A Distributed Precision Based Localization Algorithm for Ad-Hoc Networks

Leon Evers[1], Stefan Dulman[1,2], and Paul Havinga1,2

[1] EEMCS Faculty, University of Twente, the Netherlands
[2] Ambient Systems, the Netherlands
{eversl,dulman,havinga}cs.utwente.nl

Abstract. In this paper we introduce a new distributed algorithm for location discovery. It can be used in wireless ad-hoc sensor networks that are equipped with means of measuring the distances between the nodes (like the intensity of the received signal strength). The algorithm takes the reliability of measurements into account during calculation of the nodes positions. Simulation results are presented, showing the algorithms performance in relation to its accuracy, communication and calculation costs. The simulation results of our approach yield 2 to 4 times better results in position accuracy than other systems described previously. This level of performance can be reached using only few broadcast messages with small and constant size, for each node in the network.

1 Introduction

Recent advances in digital radio communication technology, combined with the continuous increasing battery capacities and ongoing miniaturization and integration of digital circuitry, have opened up a set of new application areas for digital computing devices of various scale and purpose. The effects of cheap ubiquitous connectivity that GSM phones offer are already changing the world today. New applications are at the horizon, like the use of cheap, tiny, wireless connected, digital sensors that will be able to continuously monitor their environment, and communicate their findings in real time to whomever is interested. Large-scale use of these sensors, wireless connected in randomly deployed ah-hoc networks covering complete areas will have many uses, whether it be biologists monitoring live stock, farmers keeping track of their cattle, soldiers monitoring the battlefield, the employment of forest fire warning systems, and many others.

In this paper a distributed algorithm for location discovery is presented. It can be used in wireless ad-hoc sensor networks that are equipped with means of measuring the distances between the nodes (like the intensity of the received signal strength). The algorithm takes the reliability of measurements into account during calculation of the nodes positions. Simulation results are presented, showing the algorithms performance in relation to its accuracy, communication and calculation costs.

A. Ferscha and F. Mattern (Eds.): PERVASIVE 2004, LNCS 3001, pp. 269–286, 2004.

1.1 Related Work

These kinds of applications require a specific kind of devices, with a specific set of requirements and capabilities, as well as limitations, as is described by Rentala et al. in their survey on sensor networks [10]. Especially the sheer number, in which deployment is desired, defines a set of limitations, which make the design of these devices a particularly challenging one. Low cost and maintenance-free use are among the most important properties, and restrict the technology to make and operate them. So a strong focus to minimal need for hardware, energy preservation, cooperation and autonomy are necessary in every step of the design process.

One of the many research issues is the problem of location discovery. This issue is an important one and has already had some attention in the past. In [5] Hightower and Borriello present a comparative description on existing systems. Unfortunately, none of the described systems is designed for the specific application targeted in this paper, and thus does not provide a good solution for positioning in ad hoc sensor networks. The Radar system ([2]), specifically designed for indoor use, depends on a central computer, performing the calculations, and requires a lot of manual calibration. The work described in [4], [15], [7] and [13] as well depends on a centrally performed network-wide calculation, and have strict requirements for the infrastructure set-up and/or node placement.

On the other hand, several systems have been designed with similar constraints on hardware and network configuration in mind. Three such systems are described in [12], [6] and [11]. In [12], similar algorithms are described, but an ultrasound ranging method is used, which is a more accurate distance measurement system, and doesn't deal with some of the problems that less accurate measurement systems pose, like RSSI used in this research. The systems described in [6] and [11] on the other hand, are based on similar ranging techniques, and hardware requirements. These systems, however, because of their rather low accuracy and coverage, might not be very useful for implementation in many application areas. Their results, however, can serve as a comparison to the solution proposed in this paper, which is done in Section 3.

The research described in this paper is a continuation of the work described in [8]. The theoretical results of the calculations involved in the IQL algorithm, as described in [8] have been tested inside a simulation during this research, and some improvements to it have already been made.

1.2 Paper Outline

In this paper a solution for localization in ad hoc wireless networks is proposed. We begin by describing the limitations and requirements of such a system in Section 2. The general approach of the proposed system will be explained in Section 3, and a more detailed description of the calculations involved is given in Section 4.

With the described system, a simulation environment has been implemented, to perform test on the performance. Section 5 gives a description of the details

involving the implementation, used to perform the tests, which are discussed in Section 6. Finally, a discussion on the performance in relation to other systems is presented in Section 7.

2 Preliminaries

The specific platform that our proposed system is intended for is that of a sensor network that can be deployed at random, consisting of a (possibly very large) number of similar sensor nodes, and only a very small amount of base stations or some other kind of (manually configured) fixed infrastructure. The sensor nodes will be small, cheap and battery operated, with short range radio frequency (RF) communication hardware, simple (slow) microprocessors, and additional sensing hardware.

This node is equipped with an 8 MHz 16 bits microcontroller (with 4 KB of data memory and 60 KB of program memory) and a radio transceiver with Received Signal Strength Indication (RSSI) capabilities.

Due to the nature of these nodes, some important design goals for the algorithms used to provide the localization data have been identified, much like the algorithms described and compared in Langendoen et al. [6]. Such algorithms should be truly distributed, as well as self-organizing, robust and energy-efficient. This means that the calculations should be made on each individual node, using as little as possible computation and especially communication, without relying on any fixed infrastructure or network topology, and being able to cope with changing conditions, such as node failures.

