight w is a function of RSSI, let us assume that it is an identity function:

$$
\begin{aligned}
& & w &= \text{RSSI} \\
(5.8) & \implies & w &= 10\log_{10}(P) \\
(5.9) & \implies & w &= 10\log_{10}(\alpha V^2) \\
\therefore & \implies & w &= f(V) = 10\log_{10}(\alpha V^2).
\end{aligned}
$$

Hence the maximum edge weight is $w_0 = -41$ and the compensation term is given by (5.10).

$$\chi(V) = \left| w_0 - 10\log_{10}(\alpha V^2) \right| \tag{5.10}$$

The graph of the compensation term at different voltage levels is shown in figure 5.7. The proposed technique is mentioned in algorithm 5.1 that computes the improved estimated position of the sensor nodes by compensating the edge weight loss due to battery voltage decay.

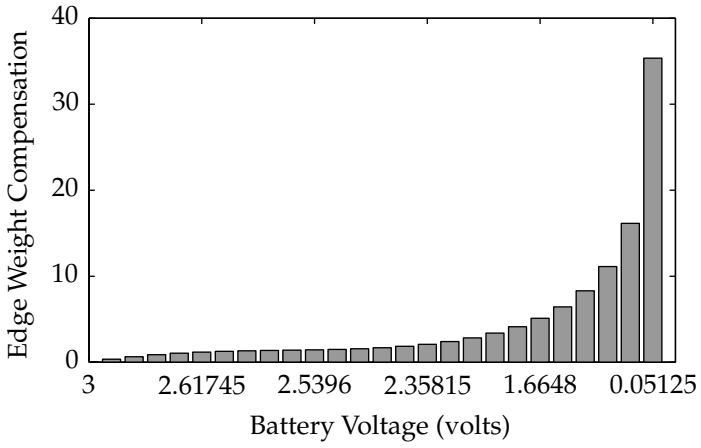


Figure 5.7: Edge weight compensation at different battery voltages. It increases with the decrease in voltage.

Algorithm 5.1 Calculation of compensated edge weight and enhanced estimated position of sensor node

Input: RSSI = R_{ij} , Battery Voltage = V
Output: Estimated position = $\mathbf{r}_{est}$
1: **for** $i = 1$ **to** k **do**
2: **for** $j = 1$ **to** n **do**
3: **Require** R_{ij} for *ith* sensor and *jth* anchor
4: $w_j \longleftarrow h(R_{ij})$
5: **Require** V
6: $\chi(V_j) \longleftarrow \left| w_j^0 - 10\log_{10}(\alpha(V_j)^2) \right|$
7: **return** $w_j := w_j + \chi(V_j)$
8: **end for**
9: **return** $\mathbf{r}_{i_{est}} := \frac{\sum_{j=1}^{k} \mathbf{r}_j w_j}{\sum_{j=1}^{k} w_j}$
10: **end for**

5.4 Simulation Results

Various simulations were performed using MATLAB® in order to validate the proposed algorithm 5.1. At any instant the compensation term for the observed edge weight is found as shown in (5.10). With the help of this compensation term the corrected edge weight is obtained as shown in line 7 of algorithm 5.1. Lastly with the help of the corrected edge weight the correct distance is calculated. In figure 5.7 we have the graph of the edge weight compensation term plotted against decreasing battery voltage. We see that with the decrease in battery voltage the magnitude of the weight to be compensated is increased. If w_{ob} represents the observed edge weight at any instant then the corrected edge weight w is given by putting for corresponding values in (5.4) as shown in (5.11).

$$w = g(w_{ob}, V) = w_{ob} + \left| w_0 - 10\log_{10}(\alpha V^2) \right| \tag{5.11}$$

Figure 5.8 shows that there is a remarkable decrease in the observed edge weight with the decrease in battery voltage. When the compensation term is found out and the corrected edge weight is computed we see that the corrected weight remains constant for a particular distance irrespective of the battery voltage. The difference between the observed distance and the corrected distance with the increase in time is shown in figure 5.9. Finally the comparison of the three positions is shown in the

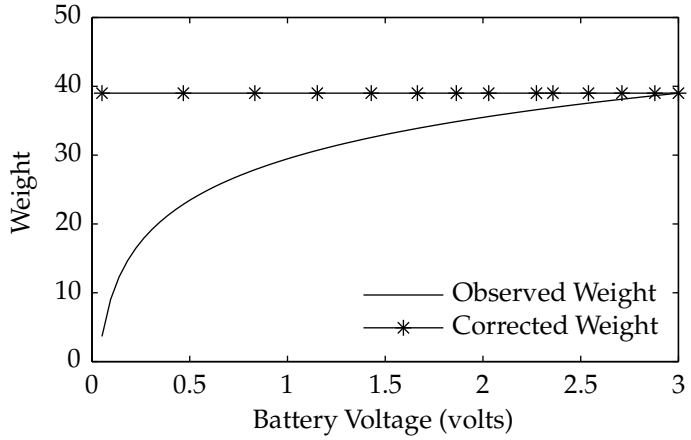


Figure 5.8: Plot of observed and corrected edge weights against the battery voltage

figure 5.10. Here we see that with the help of the compensation term added to the observed weight the localization of the concerned sensor node is improved.

