# Abstract

Localization of nodes in wireless sensor networks (WSNs) is important to context-aware and position-dependent applications; data are generally meaningless without a known location. Many algorithms exist for localizing nodes using RSS; however, a detailed quantitative comparison of these algorithms has not yet been published. In this paper we present such a quantitative comparison of algorithms which use RSS as a ranging method and present a localization software framework called Senseless. We also conducted a survey about the influence of the orientation of a node, thus the radiation pattern. Our study finds that the received power is not equal in all directions and that no single best algorithm for localization exists to date. Each algorithm has a different purpose and diverse properties. Learning from these algorithms and techniques, the correct algorithm can be chosen for the correct environment and a more advanced localization system is feasible. Using this framework, we implemented two centralized algorithms; Multilateration and MinMax.

# Introduction

A formal definition of Wireless Sensor Networks is given in [WSN: A survey]. The purpose of a WSN is to monitor some physical phenomena such as the ambient temperature, air humidity, and the presence or absence of a certain chemical. Usually, this data is collected at a common point, the data sink, where the data can be further processed or analyzed by the user. WSNs enable a great deal of new applications including environmental and habitat monitoring, smart homes and battlefield control.  
  
Accurate and low-cost sensor localization is an essential service and is important to many of these applications. The measured data is generally meaningless without knowing where it originated from. That being said, there are other reasons to acquire the location of a sensor. Location can be a goal in itself, such as in warehousing and manufacturing applications. Another added benefit is the possibility of geographical based routing.   
  
Given the popularity of GPS, one would think of it as a possible solution. However, this is considered a bad solution because of several reasons. Firstly, a great deal of WSN applications are built for indoor environments. Clearly, GPS does not suffice as it requires Line-of-Sight with at least four satellites. Secondly, GPS requires significant power to operate, which is a sparse resource in WSNs. Thirdly, GPS adds to the size and cost of a WSN node which should be kept as low as possible. This does not mean, however, that GPS is entirely out of the question. A small number of WSN nodes will need *a priori* knowledge about their location; this can be done by a GPS receiver or by manually mapping the node to a position on a map, or some other method. The number of these nodes should be kept as low as possible, in order to keep the deployment costs and time low.

Localization techniques /which are specific for WSNs are based on pair-wise measurements between nodes to estimate the positions. A small fraction of the network should have a known position as described in the previous paragraph. These nodes are called anchor nodes. The other kind of nodes, without a known position, are called blind nodes. Thus, the goal of a localization system is to determine the position of the blind nodes by communicating with the anchor nodes. We can divide these techniques into two categories: range-based and connectivity-based. Range-based methods estimate the distance between nodes with ranging method such as ToA, AoA and RSS. These techniques typically provide superior accuracy but are more complex than connectivity-based algorithms. These do not estimate the distance between nodes but determine the position of a blind node by their proximity to anchor nodes. [Weyn][HighTower][Wij] describe agreement properties of localization techniques in more detail.

There are other localization techniques which are used in other networks, the most common being RF fingerprinting. In this method, a map based on radio signals is created. In the first phase (offline-phase) we measure RSS on different points on our map and store them in a database. The second phase (real-time phase) measures the RSS of a blind node and compares it to reference points on the map constructed during the first phase, in order to determine the location of the blind node. The problem with this technique is that the RF pattern changes when the environment changes, which means that the radio map is no longer up to date and the accuracy will therefore drop dramatically. A second problem is that these maps are time-consuming to build. Other techniques exist, but they are far less common and unsuitable for WSNs.  These techniques will not be used in this paper.

Although many ranging techniques exist, in this paper we have narrowed them down to RSS-based ranging.

RSS-based ranging is founded on the principle that RSS attenuates with distance due to free-space losses and other factors. RSS is generally considered as a bad method because of its high variability due to interference, multipath and shading. Errors can be divided into two categories, environmental and device errors. Environmental errors are due to the wireless channel; these include multipath, shadowing effects and interference from other radio sources. Device errors are generally calibration errors. The most important consideration here is to keep the transmitted power constant, despite inter-device differences and depleting batteries. Environmental errors can also be divided into two parts: rapid time varying errors and static environment dependent errors. The first is due to movement of people, additive noise and interference. This can mostly be modeled as Gaussian noise. As a result, this can be reduced considerably by averaging multiple RSSI measurements [15]. The second type of error is due to the varying properties of the environment, such as multipath and shadowing. Since the layout of the environment and the placement of doors and furniture cannot be known without prior knowledge, this error should be modeled as random.

