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 localtion =S.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 a localization software framework called Seneseless =S. Our study finds 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 several algorithms in TinyOS on the TelosB platform.

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 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 (geographical-based or geographically based?) routing.

Given the popularity of GPS, one would think of ~~GPS~~ it as a possible solution. However, this is =G: tense (would/could be) / is considered a bad solution because of several reasons. Firstly, a great deal of WSN applications are suited =V (designed)/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 =ST, 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 its =R 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 amount =G: count N /number of these nodes should be kept as low as possible, in order to keep the deployment costs and time low.

Localization techniques M: relative clause /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 A 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 =S algorithms. ~~Connectivity-based algorithms~~ These do not estimate the distance between nodes but determine the position of a blind node by his =R (their) proximity to anchor nodes. [Weyn][HighTower][Wij] describes =G: subject-verb 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 M?changes when the environment changes, which means that the radio map is notno longer up to date anymore =ST: informal (use "no longer") and the (level of?)(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 narrow(ed) =G: tense M 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 dependant=S 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 make =V , construct, build) 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 Athe 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 are (=have been?) developed or selected with the following goals in mind:

- RSS-based =S: using this technique no additional hardware is required, thus the cost of a node can be kept low. However, M: linking word(since) because/while~~???~~most 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) amount =G: count N of ANs and the density of the network. If the density or the amount/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)A existing algorithms and literature.
- We compare several existing algorithms with quantitative measurements.
- We implement several algorithms in TinyOS on the TelosB platform.
- 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 suitable algorithms and work that contributes (do you mean "… of the suitable algorithms and work that contribute…", i.e. both the algorithms and work contribute to the building of a good algorithm; or "… of suitable algorithms and of the work that contributes to…", meaning only the work contributes to the building of a good algorithm?) 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

Manually assigning the location of a node is a time--consuming =S and expensive job, and therefore a self-configuring method where nodes discover their neighbors and use them to estimate the distance to their neighbors.

Sorted RSSI Quantization (SRangeQ) is a distributed algorithm and is insensitive to RSSI error on range estimation. It makes use of blind nodes that are in range of the anchor node and thus can locate his =R its position, to find his =R its own location.

The amount of literature on this topic is quite substantial. If we limit ourselves to algorithms which use RSS, the topic becomes more manageable. 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.

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, ~~such as RSS =SS~~. 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 relative =G: adj. vs. adv. 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

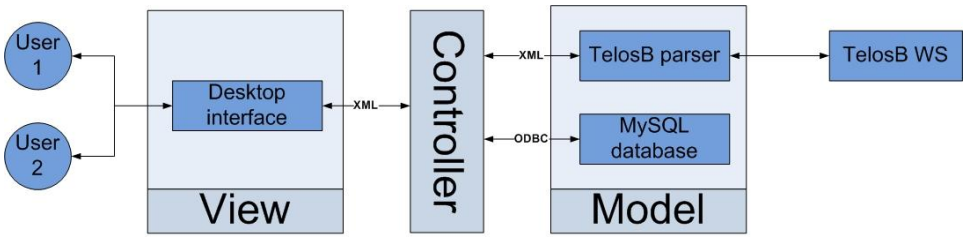
We developed =G: tensehave developed a software framework, NodeLoc, which provides a common data interface to the WSNs and GUIs. It also controls the dataflows between the WSNs and GUIs, and stores this data in a database for later retrieval. The system is capable of working with different algorithms. 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 in =Pinto? 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/include?): are as follows:

- Model: This layer defines the representation of the information which (=that) the application works with. The data is stored in a MySQL database.
- View: Information can be accessed and controlled through ~~(the)~~ /via? (This part) View. User interfaces are defined in this layer. The view does not process =S the data.
- Controller: It processes and uses polling to react on =P 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 exists out of =Vconsists of four main parts:

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

WSN

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

- TI MSP430 microcontroller with 10kB RAM
- IEEE 802.15.4 compliant radio
- 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~~. This node~~ , and acts as a bridge between the WSN and the framework. The root sends these messages to the controller via an XML parser.
- Anchor node (AN): this node has a known location, and broadcasts a message with his =R its position and ID to the blind nodes. The node also transmits hisits sensor data to the root with the sensor message.
- Blind node (BN): this node has an unknown location and receives =G: subject-verb agreement broadcast messages from the anchor nodes. The node uses this message these messages=G: sg. vs. pl. to determine the RSS, and transmits the RSS together with the position and ID of the anchor (~~node~~?)

to the root within the location message. The blind node also transmits his =Rits sensor data to the root with the sensor message.