2.1 Environment

The kind of environment in which these algorithms have to work will be one where a large amount of randomly placed nodes forms a network using the short range RF transceiver to broadcast messages to only a small subset of the nodes in close enough vicinity to be able to receive those messages. Nodes that are able to directly communicate with each other will be called neighbors throughout the rest of this paper. The amount of neighbor nodes that each node can reach, called the connectivity from here on, can be variable by changing the transmit power of each node. A constant power level has been assumed in this research though.

A small subset of the nodes in the network, called anchor nodes, will initially be equipped with its own location, expressed by its coordinates relative to some network-wide coordinate system, either by manual configuration, or by using other location sensing techniques requiring extra hardware, like for example GPS. All other nodes will initially not know their location. Anchor nodes are assumed to have the same hardware capabilities, so factors like communication capabilities and energy consumption considerations will be equal to non-anchor nodes. Ideally, these nodes should be spatially distributed equal across the network, even though in certain application areas this might not be the case, or

cannot be relied upon. Certain flexibility towards this property has to be assured by the localization algorithm. To minimize on installation and maintenance effort the fraction of anchor nodes in the network should be really small, and the location algorithm should be able to deal with this small amount of anchor nodes.

The results presented in this paper only focus on situations with fixed sensor locations, since this already proves to be enough of a challenge. Other environmental factors, like the positions of objects in between the nodes, might be changing, however, resulting in varying readings of RSSI measurements between pairs of nodes at different times. Systems where the nodes are mobile can use the described algorithms to update the nodes positions continuously. However, this has not been implemented yet and is the object of our future work. The presented results can be easily adapted and can be used as a guide when designing a mobile system though.

3 Solution Outline

As the basis of an algorithm to determine a node's location, some measurements have to be available. With the hardware and network structure described before, the received signal strength indication can be used to obtain an indication about the distance between a pair of neighbor nodes in the network. The specific calculations to translate the RSSI readings into the distance towards the sending node will not be addressed in this paper. It is assumed that such a calculation can be constructed and is available to the algorithm presented herein.

3.1 Precision of Measurements

If the distance to at least 3 neighbor nodes is known, as well as the locations of those nodes (for example because they are anchor nodes), the position in 2-dimensional space can be computed using triangulation calculations. Unfortunately, especially in indoors environments, the distances obtained from the measured RSSI will be quite imprecise, because of the fading and multi-path effects of the radio signals meeting with the various surfaces in the surroundings of the node. According to [3], this error can be as large as 50% of the measured distance. The error of the measurement does however conform to a Gaussian distributed random variable. As a descriptive value of the error distribution the standard deviation can be used. For maximum errors of 50% this means standard deviation of about 20% of the distance. This value will be referred to further on as η. Different environments with other error characteristics will result in different (possibly smaller) values of η.

By itself, with distance errors of this size, the computation through triangulation will contain a large error as well, especially because of the accumulation of the errors in subsequent calculations. This might render the result of the calculation practically useless. However, by using the connectivity of the network,

which is usually more than 3 neighbors per node, the redundancy in the network can be exploited to improve on the results of estimating a nodes location. Other factors, such as the known properties of the error distributions, as well as obtaining multiple measurements between pairs of nodes instead of just one, can be taken into account as well to try to obtain a reasonable precision of the computed location.

3.2 Iterative Multilateration

For nodes with more than the minimally required 3 neighbors with known position, the "iterative multilateration" method, described by Savvides et al. [1] can be used to calculate the position with smallest error. Because of the known error distribution of the distance measurements, the error of the obtained location can be calculated as well. This can be modelled as a Gaussian distributed random variable, denoting the probability of the real location of the node to be within a certain range of the computed value, expressed with the standard deviation of the error distribution. In other words, the standard deviation serves as a measure of the precision of the location estimation.

This newly calculated position, combined with its precision, can now be used in subsequent calculations for other still undetermined nodes. This is done by combining the precision value of the newly located node with the error of the measured range between this node and its neighbor to obtain a total precision value on the distance between the node's real location and its neighbor. When a still undetermined node obtained at least three range measurements to already determined neighbor nodes, it can itself calculate its position. The precise calculations involved in this process will be described in detail in Section 4. This iterative multilateration can be used again to refine a node's position, when more neighbor nodes have their position calculated, or after the neighbor nodes have obtained refined position estimates.

This approach of refining a node's position based on the measured ranges to neighbor nodes is used as well in the refinement stage of the system described in [12], but because a precision value of a node's initial position is known as well in the scenario described above, initial position estimates of non-anchor nodes can be used to calculate a position estimate of undetermined nodes as well. When a possibly very imprecise (with large standard deviation value) location estimate is used in subsequent estimation calculations, the large error is accumulated in the results of the new calculation. Though, this result is provided together with an even larger standard deviation of the position error meaning that the lower accuracy is provided with the result of calculation.