 In order To create an accurate localization system based on RSSI, the wireless channel properties and these other degrading effects must be modeled as accurately as possible. All these factors seem to give RSSI measurements a large random factor, thus making it very unpredictable. Even with very good modeling, it is inevitable that errors remain present because of the random factor; thus any good localization algorithm should also account for these factors. The upside of using RSS as a ranging method is that the radio can be used for communication and localization. This makes RSS very interesting because there is no need for additional hardware. Other ranging methods such as TOA, especially combined with ultrasound, usually yield better results, but require additional specialized hardware which adds to the size and cost of a node.

RSS-based localization can be divided into three categories:

* Range-based or fine-grained localization
* Connectivity-based or coarse-grained localization
* RF Fingerprinting, as described in the previous paragraph

We will restrict our algorithms to the first two types to satisfy the ad-hoc requirement of the network.   
  
As WSNs have some unique properties, the algorithms in this paper have been developed or selected with the following goals in mind:

* RSS-based : using this technique no additional hardware is required, thus the cost of a node can be kept low. However, because nodes have an antenna embedded on the PCB, it would be better to have an external antenna as these have a more uniform radiation pattern.
* Distributed and self-organizing: The algorithm should be able to run locally on the nodes to avoid a central processing dependency. This is especially important for WSNs due to the fact that individual nodes and links between nodes are more prone to failure than in a traditional computing environment. Batteries may be depleted and radio links can be obscured.
* Robust: The algorithms should account for localization errors and node failures.
* Receiver-based: The task of localization is up to the blind node so that the network scales well.
* Responsiveness: The localization latency needs to be kept as low as possible. Mobility is fairly limited in WSNs as most nodes have a static position; however, certain nodes can be mobile, so this factor needs to be accounted for as well.
* Energy usage: Given the sparse amount available, processing and communication needs to limited. Unfortunately, this means that certain applications are not suitable for WSNs because of the high computational requirements and the lightweight microcontroller that drives the nodes. Communication between nodes needs to be limited as well because the radio requires much more power than the microcontroller.
* Adaptive: We want our algorithm to be adaptive to the number of ANs and the density of the network. If the density or the number of ANs rises, the accuracy should improve. The algorithm should thus benefit from the high density of the WSN. The algorithm should still perform well with a low network density and AN ratio.
* Multihop localization: Nodes not in range of an AN should still be able to localize themselves by the use of other BNs, this is referred to as *cooperative or multihop localization* [Locating the nodes], compared to single-hop localization, where blind nodes in range of enough anchor nodes can determine their position.

The main contributions of this paper are:

* We present a detailed overview of the existing algorithms and literature.
* We compare two algorithms with quantitative measurements.
* We researched the influence of the orientation of a node compared to a node with an external antenna
* We present Senseless, a software framework to manage these WSN localization algorithms
* We interface this framework to SCALA, a localization middleware project.

The rest of the paper will be organized as follows: chapter two summarizes related work, and provides an overview of the suitable algorithms and work that contribute to building a good algorithm; chapter three presents the various algorithms that we have implemented and tested, and our software framework; in chapter four the results are presented and we conclude in chapter five.

# Related work

The amount of literature on this topic is quite substantial. It becomes more manageable If we limit ourselves to three categories: ranging algorithms, location estimation and frameworks. Limited surveys on this topic do exist; however, they fail to point out a superior algorithm and provide few quantitative comparisons, so further investigation is required.

1.

Sum Dist is a distributed multihop algorithm that makes use of the hop count as a primitive distance metric. The distance to the anchors is determined by simply adding the ranges (RSSI) encountered at each hop during the network flood. The downside of this algorithm is that the range errors rise exponentially when the beacon travels over multiple hops. In large networks with little anchor nodes, it will lead to poor ranging. A good alternative is DV-hop. It makes use of the topological information; counting the hops instead. When the topology of the network is very irregular, the ranging will be very inaccurate because of the high variance in hop distance. First the anchor nodes broadcast a beacon message to the blind node, which will forward the message to blind nodes that are out or range of the anchors after filling in the measured RSSI in the path length

The paper “RSS-based location estimation with unknow pathloss model” dynamically estimates the distance-power gradient; parameter of the radio propagation pathloss model. It adapts automatically to the environment, thus eliminating the need for extensive channel measurements. This leads to a more accurate conversion of RSSI to distance.