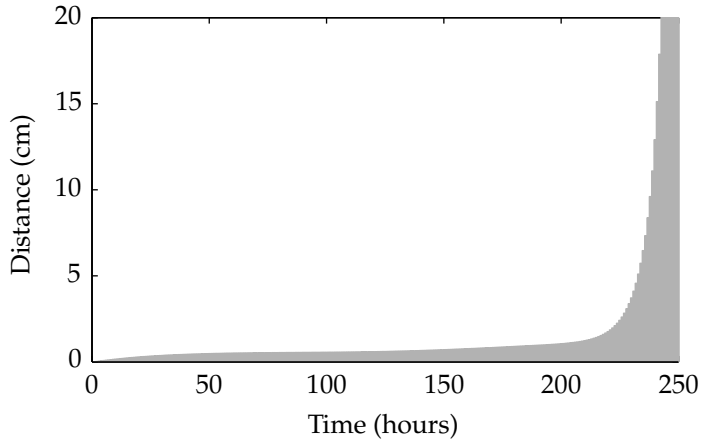


Figure 5.9: Difference between the observed and corrected positions

5.5 Treatment through Neural Network

Neural networks imitate human brain to perform intelligent tasks [Hagan et al., 1996, Bishop, 1996]. A neural network is made to learn and approximate the complicated relationships between input and output variables, and acquire knowledge about these relationships directly from the training data. A schematic diagram of the used neural network is shown in figure 5.11. We have used a multilayer perceptron neural

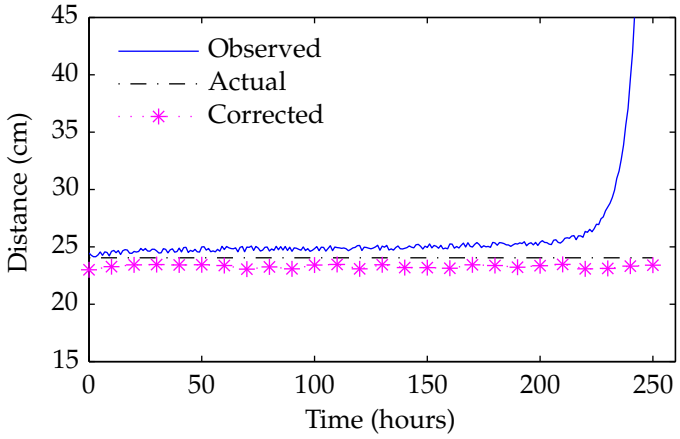


Figure 5.10: The corrected distance improves the localization of the sensor nodes

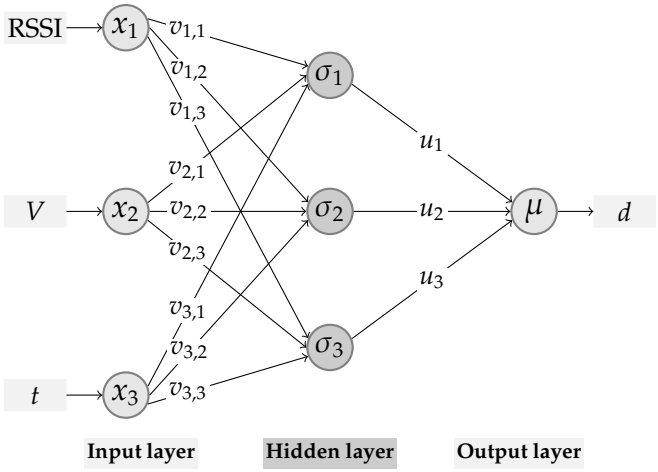


Figure 5.11: Three layered neural network with three input variables and one output variable

network with three layers: an input, a hidden and an output layer. The hidden layer has three neurons and the activation function of each of the neuron is the logsigmoid function. The components, in order, of the input vector $\mathbf{x}^T = (x_1\ ,\ x_2\ ,\ x_3)$ respectively are RSSI, voltage and the time elapsed. The square matrix of order three $\mathbf{V} = \left[v_{i,j}\right]$ represents the input-to-hidden layer weights. The activation functions of each of the hidden layer neuron are denoted by σ_i for $i = 1, 2, \text{or } 3$. Each of the σ_i is the logsigmoid function. These three activation functions are represented by a vector $\boldsymbol{\sigma}^T = (\sigma_1\ ,\ \sigma_2\ ,\ \sigma_3)$. The vector $\mathbf{u}^T = (u_1\ ,\ u_2\ ,\ u_3)$ represent the hidden-to-output layer weights. The activation function of the output layer is denoted by μ and is the linear

identity function. The single scalar output d is the real distance between the nodes:

$$d \approx \mu \left\{ \sum_{j=1}^{3} u_j \sigma_j \left(\sum_{i=1}^{3} x_i v_{i,j} \right) \right\}$$

In matrix form which is written as:

$$F_{NN}(\mathbf{x}) = \mu \left\{ \mathbf{u}^T \sigma \left(\mathbf{V}^T \mathbf{x} \right) \right\}$$

One of the neural network simulation result is shown in figure 5.12. Here we have obtained the estimated positions of the sensors with the help of distances obtained as an output of the neural network. These estimated positions are quite better than the positions obtained only from the observed RSSI. As shown in the