3.3 Multi-hop Ranging

The problem that still arises is that at the start of the location discovery algorithm, only the anchor nodes will have a known location (with infinitely high precision). Because of the small fraction of anchor nodes, in many cases there

will be no non-anchor node with at least three anchor nodes as its direct neighbors, allowing it to calculate its own position. To almost all non-anchor nodes the anchors are several hops away, and no direct range measurements can be obtained. In such situations no positioning can be achieved. It is possible though, to obtain a distance to the anchor nodes from all non-anchor nodes, by using the distances measured at the intermediate hops on the shortest path to each anchor. The multi-hop distance can be obtained by multiplying the sum of all single hop measurements by a precomputed bias factor. The standard deviation that belongs to the multi-hop distance is computed in the same way. In Section 4.2 the computation of these bias factors is explained.

4 Calculations

4.1 Iterative Weight Least Squares Estimation

The calculations of a nodes position from a set of range measurements between the node with unknown position and a set of nodes with known or estimated positions can be performed in a way similar to that of GPS [9] and the "iterative multilateration" described in [1]. The main difference with those systems and the one proposed in this paper is that the precision estimates of all data values, expressed as a standard deviation of the error distribution, are also taken into account in the whole calculation. This section deals with the description of the *Iterative Weight Least Squares Estimation* used in the proposed system.

Starting from an initial estimation, an improvement vector is calculated iteratively and added to the previous estimation until the improvement vector is smaller than a certain value. This vector is obtained through a weighted least squares estimation:

$$wA\mathbf{x} = w\mathbf{b} \tag{1}$$

where w is a weight factor, A is a matrix, and $\mathbf{x}$ and $\mathbf{b}$ are vectors.

To calculate the improvement vector we take the true position of the node to be calculated as $\mathbf{x} = (x, y)$ for a 2-dimensional system, or $\mathbf{x} = (x, y, z)$ in the 3-dimensional case. The initial estimation of the position is denoted as $\mathbf{x}^{est}$ and the positions of the n neighbor nodes as $\mathbf{x}_i$, for $i = 1..n$. The measured ranges to those nodes can be denoted as: $r_i = \|\mathbf{x}_i - \mathbf{x}\| + \epsilon_i$, $i = 1..n$, with ϵ_i denoting the measurement error, and the node's true position as: $\mathbf{x} = \mathbf{x}^{est} + \delta\mathbf{x}$.

In the same way, the distances to the estimated position are: $r^{est} = \|\mathbf{x}_i - \mathbf{x}^{est}\|$, $i = 1..n$ and

$$\delta r_i = r_i - r_i^{est} = \|\mathbf{x}_i - \mathbf{x}^{est} - \delta\mathbf{x}\| - \|\mathbf{x}_i - \mathbf{x}^{est}\| + \epsilon_i$$

$$\delta r_i \approx -\frac{(\mathbf{x}_i - \mathbf{x}^{est})}{\|\mathbf{x}_i - \mathbf{x}^{est}\|} \cdot \delta\mathbf{x} + \epsilon_i = \mathbf{1}_i \cdot \delta\mathbf{x} + \epsilon_i,\ i = 1..n \tag{2}$$

with $\mathbf{1}_i$ the direction vector of length 1 from the nodes estimated position to node i.

For each range measurement r_i a weight $w_i = 1/\sqrt{\sigma_i^{edge2} + \sigma_i^{node2}}$ is calculated, where σ_i^{edge2} is the variance of the range measurement to node i, and σ_i^{node2} is the variance of the position of the node i.

Matrix A consists of n rows, each filled with the direction vector $\mathbf{1}_i$, one for each anchor node involved in the calculation. Vector $\mathbf{b}$ is constructed as a column vector filled with the values δr_i, one for each anchor node. Note that the i'th row of the matrix A needs to correspond with the i'th row of vector $\mathbf{b}$ with respect to the anchor node the values are calculated off. The least squares solution of equation (1) will then look like this:

$$\begin{bmatrix} \mathbf{1}_1 \cdot w_1 \\ \vdots \\ \mathbf{1}_n \cdot w_n \end{bmatrix} \cdot [\delta \mathbf{x}] = \begin{bmatrix} \delta r_1 \cdot w_1 \\ \vdots \\ \delta r_n \cdot w_n \end{bmatrix} \tag{3}$$

This way the improvement $\delta \mathbf{x}$ is calculated. In general, the least squares solution of $\mathbf{x}$ for $A\mathbf{x} = \mathbf{x}$ is: $\mathbf{x} = A \cdot C \cdot \mathbf{b}^T$ where $C = (A^T \cdot A)^{-1}$.

Using covariance matrix C, a square matrix with the number of rows and columns equal to the dimensionality of the system, the node's standard deviation is calculated as:

$$\sigma^{node} = \sqrt{\frac{1}{2} \sum_i \sum_j C_{i,j}} \tag{4}$$

When the estimated position of the nodes is calculated like this, the optimal location will be calculated, based on the given ranges to and positions of the neighbor nodes.

4.2 Multi-hop Distances

Calculation of the distance between two nodes **A** and **B** that are connected only by a path of more than one hop can be performed indirectly. On each hop along the shortest path between the nodes, the range and corresponding standard deviation are measured. An estimation of the distance and precision between **A** and **B** can be made, however, by taking the sum of all these ranges. The distance calculated this way, called r_m will usually be larger than the true distance between the two nodes, and the standard deviation will be larger than the summed single-hop standard deviations σ_m, because of the error introduced by the less precise calculated distances. A better estimation of those values can be made, however, by statistically analyzing large sets of these summed measurements.