There are 3 =ST 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

We built a GUI in .NET. This GUI is capable of presenting all the data in the database and controlling parameters of the wireless network, like =ST Asuch as the sampling period, node ID's =S IDs and radio power levels. Using this GUI we can easily control the entire network.

CONTROLLER

The controller is the core of our system. Its task is to calculate the position of a blind node and route all communication between the other parts of the framework. XML over TCP/IP is used to transport the data.

WSN VS CONTROLLER

The communication between the WSN and the controller happens occurs by? / Is performed / is done / works 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 Vice-versa: if the controller needs to pass a command on to the root, it will be forwarded to the root. With the help of Dissemeniation, the command is transmitted over the WSN.

CONTROLLER VS GUI

The communication between the controller and the GUI also happens with/ is achieved by =V (e.g. occur, take place by means of) 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 this =ST: relative clause 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

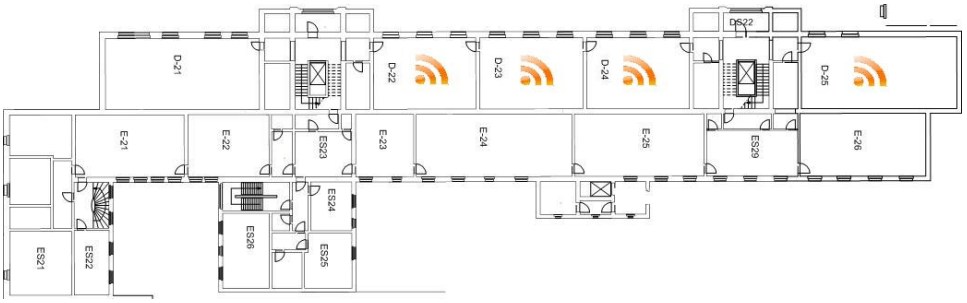
Connectivity-based algorithms use the following ~~set~~ set-up =S:

We built =G: tensehave **built** a network with 7 =ST seven nodes on the floor of a building with ~~the following~~ ???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 Aan 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 mote (=node?) node transmits this =R these?? data to the controller and saves it =R?? to the database. This =R data will be used to apply different proximity algorithms ~~to~~ and to measure the median location error. Data = always plural???

Formatted: Font: Bold

Formatted: Font: Bold

Formatted: Font: Bold



Text verder onder figuur

Range-based algorithms make use of the following setup:

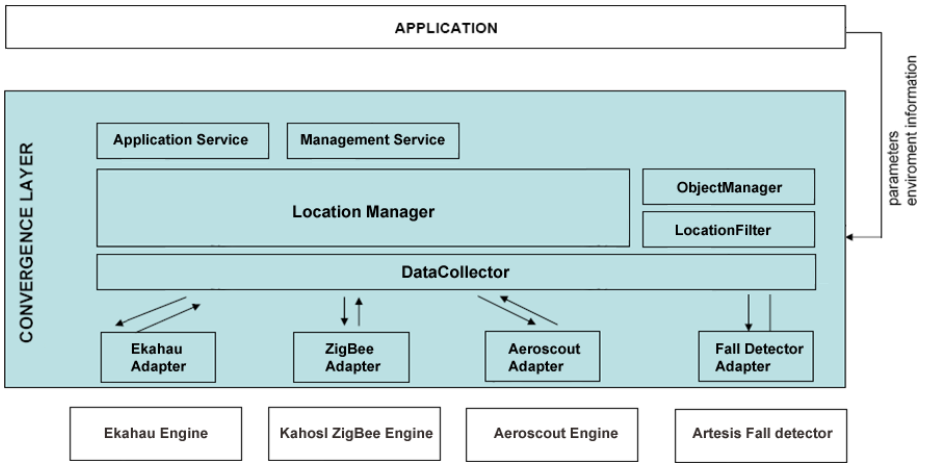
We have a network with 5 =ST fivenodes in a room, Figure Y: one root node, connected to a computer, one blind node and three anchor nodes. The blind node is again placed at a known location in the room and measures the RSSI of the anchor nodes. This =R data is =G: subject-verb agreement saved in the database.

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 software interfaces of the existing localization technologies into a common interface. Pin ~~Doing~~ doing so, the user receives location information transparent of the underlying technology.

Another feature of SCALA is the fusion of location information. The middleware will try to evaluate the information that it receives to improve the localization accuracy and robustness. The middleware will also allow the user to control various parameters of the localization technologies.



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. P in Doing so, our algorithm becomes accessible to a variety of applications.

RESULTS

The tests still need to be executed.

CONCLUSION

TODO

REFERENCES

TODO