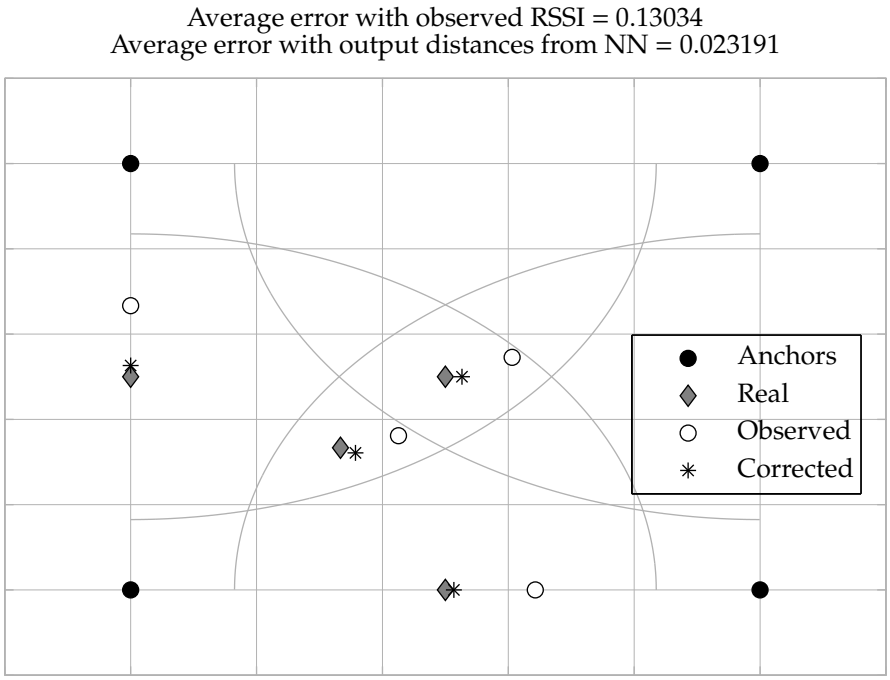


Figure 5.12: Black dots are the anchors and the grey diamonds are the real positions, circles are the positions estimated from the observed RSSI and the black asterisks are the corrected positions

figure there are four anchors at the vertices of a square of length $1.4r$ where r is the radius of transmission. Four senors are placed at different zones of connectivity. The network is already trained for the three variables. With the passage of time the error

in the RSSI measure starts increasing. The observed RSSI is used to estimate the positions of the sensors. The average error in this case is 0.13 units. Now by using the distances obtained as the output variable of the neural network we get a better estimated position of the sensor nodes. The average error in the corrected estimated positions becomes 0.02 units. Hence the battery voltage consideration yields in a more reliable localization in wireless sensor networks.

5.6 Conclusion

In this chapter we have seen that the weight assigned to an anchor node due to observed RSSI when measured without paying attention to the battery level of that anchor node may lead to a misinterpretation about the distance between the respective anchor and sensor nodes. We have proposed a compensation term in the calculation of the edge weight that improves the accuracy of the distance between the concerned anchor and sensor nodes. With this added value of adherence to the battery voltage level of the anchor nodes, the localization of the sensor nodes in a WSN is improved. The battery voltage, the emitted power, and the received power are noticed. With the help of the proposed algorithm, before the estimation of the distance, the compensation term, if needed, is added to the observed weight of the anchor node. Hence the uncertainty in the positions of the sensor nodes in a WSN, due to the proposed algorithm is reduced. The use of neural network techniques drastically reduces the computation complexity of the otherwise erroneous position estimation of the sensor nodes. The neural network also demonstrates the fact that battery voltage consideration gives an enhanced position estimation of the nodes as is shown in a simulation result. Thus the localization in wireless sensor network is improved when the time elapsed and the voltage droop are taken into account. The future works also include to tackle with the situation in which the loss in signal strength is not due to the battery voltage drooping.

Chapter 6

Conclusion and Future Perspectives

6.1 Conclusion

In this thesis three themes related to wireless sensor networks (WSNs) are covered. The first part of the thesis focuses on the detection of faults in a WSN. There is always a possibility that a sensor of a node is not giving accurate measurements all of the time. Therefore, it is necessary to find if a node has developed a faulty sensor. With the precise information about the sensor health, one can determine the extent of reliability on its sensor measurement. To equip a node with multiple sensors is not an economical solution. Thus the sensor measurements of a node are modeled with the help of the fuzzy inference system (FIS). For each node, both recurrent and non-recurrent systems are used to model its sensor measurement. An FIS for a particular node is trained with input variables as the actual sensor measurements of the neighbor nodes and with output variable as the real sensor measurements of that node. The difference between the FIS approximated value and the actual measurement of the sensor is used as an indication for whether or not to declare a node as faulty.

In the second part of the thesis a position estimation method for localization of nodes in a WSN is proposed. Once the intermediate distances between the connected nodes have been calculated, the task remained to be accomplished is to find the geographical position of all the nodes. In order to do so, the nodes require some reference points or landmarks with known positions to calculate their own location in relation to these landmarks. If some nodes with known positions are used as landmarks, such nodes are called *anchor nodes* or simply as *anchors*. In the proposed localization algorithm anchor nodes are used as landmark points. Many attempts

have been proposed that address the problem of reducing the number of anchors for localization in a WSN. Some of these are cooperative approaches in which the relative placements of adjacent nodes are also taken into account in addition to the known positions of the anchors. Usually the anchors are placed at the boundary of the WSN. The localization method proposed here does not require any constraint on the placement of the anchors; rather any three randomly chosen nodes can serve as anchors.

There are two steps involved in the position estimation of all the nodes by using the proposed localization method. The first step is to find a relative topology of the WSN nodes. The second step is to find the symmetry, orientation, and position of the topology in the plane. A heuristic approach is used to find the relative topology with the help of distance matrix. The purpose of the distance matrix is to indicate whether or not a pair of nodes has a connection between them and in case of connectivity it gives the estimated distance between the nodes. By using the information of connectivity between the nodes and their respective distances the topology of the nodes is calculated. This method is heuristic because it uses the point solution from the intersection of two circles instead of conventional triangulation method, where a system of three quadratic equations in two variables is used whereby the computational complexity of the position estimation method is increased.