Every measured summed distance r_m can be seen as a sample of a random variable R_{r_m}. If this variable is of Gaussian distribution, it serves as a good bases to obtain estimation of the actual distance. The expected value of R_{r_m} is the mean of the distribution, $E(R_{r_m}) = \mu_R$. Normalizing R_{r_m} by the measured distance r_m results in a normalized random variable $Q = R_m/r_m$ for all values of

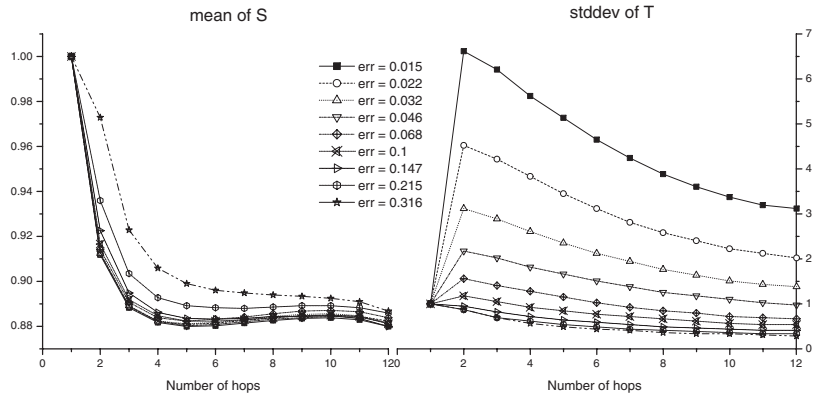

Fig. 1. Bias table values for μ_S and σ_T

r_m measured along paths with equal number of hops, from which the estimated value of the true distance r_t can be calculated as: $r^{est} = \mu_Q \cdot r_m$. The error between the measured distance and the estimated true distance $\epsilon_{r_m} = r_m - r^{est}$ can, in the same way, be seen as a sample of the random variable S_{r_m}. The standard deviation σ_S of S_{r_m} is the precision of the estimated distance r^{est}.

A normalized random variable T is defined as having samples ϵ_T, obtained by dividing all samples ϵ_{r_m} by the expected error η on the true distance r_t: $\epsilon_T = \dfrac{(r_m - r^{est})}{r_t \cdot \eta}$.

The expected error η is an environment and hardware dependent value of the range measurement error (given as a percentage of the range) as mentioned earlier in Section 3.1. From the standard deviation σ_T of T the precision of the estimated true range r^{est} can now be calculated from σ_m and σ_T as: $\sigma^{est} = \sigma_m \cdot \sigma_T \cdot \mu_Q$.

The values for σ_T and μ_Q can be calculated offline, and stored in a table, containing one set of (σ_T, μ_Q) values for all possible hop lengths of the shortest hop paths between any pair of nodes. These values for single hop paths (for nodes that are each others direct neighbors) will of course be $(1, 1)$. The value of σ_T does depend on the η, so for different values of η another table will have to be used. This value does however only depend on the hardware and environment used for the network, which will be known before deployment of the network, so only one table will have to be stored or in active use throughout the whole localization calculation. Figure 1 shows the values of σ_T and μ_Q for values of η and different hop counts.

5 Simulation

5.1 General Set-up

To test the performance of described system, a simulation environment was built using the OMNeT++ discrete event simulator [14]. The tests performed in the simulation environment are described is detail in Section 6. The network communication model used in the simulation makes use only of local broadcasts, where messages are delivered to all nodes within a certain and fixed range from the sending node. Colliding transmissions and message corruption are abstracted from (it is assumed that lower network layers will be in charge of providing a reliable broadcast service). The positions of all nodes are fixed during a simulation run, and are determined at initialization. The simulation environment uses a 2-dimensional coordinate system. Each node's coordinates are selected randomly and uniformly distributed within a square region of a size in units equal to the amount of nodes in the network.

During network initialization a number of nodes is selected as anchor nodes, in a way similar to the one used by Langendoen et al. [6]. The anchors are selected in a grid-like structure, where the nodes closest to the points of the grid will be chosen to be anchors. The grid is of size $s \times s$, $s \leq N$ with N the number of anchor nodes, and s the largest number that still satisfies the inequality. The rest of the anchors are selected randomly from the remaining nodes. This way of selecting the anchor nodes is chosen because it provides a more or less equal environment for all nodes, where the closest anchors will always be a certain maximum distance away. It is beneficent as well for the performance of the algorithm, because it proves difficult to obtain a correct estimation of a node's position if it is not surrounded by anchor nodes, as can be seen in the simulation results. Another reason to choose this strategy is because it allows easy comparison to the results described in [6].

The distance measurements between nodes are calculated from the signal strength of a transmission. This means that every received message also provides a range indication between itself and the sending node. In reality the calculation of this range infers a certain amount of error, as mentioned earlier. In the simulation the range measurement including error is modelled by drawing a random value from a Gaussian distribution with the true range as its mean, and a standard deviation of $\eta \cdot true\,range$. The implementation of the algorithm records all range measurements from each message it receives, and uses the average distance measured to a neighbor node in all further calculations. This way, as more messages are received, the measured distance will converge to an ever more constant value, which leads to better position estimates during refinement.