Sorted RSSI Quantization [3] is a connectivity based algorithm that uses hops and RSSI as a ranging method. It multiplies a hop by the radio range or a chosen distance. It sorts the obtained RSSI and applies a quantizer that represents a level of range in the hop. This makes the algorithm insensitive to RSSI errors.

2.

Centroid localization is a simple approach for coarse grained localization. All blind nodes calculate their position as the centroid of the anchor nodes within their communication range. This algorithm has a low accuracy because it does not use signal strength to denote the range. Many solutions exist to make CL more accurate like Weighted centroid localization (WCL)[Improved weighted centroid localization in smart ubiquitous environments] and Modifief Weighted Centroid Localization.

3.

MoteTrack is decentralized location tracking system based on RF. It is similar to RADAR, but does not rely upon back-end server or network infrastructure. The location of each blind node is computed using a RSS signature from the anchor nodes to a database of signatures. This database is stored at the anchor nodes themselves.

Cricket is decentralized and uses RF and ultrasound to determine the location of a blind node. Anchor nodes broadcast beacon messages. Together with that RF message it will transmit a ultrasonic pulse. Blind nodes listen to beacon messages and upon receipt, it will listen to the corresponding ultrasonic pulse. A distance to the transmitting anchor node can be estimated with that pulse.

A noteworthy survey is [K Langendoen.] by K. Langendoen. This survey describes three algorithms:

* Ad-hoc positioning by Niculescuand Nath [10],
* N-hop multilateration by Savvides et al. [12],
* Robust positioning by Savarese et al. [11].

 These algorithms are fully distributed algorithms; they require no central processing node and are designed towards multihop localization. The survey concludes that no single algorithm performs best under different circumstances. Robust positioning works best when no or very bad ranging information is available. Ad-Hoc positioning only works well when the ranging error is very low (<20%). The N-hop multilateration is to be preferred in other situations.

This survey identifies a common three-phase structure in these localization algorithms. The first phase is determines the distances between blind and anchor nodes. Note, however, that this does not mean that a specific ranging method, such as RSS, should be used,. The second phase derives a position using the anchor nodes. These two phases are roughly equal to what was described in the introduction. Finally, there is a third phase called the refinement phase, where the positions are refined through iterative measurements.

 Another comparison is given in [] by Zanca et.al. This paper compares four algorithms:

* Min-Max [10|11] by …
* Multilateration [] by..
* Maximum Likelihood [5|12] by
* ROCRSSI [13] by

A brief introduction to the radio channel is provided. The absolute ranging errors of the algorithms are presented with the number of anchor nodes as a parameter. The authors conclude that ML provides superior accuracy compared to the other algorithms when the number of anchor nodes is high enough. Interestingly, despite its simplicity, Min-Max achieves reasonable performance. This is probably due to the fact that it localizes the node in the center of the estimated area. The authors also note that a good radio channel model is required to obtain a relatively high accuracy. The algorithms presented in this paper are one-hop algorithms; they can only localize nodes in reach of enough anchor nodes.

# Framework

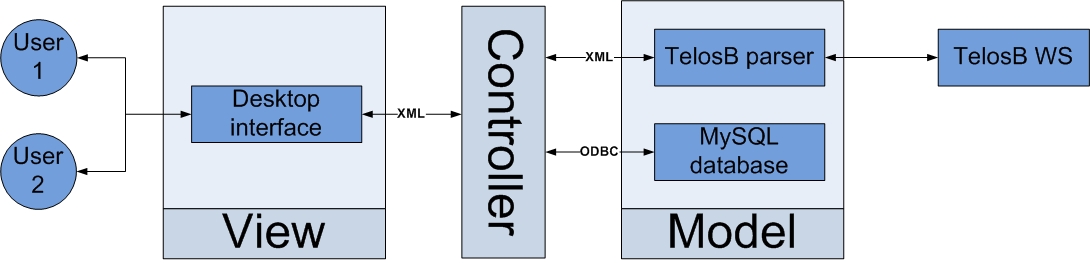
We have developed a software framework, Senseless, which provides a common data interface to the WSNs and GUIs. It also controls the data flows between the WSNs and GUIs, and stores this data in a database for later retrieval. The system is capable of working with different algorithms. If there are three anchor nodes available, we can work with range-based algorithms, thus obtaining a better accuracy. If, on the other hand, only one anchor node is available, a connectivity-based algorithm can and must be used. We will use this system to test the different localization algorithms and analyze the RSS data.