When two connected nodes have another node in common, then by using the information of distances between these interconnected nodes, two possible positions are calculated for the third node. The presence or absence of a connection between the third node and a fourth node helps in finding the accurate possibility out of the two. This process is iterated till all the nodes have been relatively placed.

Once the relative topology has been calculated, we need to find the exact symmetry, orientation, and position of this topology in the plane. It is at this moment the knowledge of three nodes positions comes into action. From the relative topology we know the temporary coordinates of the nodes. By having a comparison of certain characteristics between the temporary coordinates and the exact coordinates; first the symmetry of relative topology is obtained that would correspond to the original topology. In other words it tells whether or not the relative topology is a mirror image of the original topology. Then some geometrical operators are used to correct the topology position and orientation. Thus, all the nodes in the WSN are localized using exactly three anchors.

In itself the process was challenging to find the planar layout of the nodes by

using only the connectivity information amongst them. The assumptions made in the proposed algorithm are more or less the same as explicitly or implicitly stated in other research works. That is there are no orphan nodes (a node whose node degree = 1), the network is not divisible into two subnetworks such that both of them have either one node or one link in common. And that each node has at least a node degree of 3. In case of densely deployed network, the number of steps to find the topology of the network are reduced by ignoring the redundancy in connectivity. Hence providing an efficient algorithm as far as localization is concerned. The strong point is that our algorithm uses the distance matrix and exactly three anchors.

The last part concerns the power loss in a node signal due to voltage droop in the battery of the node. There are multiple localization methods that use the received signal strength (RSS) to calculate the distance between the connected nodes. There is a negative correlation between the RSS and the emitter-receiver distance. If a WSN node in receiving mode measures a low value RSS from a transmitting node, an obvious interpretation is the increase of separation between the two nodes. Thus, an error in RSS measurement shall manifest itself in the form of incorrect calculation of distance between the concerned nodes. Therefore, for such localization methods knowing sources that create error in RSS are very important.

One such source is the decrease in the battery voltage of the emitter node. With decaying battery the transmitter of an emitter node will receive less energy and hence will send signals with less power. Therefore, at the receiving node, the power in the signal is even less. Thus a decrease in RSS could have two explanations. It could either be due to the increase in distance between the transmitting node and the receiver node; or it could be due to the loss of battery voltage at the transmitting node. Hence the change (decrease) in the RSS due to the change (decrease) in the battery voltage of the sending node would lead to misinterpretation in terms of increase in the distance between the nodes. Consequently, paying attention to the battery voltage of the emitter node is very crucial for the RSS based localization methods.

In the last part of the thesis a method is proposed to compensate for the apparent increase in the calculated distance between the related nodes due to decrease in the voltage of the signal sending node battery. This objective is achieved by studying the relation between the decrease in battery voltage and the time elapsed since the node is in working mode. Then the relation between the RSS and the distance between the connected nodes with fully charged batteries is calculated. Afterwards the RSS is

measured by varying the battery voltage of the emitter node and keeping the receiver node at a constant distance. Finally, a function is proposed whose arguments are the apparently observed RSS and the current voltage of the emitter node battery. The return of the function is the corrected RSS that corresponds to the actual distance amongst the connected nodes. Hence increasing the efficiency of the RSS based localization methods in WSNs.

6.2 Perspectives

In the following we list the possible works that emerge from the work done in the present thesis:

1. In the proposed localization algorithm there is a strict constraint on the connectivity between the nodes. A natural extension of the work is to relax the conditions on the connectivity between the nodes.

2. We shall modify the localization algorithm so that it is applicable to a rapidly changing topology. In the present form, a rapidly changing topology shall ignite a rapid change of messages between the nodes and the base station, which could create a congestion in the network and eventually a loss in the information being transferred.

3. There are multiple number of scenarios in wireless sensor networks for which the presented localization scheme can be extended. For example, one such scenario is a WSN where the nodes do not form a convex set.

4. We shall look for the refinement of RSS-distance models. There are multiple attenuation sources that result to inaccuracy in the measurement of RSS. A future work is to tackle such error creating sources.

5. Although at present the computational complexity of the localization algorithm is $O(n)$, where n is the total number of nodes. We shall be working on to reduce even further the computation complexity. It means that we shall evolve the positioning strategy to reduce the number of exchange of messages amongst the nodes.

6. Making a distributed version of the present scheme would be very interesting to work in the future.

7. Equally interesting would be the extension of the scheme in 3D.

8. In future we are also interested in fault detection strategies that can also handle non-homogenous physical quantities.

References

[Tin, 2004a] (2004a). Tinyos community forum. from http://www.tinyos.net/.

[Tin, 2004b] (2004b). Tinyos community forum related work. from http://www.tinyos.net/related.html.

[Agre and Clare, 2000] Agre, J. and Clare, L. (2000). An integrated architecture for cooperative sensing networks. *IEEE Computer*, 33(5):106 –108.

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

[Akyildiz et al., 2002b] Akyildiz, I., Su, W., Sankarasubramaniam, Y., and Cayirci, E. (2002b). Wireless sensor networks: A survey. *Computer Networks*, 38(4):393–422.

[Akyildiz et al., 2002c] Akyildiz, I., Su, W., Sankarasubramaniam, Y., and Cayirci, E. (2002c). Wireless sensor networks: A survey. *Computer Networks*, 38(4):393–422.