5.2 Algorithm Details

The protocol uses a two-phase approach, and relies on two corresponding kinds of messages being passed between the network nodes; start-up messages and refinement messages. During the start-up phase, a node attempts to calculate

an initial position estimates, based on the distances towards the anchors. This initial position will then be improved to get a more accurate estimate during the refinement phase.

Start-up messages contain information about the distance and hop count, as well as the position of another node (with known, or at least estimated position) to the sending node. Refinement messages contain the position of the node that sent it. When the localization algorithm starts, usually when the node is switched on, or awakes from a sleep state, a non-anchor node starts by broadcasting an announcement message, indicating its presence and requesting for information from nearby nodes. On activation, an anchor node directly starts by broadcasting its own position. For its surrounding nodes, this is an indication of its presence as well. Upon receiving an announcement message, a node broadcasts all information currently known to that node. This way, the new node quickly learns the positions of all nearby nodes, and helps the newly connected node to catch up on the current state of the network. For all nodes in start-up state, all received messages with hop count lower than the maximum are stored, until enough distances (at least 3 in the 2-dimensional simulation) are known to be able to calculate its own position estimate. If an estimate can be calculated, the node will proceed to refinement state, and broadcast the position estimate just calculated.

Nodes in refinement state only collect refinement messages from direct neighbors. After receiving a position update from one of its neighbors, the node can re-calculate its own position, based on the new or more precise position just received, together with all other neighbor's positions. If this new position estimate is more precise than the previously known location (smaller σ), this new position estimate is broadcast in a refinement message. This strategy ensures finiteness of the algorithm. After each round of message passing, the improvement in calculated precision will be smaller. If a node cannot calculate a more precise position update, it will not rebroadcast its position. All other nodes will recalculate their positions based on fewer position updates from their neighbors, and the new estimate will be more likely not to be more precise, in which case the node will stop re-broadcasting as well. After a few rounds no node will be able to improve its position estimate, and the algorithm stops.

As the protocol progresses, nodes update their position estimates based on the improved position estimates of their neighbor nodes, and possibly a neighboring anchor node. These improved position estimates of the neighbor nodes in their turn, were calculated from the previous estimate of their neighbors. So, in effect, a node's new position is calculated indirectly from its own previous improved estimate. It is important that all nodes recalculate their positions on a similar rate, so ensure each node keeps a reliable calculated precision of its position estimate, relative to it's neighbors. This is ensured by allowing a node to re-broadcast only after a certain, network wide, fixed time after the previous position estimate was broadcast. This also keeps the number of broadcasts low.

Anchor nodes already know their position from the start. Because their position is obtained in a way external to this algorithm, they do not recalculate it.

As a result, for the majority of the time, they can be silent, and only react to announcement messages by rebroadcasting their fixed position. The precision of anchor nodes positions is obtained externally as well, and can be much greater than possible to achieve with the calculations of this localization algorithm. In the simulations, the anchor nodes precision is set to be infinitely high.

6 Results

With the simulator described above, a series of tests have been performed to obtain measurements on different performance factors of the system. The performance characteristics depend on several factors in the environment of the system, like the number of nodes, connectivity, error size, and number of anchors. These values can be set as input parameters for the simulator. This large set of inputs makes it impossible to thoroughly test the system through the complete range of all combinations of input parameters. Therefore, a set of parameter values is chosen, from which one value is changed every time to test the system's sensitivity to that particular parameter. These standard values are:

- Number of nodes: 225, placed on a square area of 15 units length;
- Radio range: 2.1 units, results in connectivity of about 12;
- Relative range error η: 10%;
- Number of anchors: 5% = 11 anchors;
- Multihop distance hop count: 4 hops.

These values are chosen to be equivalent with the parameter values of the standard scenario in [6], again for easy comparison of the respective results.

Because of random effects, produced by the use of randomly selected error values, node positions and possibly other factors, consistent results on all of the performed tests can only be obtained by averaging over large quantities of individual test runs. To ensure consistency, all tests performed have been repeated at least 50 times.

6.1 Distance Error

As a measure of the performance of the described algorithm, as well as an indication of the usefulness in specific application areas, the accuracy of the algorithm, as well as its precision are important indicators. The accuracy can be described as the average distance between a node's actual position and its position estimate. The precision describes to what extent the real position error is near this average distance error. This is calculated using the standard deviation of the error values. In order to test those values, an initial test has been performed using the parameter values of the standard scenario. Figure 2 shows the error distribution of the relative position error along with the average and standard deviation values, indicated with the dashed line and two dash-dotted lines at each side of it at $\mu - \sigma$ and $\mu + \sigma$. Note that, even though the majority of the error values stay below 10%, the average error value is around 17%. This is due

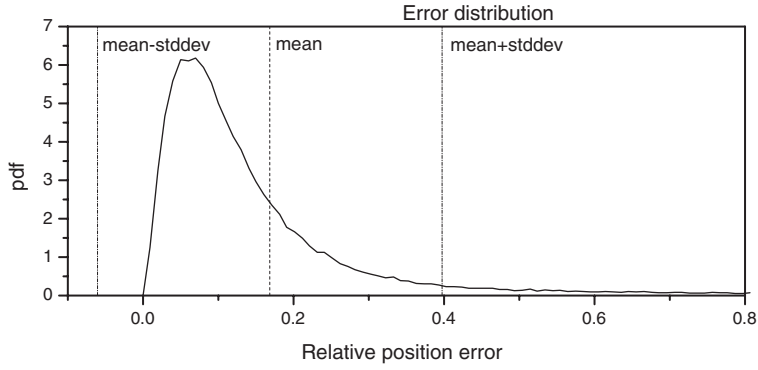

Fig. 2. Position error distribution for standard scenario

to the fact that a small number of nodes have relatively large errors, of up to and beyond the radio range. This fact is illustrated as well by the large standard deviation of about 0.23.