Senseless has a Model-View-Controller (MVC) design; the system is divided into three different parts each with different tasks. The separation of these responsibilities enhances the modularity of the system.

The details of this design are as follows:

* Model: This layer defines the representation of the information which the application works with. The data is stored in a MySQL database.
* View: Information can be accessed and controlled via this part. User interfaces are defined in this layer. The view does not process data.
* Controller: It processes and uses polling to react to events, mostly caused by the actions of the user and the data delivered by the WSN.

The advantage of this design pattern is that we can easily add views and models without changing the whole system.



## Functionality

The system consists of four main parts:

* WSN
* Database
* Graphical Unit Interface (GUI)
* Controller

### WSN

The WSN consists of telos rev.B nodes, which have the following specifications:

* TI MSP430 microcontroller with 10kB RAM
* IEEE 802.15.4 compliant CC2420 radio: it supports eight discrete power levels and 16 channels
* Integrated temperature, light, humidity and voltage sensor
* TinyOS 2.X compatible
* Programmable via USB interface
* Integrated antenna

Each node fulfills one of the three different roles:

* Root node (RN): this node receives data from the rest of the network , and acts as a bridge between the WSN and the rest of the framework. The root sends these messages to the controller via an XML parser. It also receives commands from the controller and disseminates these into the WSN
* Anchor node (AN): this node has a known location, and broadcasts a message with its ID to the blind nodes and anchor nodes for calibration. The node also transmits its sensor data to the root with the sensor message.
* Blind node (BN): this node has an unknown location and receives broadcast messages from the anchor nodes. The node uses these messages to determine the RSS, and transmits the RSS together with the ID of the anchor node to the root within the location message. The blind node also transmits its sensor data to the root with the sensor message.

There are three different data messages which are collected from the wireless network:

* Sensor messages, which contain the data collected by the sensors of the nodes
* Location messages, which contain data relevant to locate the blind node
* Status messages, which contain data that represent the status of the node transmitting the message

### database

We implemented a MySQL 5.0 database to store the data generated by the system, but any ODBC-compliant database can be used.

### gui

The user interfaces provides us with the ability to easily control and monitor the WSN. Rapid deployment was the key motivator to build this component. Using this, the user can set several node parameters, with a focus on localization. For example, the end-user can set the coordinates of an anchor node. This can be done in a few seconds. In contrast to manually hardcoding every single node of the network, which can be very time-consuming and sometimes impossible due to the fact that TinyOS-programmed nodes and other types of nodes usually provide very little to no user interaction.

### controller

The controller is the core of our system. The controller is programmed in C# using the .NET 3.5 framework. It is divided into several class library’s and a single Windows Forms project.

It has four main functions:

Firstly, the controller acts as a gatekeeper to the database, ensuring that all data is stored in the correct table and is of the correct type. This is especially important for WSNs as a plethora of hardware platforms exist. These platforms have different data types and can have a different endianness.

Secondly, the controller is also a central gathering point for all the data. By using the controller in our framework, every other component can use a single data interface and should only be aware of the location of the controller.

Thirdly, an interface to Scala is provided as well. Scala is middleware for location systems. Scala and the interface will be described in the designated sections

Finally, the controller implements the centralized versions of our localization algorithms. The user can instruct the controller to use an algorithm via a simple Windows Forms GUI

#### WSN vs Controller

The communication between the WSN and the controller is done with an XML parser, which translates the messages of both sides into XML and back into an internal format. The root node of the WSN receives all the messages (Sensor, location and status) from the nodes and forwards these to the controller, or if the controller needs to pass a command on to the root, it will be forwarded to the root. With the help of the dissemination protocol, the command is transmitted over the WSN.