[Akyildiz et al., 2002d] Akyildiz, I., Su, W., Sankarasubramaniam, Y., and Cyirci, E. (2002d). Wireless sensor networks: A survay. *Computer Networks*, 38(4):393–422.

[Akyildiz et al., 2002e] Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., and Cayirci, E. (2002e). Wireless sensor networks: a survey. *Computer Networks*, 38(4):393–422.

[Alfakih et al., 1999] Alfakih, A., Khandani, A., and Wolkowicz, H. (1999). Solving Euclidean distance matrix completion problems via semidefinite programming. *Computational Optimization and Applications*, 12(1):13–30.

[Anastasi et al., 2009] Anastasi, G., Conti, M., Francesco, M. D., and Passarella, A. (2009). Energy conservation in wireless sensor networks: a survey. *Ad Hoc Networks*, 7(3):537–568.

[Bahl and Padmanabhan, 2000] Bahl, P. and Padmanabhan, V. (2000). RADAR: An in-building RF-based user location and tracking system. In *INFOCOM Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies*, pages 775–784.

[Bergamo and Mazzini, 2002] Bergamo, P. and Mazzini, G. (2002). Localization in sensor networks with fading and mobility. In *The Thirteenth IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*.

[Biagioni, 2001] Biagioni, E. (2001). Pods: Interpreting spatial and temporal environmental information. In *In Usability Evaluation and Interface Design: Cognitive Engineering, Intelligent Agents, and Virtual Reality, Volume I of the Proceedings of HCI International 2001, the 9th International Conference on Human-Computer Interaction*, pages 317–321.

[Biagioni and Bridges, 2002] Biagioni, E. S. and Bridges, K. W. (2002). The application of remote sensor technology to assist the recovery of rare and endangered species. *International Journal of High Performance Computing Applications*, 16(3):112–121.

[Bischoff et al., 2006] Bischoff, U., Strohbach, M., Hazas, M., and Kortuem, G. (2006). Constraint-based distance estimation in ad-hoc wireless sensor networks. *Wireless Sensor Networks*, 3868:54–68.

[Bishop, 1996] Bishop, C. M. (1996). *Neural Networks for Pattern Recognition*. Oxford University Press Inc., New York.

[Bonnet et al., 2000] Bonnet, P., Gehrke, J., and Seshadri, P. (2000). Querying the physical world. *IEEE Personal Communications*, 7(5):10 –15.

[Bose et al., 2001] Bose, P., Morin, P., Stojmenovic, I., and Urrutia, J. (2001). Routing with guaranteed delivery in ad hoc wireless networks. *Wireless Networking*, 7(6):609–616.

[Bulusu et al., 2000] Bulusu, N., Heidemann, J., and Estrin, D. (2000). GPS-less low-cost outdoor localization for very small devices. *IEEE Personal Communications*, 7(5):28–34.

[Calamari, 2004] Calamari (2004). Calamari: A sensor field localization system. from http://www.cs.berkeley.edu/kamin/calamari/.

[Cassioli, 2009] Cassioli, A. (2009). Solving the sensor network localization problem using an heuristic multistage approach. Technical report, Universita degli Studi di Firenze.

[Chandy and Lamport, 1985] Chandy, K. M. and Lamport, L. (1985). Distributed snapshots: determining global states of distributed systems. *ACM Trans. Comput. Syst.*, 3:63–75.

[Chen et al., 2006] Chen, J., Kher, S., and Somani, A. (2006). Distributed fault detection of wireless sensor networks. In *Proceedings of DIWANS Workshop*, pages 65–72.

[Chessa and Santi, 2001] Chessa, S. and Santi, P. (2001). Comparison-based system-level fault diagnosis in ad hoc networks. In *20th IEEE Symposium on Reliable Distrubyted Systems*, pages 257–266.

[Cong and Zhuang, 2002] Cong, L. and Zhuang, W. (2002). Hybrid TDoA/AoA mobile user location for wideband CDMA cellular systems. *IEEE Transactions on Wireless Communications*, 1(3).

[CotsBots, 2004] CotsBots (2004). Cotsbots. from http://www-bsac.eecs.berkeley.edu/projects/cotsbots/.

[Crossbow_Technology_Incorporation,] Crossbow_Technology_Incorporation. Battery life test for MICAz mote. Technical report. Available online at http://www.xbow.com/Support/Support_pdf_files/MICA2_BatteryLifeTest.pdf.

[Dai and Han, 2004] Dai, H. and Han, R. (2004). Tsync: a lightweight bidirectional time synchronization service for wireless sensor networks. *SIGMOBILE Mob. Comput. Commun. Rev.*, 8:125–139.

[Dakkak et al., 2011] Dakkak, M., Nakib, A., Daachi, B., Siarry, P., and Lemoine, J. (2011). Indoor localization method based on RTT and AoA using coordinates clustering. *Computer Networks*, 55(8):1794–1803.

[Di et al., 2000] Di, X., Gosh, B. K., and Tarn, T. (2000). Sensor-based hybrid position/force control of a robot manipulator in an uncalibrated environment. *IEEE Transactions on Control Systems Technology*, 8(4):635–645.

[Ding et al., 2005] Ding, M., Chen, D., Xing, K., and Cheng, X. (2005). Localized fault-tolerant event boundary detection in sensor networks. In *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, volume 2, pages 902 – 913 vol. 2.