Even though, as the results show, each node's position cannot always be estimated very accurately, it does perform well on assigning a position estimate to all nodes. On average over the test runs executed in this first test a coverage factor of 98.9 % has been measured, meaning almost all of the nodes have obtained a position estimate, with the exception of the few nodes that do not connect to enough neighbors to actually start an estimation.

In the next series of simulations, the algorithm's sensitivity to variations in radio range, range error, and the fraction of anchors is tested. Figure 3 shows the results of these tests. In all three diagrams, both the average distance error and the standard deviation are shown, as well as the coverage factor.

The simulation results show that the system hardly is sensitive towards number of anchors, except for really small numbers. At an anchor fraction value of 1.8%, which is 4 anchors, no nodes are able to obtain a position estimate during the start-up phase. This is because the four nodes are placed at the corners of the area, and there is no node that has 3 anchors near enough to receive multi-hop distance information to them. This can be solved of course by increasing the multi-hop hop count, so the distance information from the anchors travels for a greater number of hops. From a value of 4% onwards the system shows little change in both relative error (average and standard deviation) and coverage.

As for the radio range, from a connectivity of about 8 to 10, the result is nearly the same, showing little change in relative range error. Note that the radio range and connectivity are related linearly, because of the uniform distribution of the nodes in the deployment area. In the results for both the anchor fraction and radio range variations, the average and standard deviation of the distance

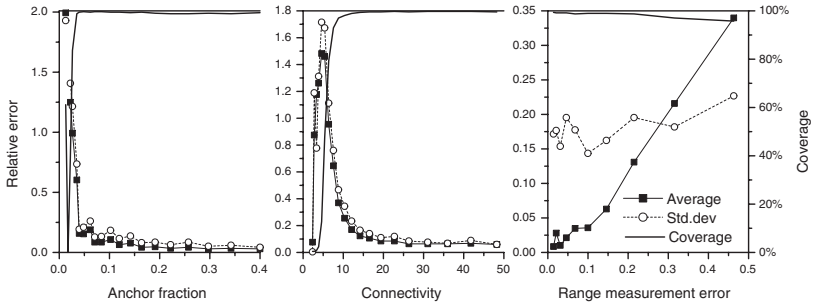

Fig. 3. Sensitivity towards anchor fraction, radio range and range error: connectivity (unmarked, right scale), average (square mark) and standard deviation (circle mark)

error have proportional values. This indicates that the distribution of the range error is similar to the one shown in Figure 2, scaled along the x axis.

With respect to the range error sensitivity, this is somewhat different. While the standard deviation keeps a more or less constant value around 0.27, the average error does change linearly with η. The size of the range error is directly reflected by the accuracy of the system, whereas the precision remains largely unchanged.

6.2 Network Communication Cost

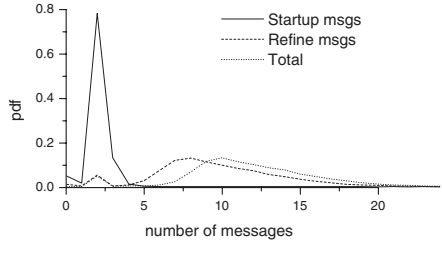

Fig. 4. Distribution of number of Startup and Refine messages for standard scenario

In a real-world implementation of an ad-hoc sensor network, energy preservation is of great importance. Because network communication is largely responsible for the energy consumption, this figure is of major interest as well. Some

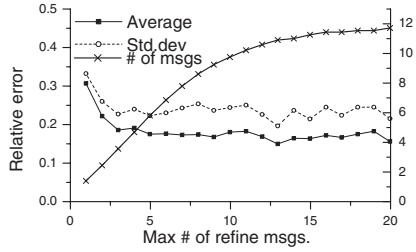

Fig. 5. Relative distance error and average number of Refine messages as a result of limiting the number of Refine messages

additional tests have been performed to quantify the network communication cost of this system.

Both the Startup and Refine messages have a small, fixed size. The network communication cost can thus be quantified by the number of messages broadcast per node. Figure 4 shows the probability distribution of both message types, and the total number of messages for the standard scenario. It is clear that the Refine messages take up the majority of the total number. Decreasing the number of Refine messages directly improves the total communication costs.

The algorithm can be adapted so that every node will only send a certain amount of Refine messages. The implications of this in relation to the distance error are shown in Figure 5. It clearly shows that after the first 3 or 4 Refine messages more communication hardly makes any improvement in accuracy and precision, while the average number of messages increases linearly. Limiting the number of Refine messages is an easy way to improve the total performance of the system.

6.3 Processing Requirements

Considering the small size and low cost of the targeted devices, only very limited processing resources will be available. The processing time to complete the localization process is an important aspect of this particular system.

As the central part of the system, the Iterative Weight Least Squares Estimation takes up the majority of processor cycles in the total calculation. This part of the calculation is therefore a good indicator of the total processing time used.