#### Controller vs GUI

The communication between the controller and the GUI also happens with an XML parser, which translates the messages that need to be exchanged and back. Firstly, the GUI displays the data from the WSN, thus a request needs to be send to the controller. The controller receives the request, and gets the data out of the database and sends it to the GUI. Secondly, the GUI is used to control the WSN. If, for example, you want to change the sample rate of the WSN, then a command will be sent to the controller which forwards it to the root node of the WSN.

#### Controller vs databases

The communication here makes use of ODBC (Open Database Connectivity), which is an universal database interface. By using this interface, the controller does not need to worry about the database that is used. Stored procedures are used instead of full SQL syntax.

# Method

## propagation model

There exist a number of propagation models:

* the Rayleigh fading model:
* Rician distribution model: the determination of the model parameters is difficult
* Floor attenuation factor propagation model
* log-normal-shadowing model

We use the latter as our propagation model. It is the most widely used signal propagation model and the determination of the parameters is simple.

RSSI(d) = PT – PL(d0) − 10η log10 ( d/d0 )+ Xσ

Where,

PT = Transmit Power

PL(d0) = Path loss for a reference distance d0

d0 = Reference distance

η = Path loss exponent (the rate at which the path loss increases with distance)

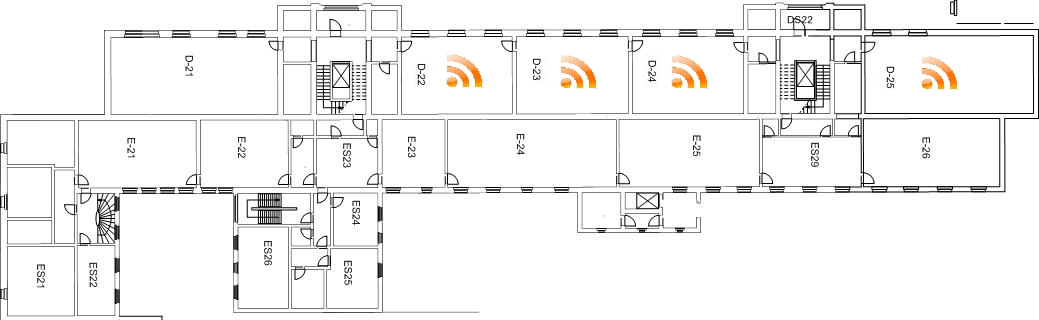
Xσ = a gaussian random variable with zero mean (Gaussian viariate) and standard deviation σ dB.

The most common reference distance is one meter. In the calibration phase of the WSN, we determine the path loss exponent, so that the propagation model is adapted to the environment.

## set up

Connectivity-based algorithms use the following set-up :

We have built a network with seven nodes on the floor of a building with blueprint, Figure X: one node acts as the root node (RN) and is connected to a computer, the second node acts as the blind node, thus with an unknown location, and the other four nodes are anchor nodes spread over four different rooms. We place the blind node at a known location in a room and measure the RSSI of the anchor nodes that are in range. The blind node transmits these data to the controller and saves the data to the database. The data will be used to apply different proximity algorithms and to measure the median location error.



Text verder onder figuur

Range-based algorithms make use of the following setup:

We have a network with five nodes in a room, Figure Y: one root node, connected to a computer, one blind node and three anchor nodes. The blind node is again placed at a known location in the room and measures the RSSI of the anchor nodes. This data is saved in the database.



## test: orientation

Two Telos rev. B nodes were set in an obstacle-free environment (basketball court).

In the first scenario the two nodes are equipped with an external antenna with a gain of 6dBi and placed at a distance of one meter from each other and with a height of one meter. This way the ground will not absorb or weaken the signal. One node is set as an anchor node and will broadcast beacon messages at a rate of 200 ms. The other node is the blind node and sends RSSI to the database.

In the next scenario, the nodes are placed at a distance five meters from each other.

In the following test, only one node is equipped with the external antenna and place at a distance of one meter. This node is set as the blind node and receives the same power in every direction. The other node with an integrated antenna is the anchor node that will broadcast.