[Elson and Estrin, 2001] Elson, J. and Estrin, D. (2001). Time synchronization for wireless sensor networks. In *International Parallel and Distributed Processing Symposium (IPDPS), Workshop on Parallel and Distributed Computing Issues in Wireless Networks and Mobile Computing*, San Francisco, CA, USA.

[Elson et al., 2002] Elson, J., Girod, L., and Estrin, D. (2002). Fine-grained network time synchronization using reference broadcasts. *SIGOPS Oper. Syst. Rev.*, 36:147–163.

[Fidge, 1988a] Fidge, C. J. (1988a). Partial orders for parallel debugging. *SIGPLAN Not.*, 24:183–194.

[Fidge, 1988b] Fidge, C. J. (1988b). Partial orders for parallel debugging. In *Proceedings of the 1988 ACM SIGPLAN and SIGOPS workshop on Parallel and distributed debugging*, PADD '88, pages 183–194, New York, NY, USA. ACM.

[FireBug, 2004] FireBug (2004). Firebug. from http://firebug.sourceforge.net/.

[GalsC,] GalsC. galsC: A language for event-driven embedded systems. from http://galsc.sourceforge.net/.

[Ganeriwal et al., 2003] Ganeriwal, S., Kumar, R., and Srivastava, M. (2003). Timing-sync protocol for sensor networks. In *1st International Conference on Embedded Networked Sensor Systems (SenSys)*, pages 138–149. ACM Press.

[Hagan et al., 1996] Hagan, M. T., Demuth, H. B., and Beale, M. (1996). *Neural Network Design*. PWS Publishing Company.

[Hamam et al., 2009] Hamam, Y., Djouani, K., and Oyedapo, O. (2009). A distributed nodes' localisation approach in wireless sensor networks. In *Mobile and Ubiquitous Systems: Networking Services, MobiQuitous, 2009. MobiQuitous '09. 6th Annual International*, pages 1 –5.

[He et al., 2003] He, T., Huang, C., Blum, B., Stankovic, J., and Abdelzaher, T. (2003). Range-free localization schemes for large scale sensor networks. In *Proceedings of the Ninth Annual International MobiCom Conference*. ACM.

[Heidemann et al., 2001] Heidemann, J., Silva, F., Intanagonwiwat, C., Govindan, R., Estrin, D., and Ganesan, D. (2001). Building efficient wireless sensor networks with low-level naming. In *18th ACM Symposium on Operating Systems Principles*, pages 146–159. ACM Press.

[Hightower et al., 2000] Hightower, J., Want, R., and Borriello, G. (2000). SpotON: An indoor 3D location sensing technology based on RF signal strength. Technical report.

[Hill et al., 2000] Hill, J., Szewczyk, R., Woo, A., Hollar, S., Culler, D., and Pister, K. (2000). System architecture directions for network sensors. In *9th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS-IX)*, pages 93–104, Cambridge, Massachusetts.

[J. van Greunen and Rabaey, 2003] J. van Greunen and Rabaey, J. (2003). Lightweight time synchronization for sensor networks. In *2nd ACM International Conference on Wireless Sensor Networks and Applications*, pages 11–19. ACM Press.

[Jaikaeo et al., 2001] Jaikaeo, C., Srisathapornphat, C., and Shen, C. (2001). Diagnosis of sensor networks. In *International Conference on Communications*, pages 1627–1632.

[Katsura et al., 2003] Katsura, S., Matsumoto, Y., and Ohnishi, K. (2003). Analysis and experimental validation of force bandwidth for force control. In *IEEE Int. Conf. Control Syst. Technol.*, pages 796–8001.

[Khan et al., 2010a] Khan, S. A., Daachi, B., and Djouani, K. (2010a). Enhanced sensor localization through compensation of battery level decay. *Fifth International Conference on Broadband and Biomedical Communications 2010 (IB2Com)*, pages 1–5.

[Khan et al., 2010b] Khan, S. A., Daachi, B., and Djouani, K. (2010b). Enhanced sensor localization through compensation of battery level decay. In *Fifth International Conference on Broadband and Biomedical Communications 2010 (IB2Com)*, pages 1–5, Malaga, Spain.

[Khan et al., 2011] Khan, S. A., Daachi, B., and Djouani, K. (2011). Overcoming localization errors due to node power drooping in a wireless sensor network. *International Journal of Electronics and Telecommunications*, 57(3):341–346.

[Kim and Kwon, 2005] Kim, S. and Kwon, O. (2005). Location estimation based on edge weights in wireless sensor networks. *Journal of Korea Information and Communication Society*, 30(10A).

[Koushanfar et al., 2003] Koushanfar, F., Potkonjak, M., and Sangiovanni-Vincentelli, A. (2003). On-line fault detection of sensor measurements. *IEEE Sensors*, 2:974–980.

[Krishnamachari et al., 2002] Krishnamachari, B., Estrin, D., and Wicker, S. (2002). Impact of data aggregation in wireless sensor networks. In *International Workshop of Distributed Event Based Systems (DEBS)*.

[Krishnamachari and Iyengar, 2004] Krishnamachari, B. and Iyengar, S. (2004). Distributed bayesian algorithms for fault-tolerant event region detection in wireless sensor networks. *IEEE Transactions on Computers*, 53(3).

[Lamport, 1978] Lamport, L. (1978). Time, clocks, and the ordering of events in a distributed system. *ACM Communications*, 21:558–565.

[Lee and Choi, 2008a] Lee, M. and Choi, Y. (2008a). Fault detection of wireless sensor networks. *Computer Communications*, 31(14):3469–3475.