A number of factors are responsible for the total time each node spends calculating its (updated) position. At first, the size of the matrices used, and thus the spent calculation time, is proportional to the amount of neighbors involved in the calculation. Besides that, the number of iterations taken for the calculation to complete is a proportional factor to the total calculation time. At last, a third factor is the number of times the IWLS calculation is performed. In general, at

least every time before a Refine message is sent, a node's position is re-calculated several times, until a more precise result is found.

The factors involved in calculation time, as mentioned above, have been examined, using the data sets generated for the earlier tests, described in Section 6.1. Figure 6 shows the number of times a calculation is performed per node, average number of IWLS iterations per calculation, average number of neighbors involved, and number of calculations per broadcast Refine message. The solid, unmarked line, on the right scale, shows the grand total of the number of calculations, number of iterations and number of neighbors multiplied. This is a number proportional to the total calculation time spent per node in the IWLS calculation for the whole localization algorithm.

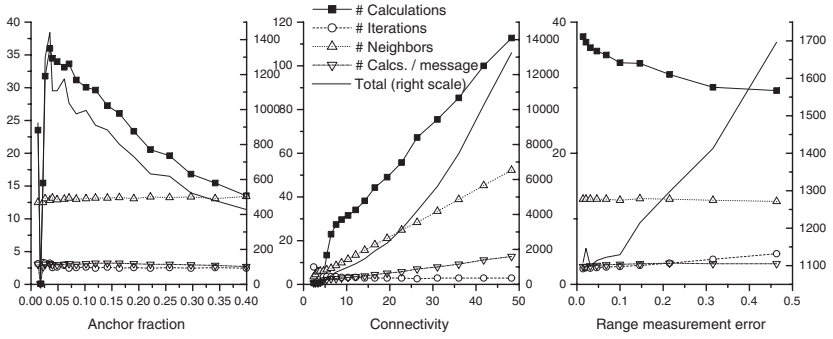

Fig. 6. Nr. of calculations for different parameter values. The number of calculations, iterations, neighbors and calculations per message are drawn on the left scale; the total calculation time per node is shown by the unmarked solid line, drawn on the right scale.

For the targeted devices, it is very important to only spend energy and processing time as long as it gives improvement to the positioning accuracy. Selecting the right set of parameters optimal to the deployment area to ensure this is should be considered carefully. Reducing the processing cost can be achieved by reducing either of the above-mentioned factors.

Comparison with Figure 3 shows that limiting the amount of neighbors in the IWLS calculation greatly reduces the processing cost, while this hardly has an influence on the range error performance of the system. From the graphs it is clear that the number of calculations per broadcast message is an almost constant factor.

7 Discussion

The previous chapters have covered the description of the algorithm used in obtaining the position estimate in a distributed network.

The implementation of the system presented in this paper is only a simple, straightforward one. As has been shown in the previous chapter, performance improvements can be made, by making small changes, based on close inspection of the results obtained by selectively changing certain parameters that are of importance in the whole calculation of the algorithm. As one of such improvements, multi-hop distances towards indirect neighbors could also be used in the refinements phase, instead of just to obtain initial position estimates. This could be especially useful for networks with a low connectivity. It does promise to pay off in terms of performance increase, to spend some time fine-tuning the algorithm, based on a particular application area.

As noted before, similar algorithms have been designed, and can serve as good comparison material. Langendoen ([6]) provides a comparative test for combinations of different algorithms. Even though the general approach is somewhat different, the refinement steps are almost identical to the one used in our system. The data after refinement are comparable to the results presented in this paper.

The multi-hop distances and IWLS calculation are quite similar to the *sumdist* and *lateration* methods described in [6] and others, with the exception that use is made in our system of the precision indications available with the messages.

From the results presented in [6], it is clear that the refinement algorithm only is not a very useful addition to the first two phases except in very specific conditions. The use of refinement dramatically reduces the coverage of the whole system to levels below 50% in all cases except when the distance measurement errors are very small ($\eta < 0.05$). Under these conditions, the position error can drop to 25% or even 20%, although coverage of just 60% can be reached. Without the refinement phase, the various combinations of stage 1 and 2 algorithms can only reach a position error of 42%, and less for higher connectivity, but with 100% coverage. Having full coverage is very important, of course, since acquiring a position estimate is the intended goal of these algorithms.

Our results, on the other hand, show better position estimates, while practically full coverage can be reached, in almost all situations. In all but the most extreme environments, a position error of 25% or less can be achieved, while keeping 95% coverage. Large distant measurement errors or low connectivity do make the numbers less feasible, showing the application limits for this algorithm. But in more optimal conditions, position errors as low as 10% can be reached, with near 100% coverage. Table 1 summarizes the results for the algorithms mentioned above.

The improvement in position error with our proposed solution is most likely caused by the availability of more measurement data, namely the precision indication, even though not a lot can be said about this in general, since the obtained location precision is a result of many interrelated factors.