In the last scenario the nodes are place at a distance of five meters from each other.

## test: positioning in openair

We placed a total of 10 nodes in an obstacle-free environment. Nine of them are configured as anchor node and the last one as a blind node. All the nodes are placed at a height of one meter.

The anchors are placed randomly at fixed locations (in meters):

* Node one: 0 , 1
* Node two: 1 , 1
* Node three : 2 , 1
* Node four: 3 , 1
* Node five: 4 , 1
* Node six: 2 , 2
* Node seven: 2 , 4
* Node eight: 3 , 3
* Node nine: 0.00 ; 0.00

The blind node (10) will be located at the following locations (in meters):

1. 0 , 2
2. 1 , 3
3. 2 , 3
4. 1 , 0
5. 0, 0
6. 1 , 1
7. 2 , 2

In the first scenario 3 anchors are active: node one, node two and node three. The blind node will be placed at the 7 different positions with an interval of five minutes.

In the second scenario two more anchor nodes are active: node four and node five.

In the third scenario two more anchor nodes are active: node six and node seven.

In the last scenario two more anchor nodes are for a total of nine: node eight and node nine.

We will repeat this test for the scenario where we expand the scale of the previous scenario with a factor of three, so each coordinate is multiplied with three.

## test: positioning indoor

We repeat the previous test in an indoor environment.

The anchors are placed randomly at fixed locations (in meters):

* Node one: 1.19 ; 6.98
* Node two: 2.00 ; 8.48
* Node three : 3.00 ; 1.50
* Node four: 3.19 ; 6.23
* Node five: 1.19 ; 5.14
* Node six: 4.64 ; 3.88
* Node seven: 4.67 ; 0.00
* Node eight: 2.50 ; 0.00
* Node nine: 0.00 ; 0.00

The blind node (10) will be located at the following locations (in meters):

1. 5.07 ; 8. 48
2. 5.07; 4.63
3. 2.00 ; 1.50
4. 0.00 ; 3.29
5. 3.19 , 6.23
6. 1.19 , 8.48

In the first scenario 3 anchors are active: node one, node two and node three. The blind node will be placed at the 7 different positions with an interval of five minutes.

In the second scenario two more anchor nodes are active: node four and node five.

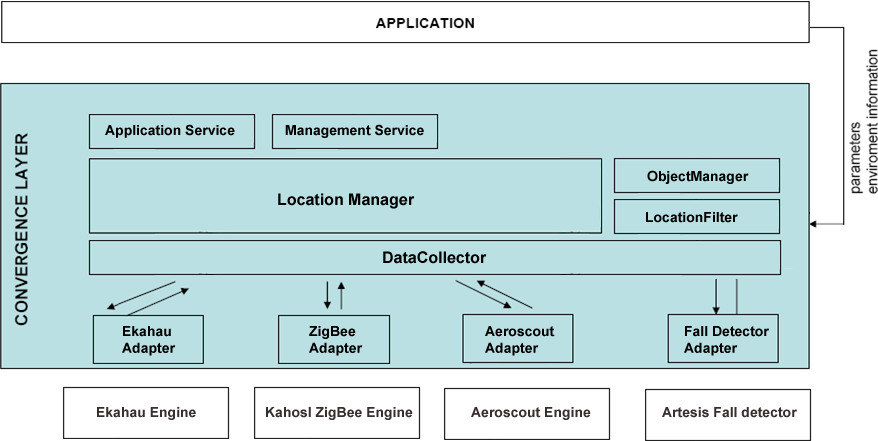
In the third scenario two more anchor nodes are active: node six and node seven.

In the last scenario two more anchor nodes are for a total of nine: node eight and node nine.

## scala

SCALA is a TETRA [http://www.iwt.be/steun/steunpro/tetra/index.html] project which aims to shorten the gap between localization technologies and possible end-user applications. The goal of this project is to ease the development of location aware applications by providing these applications with a common interface to the location technologies.  
  
From a technical point of view SCALA adapts the interfaces of the existing localization technologies into a common interface. Doing so, the user can receive location information transparent of the underlying technology.

Another feature of SCALA is the fusion of location information.. By combining the information the middleware receives from different localization technologies a more accurate and robust position can be determined.  
  