[Lee and Choi, 2008b] Lee, M. and Choi, Y. (2008b). Localized detection of faults in wireless sensor networks. In *ICACT*, pages 637–641.

[Leushen et al., 2002] Leushen, M., Cavallaro, J., and Walker, I. (2002). Robotic fault detection using nonlinear analytical redundancy. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 456–463.

[Locke et al., 2000] Locke, C., Carbone, G., Filippi, A., Sadler, E., Gerwig, B., and Evans, D. (2000). Using remote sensing and modeling to measure crop biophysical variability. In *5th International Conference on Precision Agriculture*.

[Luo et al., 2006] Luo, X., Dong, M., and Huang, Y. (2006). On distributed fault-tolerant detection in wireless sensor networks. *IEEE Transactions on Computers*, 55(1):58–70.

[Lysheyski, 2002] Lysheyski, S. (2002). Smart flight conrrol surfaces with micro-electromechanical systems. *IEEE Transactions on Aerospace and Electronic Systems*, 38(2):543–552.

[Mate, 2004] Mate (2004). Mate. from http://www.cs.berkeley.edu/pal/mate-web/.

[Mattern, 1989] Mattern, F. (1989). Virtual Time and Global States of Distributed Systems. In *Parallel and Distributed Algorithms: proceedings of the International Workshop on Parallel and Distributed Algorithms*.

[Meguerdichian et al., 2001] Meguerdichian, S., Slijepcevic, S., Karayan, V., and Potkonjak, M. (2001). Localized algorithms in wireless ad-hoc networks: Location discovery and sensor exposure. In *2nd ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pages 106–116. ACM Press.

[Mills, 1991] Mills, D. (1991). Internet time synchronization: the network time protocol. *IEEE Transactions on Communications*, 39(10):1482 –1493.

[Moustapha and Selmic, 2008] Moustapha, A. I. and Selmic, R. R. (2008). Wireless sensor network modeling using modified recurrent neural networks: Application to fault detection. *IEEE Transactions on Instrumentation and Measurement*, 57(5):981–988.

[Niculescu and Nath, 2001] Niculescu, D. and Nath, B. (2001). Ad hoc positioning system (aps). In *Global Telecommunications Conference, 2001. GLOBECOM '01. IEEE*, volume 5, pages 2926 –2931 vol.5.

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

[Niculescu and Nath, 2003b] Niculescu, D. and Nath, B. (2003b). Ad hoc positioning system (APS) using AoA. In *INFOCOM Twenty Second Annual Joint Conference of the IEEE Computer and Communications Societies*.

[Niculescu and Nath, 2003c] Niculescu, D. and Nath, B. (2003c). Localized positioning in ad-hoc networks. In *1st IEEE International Workshop on Sensor Network Protocols and Applications*, Anchorage, Alaska.

[Patil et al., 2005] Patil, M. M., Shaha, U., Desai, U., and Merchant, S. (2005). Localization in wireless sensor networks using three masters. In *IEEE International Conference on Personal Wireless Communications*, pages 384–388.

[PicoRadio, 2004] PicoRadio (2004). Picoradio. from http://bwrc.eecs.berkeley.edu/research/picoradio/.

[PODS, 2000] PODS (2000). Online at http://www.botany.hawaii.edu/pods/.

[Pouliezos and Stavrakankis, 1994] Pouliezos, A. and Stavrakankis, G. (1994). *Real Time Fault Monitoring of Industrial Processes*. Kluwer Academic Publishers, Norwell, MA.

[Priyantha et al., 2000] Priyantha, N. B., Chakraborty, A., and Balakrishnan, H. (2000). The cricket location-support system. In *Proceedings of the 6th annual international conference on Mobile computing and networking*, MobiCom '00, pages 32–43, Boston, Massachusetts, US. ACM.

[Priyantha et al., 2001] Priyantha, N. B., Miu, A. K., Balakrishnan, H., and Teller, S. (2001). The cricket compass for context-aware mobile applications. In *Proceedings of the 7th annual international conference on Mobile computing and networking*, MobiCom '01, pages 1–14, Rome, Italy. ACM Press.

[Rappaport et al., 1996] Rappaport, T., Reed, J., and Woerner, B. (1996). Position location using wireless communications on highways of the future. *IEEE Communications Magazine*, 34(10):33–41.

[Ratnasamy et al., 2003] Ratnasamy, S., Karp, B., Shenker, S., Estrin, D., Govindan, R., Yin, L., and Yu, F. (2003). Data-centric storage in sensornets with ght, a geographic hash table. *Mobile Networks and Applications*, 8(4):427–442.

[Ratnasamy et al., 2002] Ratnasamy, S., Karp, B., Yin, L., Yu, F., Estrin, D., Govindan, R., and Shenker, S. (2002). Ght: A geographic hash table for data-centric storage. In *1st ACM international workshop on Wireless sensor networks and applications*, WSNA '02, pages 78–87, Atlanta, Georgia, USA. ACM Press.

[Regalia and Wang, 2010] Regalia, P. and Wang, J. (2010). On distance reconstruction for sensor network localization. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 2866–2869.

[Ruiz et al., 2004] Ruiz, L., Siqueira, I., Oliveira, L., Wong, H., Nogueira, J., and Liureiro, A. (2004). Fault management in event-driven wireless sensor networks. In *MSWIM'04*.

[Savvides et al., 2001] Savvides, A., Han, C., and Srivastava, M. (2001). Dynamic fine-grained localization in ad-hoc networks of sensors. In *7th Annual International Conference on Mobile Computing and Networking*, pages 166–179. ACM Press.