In terms of communication overhead, our solution proves to be competitive as well. In [6] it is stated, that the initial flooding (including calibration) of the

Table 1. Comparison of the presented algorithm with the ones described in [6], on relative position and connectivity

	Condition	Pos.err.	Coverage
Traditional alg. + refine	$\eta < 0.05$	20-25%	50-60%
Traditional alg. phase 1+2	$conn. > 12$	35-42%	100%
Precision based algorithm	$\eta < 0.25$	15-25%	95-100%
Precision based algorithm	$\eta < 0.1, conn > 12$	10-15%	98-100%

network costs about 3.5 to 4.8 messages per node, depending on the algorithm used. This is comparable to the average number of startup messages, which is 2.1 in our solution. The amount of messages needed during refinement is largely dependent on the limit imposed on the system, and in both algorithms, the resulting position error is constant from about 5 messages onwards. All systems use messages of a small, constant size.

It is hard to compare the calculation requirements of all algorithms, especially because only simulations are available up to this pint in time of both the system described in this paper and the ones in [6]. Besides that, Langendoen et al. do not give any metrics about calculation time or complexity in their paper. At this point, no conclusions can be made about how the different algorithms compare in terms of calculation time.

It is shown that, using RSSI as a distance measurement system, position errors of as low as 10% can be reached. Even though this could be not sufficiently accurate for use in all kinds of situations, it does provide good enough results to at least be able to know about the networks topology, and coverage area of the individual nodes. Altogether, it can be concluded that the goal of obtaining position information of nodes in a distributed ad-hoc network, making use of only rather inaccurate measurement techniques, is within reach.

8 Conclusions and Future Work

In this paper we presented a distributed localization algorithm for ad hoc wireless networks, which takes into account the precision of measurements. It is aimed at networks where the nodes are static. The algorithm takes a two-step approach: first an initial location estimation is made, based on distance measurements obtained from RSSI readings. Subsequently, refinements are calculated. Both steps, including the calculations involved, are described in detail, after which the performance is discussed. In a 225-node network, of which 5% are anchors, with 10% range error, a relative distance error of about 16% can be achieved, with a nearly 100% coverage. In general, the results of this approach yield 2 to 4 times better results in position accuracy than other systems described previously. This level of performance can be reached with just 10 or fewer messages broadcast per node in the network, which are of small, constant size. Details about the calculation cost are discussed as well, and some suggestions are given on how to optimize the performance of the algorithm for real world implementations.

The results provided in this paper are all obtained from simulations. Further research will have to be done to find how the algorithm performs in a real world situation, where certain parameters could be different from the assumptions made in this paper. A hardware platform is being designed that can be used to further test the system.

The described system is applicable only in networks where the nodes are static, but many applications include mobile nodes in the network. By itself, that poses many new problems. The nature of our system is designed in such a way that with only little adjustments, it can also be made useful in networks with moving nodes. In the future, more research can be done to make this possible.

References

1. M.Srivastava A.Savvides, C.-C.Han. Dynamic fine-grained localization in ad-hoc networks of sensors. In *7th ACM Intl. Conf on Mobile Computing and Networking (MOBICOM), p.166-179*, 2001.
2. Paramvir Bahl and Venkata N. Padmanabhan. RADAR: An in-building RF-based user location and tracking system. In *INFOCOM vol.2*, pages 775–784, 2000.
3. Jan Beutel. Geolocation in a picoradio environment. In *MS Thesis, ETH Zurich, Electronics Lab*, 1999.
4. N. Bulusu, J. Heidemann, and D. Estrin. Gps-less low cost outdoor localization for very small devices. In *IEEE personal communications*, pages 28–34, 2000.
5. Jeffrey Hightower and Gaetano Borriella. Location systems for ubiquitous computing. *IEEE Computer*, 34(8):57–66, 2001.
6. Koen Langendoen and Niels Reijers. Distributed localization in wireless sensor networks: A quantitative comparison. In *Computer Networks (Elsevier), special issue on Wireless Sensor Networks*, 2003.
7. L.Doherty, K.Pister, and L.El Ghaoui. Convex position estimation in wireless sensor networks. In *IEEE INFOCOM, Anchorage, AK*, 2001.
8. L.Evers, W.Bach, D.Dam, M.Jonker, H.Scholten, and P.Havinga. An iterative quality based localization algorithm for adhoc networks. In *Department of Computer Science, University of Twente*, 2002.
9. P.Misra, B.P.Burke, and M.M.Pratt. Gps performance in navigation. In *Proceedings of IEEE, vol.87, nr.1, pp.65-85*, 1999.
10. P.Rentala, R.Musunuri, S. Gandham, and U. Saxena. Survey on sensor networks. Department of Computer Science, University of Texas.
11. Chris Savarese, Jan Rabaey, and Koen Langendoen. Robust positioning algorithms for distributed ad-hoc wireless sensor networks. In *USENIX technical annual conference, pp.317-328, Monterey, CA*, 2002.
12. Andreas Savvides, Heemin Park, and Mani B. Srivastava. The bits and flops of the n-hop multilateration primitive for node localization problems. In *WSNA, Atlanta, GA, pp.112-121*, 2002.
13. S. Simic and S. Sastry. Distributed localization in wireless ad hoc networks. Tech. report, UC Berkeley, 2002, Memorandum No. UCB/ERL M02/26., 2002.
14. Andras Varga. The omnet++ discrete event simulation system. In *Proceedings of the European Simulation Multiconference (ESM'2001)*, 2001.
15. Roy Want, Andy Hopper, Veronica Falcão, and Jonathan Gibbons. The active badge location system. In *ACM Transactions on Information Systems*, pages 91–102, 1992.