  
The Senseless framework will provide an interface to SCALA. The framework will plug into SCALA as an engine, which can be seen in the bottom layer of Figure X. Doing so, our algorithm becomes accessible to a variety of applications.



The interface with the middleware is constructed in the controller. Two types of communication are available: polling and event-based. By polling the engine, information Is requested only when needed, as opposed to event-based communication where, the middleware subscribes to information coming from the WSN. The interface is loosely based on the ANSI Rtls API [Ref].

The API documents a polling-based system. Possible data fields are specified and a method of filtering data is documented as well.

# Antenna orientation

The relative antenna orientation between receiver-transmitter pairs is a major factor in signal strength variability, even in the absence of multipath and shading effects. This is due to the fact that antenna’s are not perfectly omnidirectional. Ideally, the radiation pattern of an antenna should be uniform and it should look like a circle (2-D space) or a sphere (3-D space). However, in practice, the orientation of the antenna can influence the signal strength by several decibels. Thus, different antenna orientations can produce different sets of RSSI values for the same distances between receiver and transmitter.

Therefore, it is imperative that the antenna’s orientation should be accounted for as well. One possible solution would to use a compass to determine the nodes orientation. Given the antennas radiation pattern the transmitted power can easily be obtained.

A straightforward solution would be to use a more omnidirectional antenna to minimize these effects. For example, TelosB nodes use an onboard antenna. The power received from this antenna can differ by as much as 20dB depending on the orientation. Fortunately an external antenna is supported. This can be mounted on the circuit board via an optional standard SMA connector.

//Uitleg waarom

The influence of the onboard antenna as compared to the external antenna

Two nodes were put at a constant distance in an outdoor environment, a basketball court. One node served as the anchor node and continuously broadcasted messages to the other node. RSS samples were taken at this node. The anchor node was rotated and samples were taken at every 20 degrees. Approximately 25 samples were taken at every orientation.

Figure X plots the results of our test. Averaged RSS values and standard deviation are plotted for every sampled orientation. Table X displays the total standard deviation. These results show that RSS readings are generally more stable when the node is equipped with an external antenna. However the readings with onboard antenna are fairly stable as well.

The results also clearly show that the external antenna is more invariant to orientation than the onboard antenna.

# Algorithms

MinMax is a popular and a very easy algorithm to implement. Anchor nodes, that are in range of the blind nodes, will create a box around them. This box has the anchor node as his center and has a height and width of twice the estimated distance to the blind node. In an ideal situation, will this algorithms work, but the estimated distance between the node is often underrated. So, one or more boxes will not collide and thus a location can not be determined. In this case, the estimated distance between the nodes is expanded with 10%.

Trilateration is used in a variety of localization systems including GPS. Trilateration calculates the intersection of three or more circles. If these circles intersect in exactly one point, the coordinates of this point can be determined by linear equations. The problem with this method is that the circles almost never intersect in a single point. This is due to the erroneous ranging. The range is over or underestimated. This results in circles that do not intersect at all or overlap a part of each other. In more dramatic cases one circle can entirely overlap another circle.

[2] presents two solutions to solve this problem. The first solution is based on non-linear least squares. Through an iterative process the range of the circles is changed until a single point of intersection is found. This method can yield fairly accurate results. The downside however is that it is computationally intensive. The second solution is much simpler and requires that the circles overlap. Given x circles, the algorithm determines the x-nearest points. The centroid of these points is taken as the result.

The second solution is simpler to implement but has the requirement that the three circles should overlap a part of each other. We present a simple solution to this problem. There are three cases in which the range of the circle should be modified:

* The circles are too far from each other and do not intersect
* One circle completely overlaps another circle
* A circle is completely inside another circle

In each of these cases the range should be adapted accordingly until these conditions are no longer met. The algorithm iterates through all the circles so that all circles are slightly changed instead of drastically changing one circle. The algorithm finally converges to the situation where all circles overlap each other.

# results

The tests still need to be executed.

# Conclusion

This paper has investigated the core aspects of WSN localization systems using RSS. Different types of localization systems were introduced and important properties were discussed as well. An overview of our localization framework was given. Three centralized localization systems were discussed

# References

Will be added.