[Savvides et al., 2002] Savvides, A., Park, H., and Srivastava, M. (2002). The bits and flops of the n-hop multilateration primitive for node localization problem. In *1st ACM International Workshop on Wireless Sensor Networks and Applications*, pages 112–121. ACM Press.

[Schwiebert et al., 2001] Schwiebert, L., Gupta, S. K., and Weinmann, J. (2001). Research challenges in wireless networks of biomedical sensors. In *7th Annual International Conference on Mobile Computing and Networking*, MobiCom '01, pages 151–165, Rome, Italy. ACM.

[S.C.Lee, 1994] S.C.Lee (1994). Sensor value validation based on systematic explorationo f the sensor redundancy for fault diagnosis. *IEEE Transactions on Systems, Man and Cybernetics*, 24(4):594–605.

[Seidel and Rappaport, 1992] Seidel, S. and Rappaport, T. (1992). 914 mhz path loss prediction models for indoor wireless communications in multifloored buildings. *Antennas and Propagation, IEEE Transactions on*, 40(2):207 –217.

[Shang et al., 2004] Shang, Y., Rumi, W., Zhang, Y., and Fromherz, M. (2004). Localization from connectivity in sensor networks. *Parallel and Distributed Systems, IEEE Transactions on*, 15(11):961–974.

[Sichitiu and Veerarittiphan, 2003] Sichitiu, M. and Veerarittiphan, C. (2003). Simple, accurate time synchronization for wireless sensor networks. In *IEEE Wireless Communications and Networking Conference (WCNC)*, volume 2, pages 1266–1273, New Orleans, Louisiana.

[Sivrikaya and Yener, 2004] Sivrikaya, F. and Yener, B. (2004). Time synchronization in sensor networks: A survey. *IEEE Network*, 18(4):45 – 50.

[Smith, 1998] Smith, A. (1998). *Radio Frequency Principles and Applications : The Generation, Propagation, and Reception of Signals and Noise*. Wiley-IEEE Press.

[Sudduth, 1999] Sudduth, K. A. (1999). Engineering technologies for precision farming. In *International Seminar on Agricultural Mechanization Technology for Precision Farming*, Suwon, Korea.

[Sugeno and Kang, 1986] Sugeno, M. and Kang, G. T. (1986). Fuzzy modelling and control of multilayer incinerator. *Fuzzy Sets and Systems*, 18(3):329 – 345.

[Takagi and Sugeno, 1985] Takagi, T. and Sugeno, M. (1985). Fuzzy identification of systems and its applications to modeling and control. *IEEE Transactions on Systems, Man, and Cybernetics*, 15(1):116–132.

[Tilak et al., 2002] Tilak, S., Abu-Ghazaleh, N., and Heinzelman, W. (2002). Infrastructure tradeoffs for sensor networks. In *1st ACM International Workshop on Wireless Sensor Networks and Applications*, pages 49–58. ACM Press.

[TinyDB,] TinyDB. TinyDB: A declarative database for sensor networks. from http://telegraph.cs.berkeley.edu/tinydb/.

[TinyGALS, 2004] TinyGALS (2004). TinyGALS: A programming model for event driven embedded systems. from http://ptolemy.eecs.berkeley.edu/papers/03/tinygals/.

[Townsend and Arms, 2004] Townsend, C. and Arms, S. (2004). Wireless sensor networks: Principles and applications. *MicroStrain, Inc*, pages 439–449.

[Wong et al., 2005] Wong, K., Tsang, I., Cheung, V., Chan, S., and Kwok, J. (2005). Position estimation for wireless sensor networks. In *IEEE Global Telecommunications Conference*, pages 2772–2776.

[xbow, 2004a] xbow (2004a). Onlie at http://www.xbow.com/products/productsdetails.aspx?sid=101.

[xbow, 2004b] xbow (2004b). Crossbow technology inc. from http://www.xbow.com.

[xbow, 2004c] xbow (2004c). *Crossbow Technology MPR2400 MICAz*. from http://www.xbow.com/products/product_pdf_files/wireless_pdf/6020-0060-01_a_micaz.pdf/.

[xbow, 2004d] xbow (2004d). *Crossbow Technology's MicaZ sensor mote*. from http://gyro.xbow.com/other/micaz_new.jpg.

[Ye et al., 2002] Ye, F., Luo, H., Cheng, J., Lu, S., and Zhang, L. (2002). A two-tier data dissemination model for large-scale wireless sensor networks. In *8th Annual*

International Conference on Mobile Computing and Networking, pages 148–159. ACM Press.

[Yu et al., 2007] Yu, M., Mokhtar, H., and Merabti, M. (2007). Fault management in wireless sensor networks. *IEEE Wireless Communications*, pages 13–19.

[Yun et al., 2007] Yun, S., Lee, J., Chung, W., and Kim, E. (2007). Centroid localization method in wireless sensor networks using TSK fuzzy modelling. In *Eigth International Symposium on Advanced Intelligent Systems*.

[Yun et al., 2009] Yun, S., Lee, J., Chung, W., Kim, E., and Kim, S. (2009). A soft computing approach to localization in wireless sensor networks. *Expert Systems with Applications*, 36(4):7552–7561.

[Zhirabok and Preobragenskaya, 1993] Zhirabok, A. and Preobragenskaya, O. (1993). Instrument fault detection in nonlinear dynamic systems. In *Syst. man Cybern.*, pages 114–119.