Efficient Localization for Wireless Sensor Networks Using Power Measurements Sampling

Charalampos Papamanthou* Franco P. Preparata* Roberto Tamassia*

Abstract

In this paper, we re-examine the RSSI measurement model for location estimation and provide the first detailed formulation of the probability distribution of the position of a sensor node. We also show how to use this probabilistic model to efficiently compute a good estimation of the position of the sensor node by sampling multiple readings from the beacons. The results of the simulation of our method in TOSSIM indicate that it is competitive with previous approaches.

1 Introduction

Estimating the location of a roaming sensor is a fundamental task for most sensor networks applications. For example, if a sensor network has been deployed to provide protection against fire (in this case, sensor nodes report a sudden increase in temperature), we want to know the location of the sensor that triggers an alert so that action can be taken accordingly. Additionally, some routing protocols for sensor networks, such as geographical routing [15, 40], make routing decisions based on the knowledge of the locations of the sensor nodes.

There are several proposed location estimation protocols for sensor networks, see, e.g., [5, 7, 8, 11, 21, 22, 23, 24]. All these protocols use the same model, where some nodes *know* their location (either because they are fixed or by using GPS) and are called *beacons* or *anchor nodes*, and some other nodes, called *sensor nodes*, estimate their location using the information they receive from the beacons. This information consists of the beacons' coordinates and of features of the beacon signal, such as the *received signal strength indicator* (RSSI) or the *time difference of arrival* (TDoA). Also, other protocols (e.g., [23]) are based on the capability of the nodes to sense the angle from which a signal is received.

After performing a certain number of such measurements for different beacons, the sensor node has to combine all this information (for RSSI, this information is the power of each individual signal and the coordinates of the corresponding transmitter) in order to estimate its location. The location estimation algorithm has the following requirements:

- The sensor node should avoid complex and time consuming computations, which would deplete its energy supply (typically a low-cost battery) rapidly.
- The computations should take into consideration the error in the measurements, which can be large.

Several previous approaches use computationally demanding methods, such as convex optimization [8], systems of complex equations [23], *minimum mean square error* (*MMSE*) methods [6, 30], and Kalman filters [31]. In these approaches, the measurement model is not adequately analyzed and the error is assumed to be small, which is not the case in most real applications of sensor networks.

Other approaches, notably [16, 27, 33], estimate the location of a node using the RSSI method (analyzed in [28]), which is the most realistic model for sensor network communication. In [16], the authors evaluate the ability of a sensor network to estimate the location of sensor nodes. They assume that the location of the sensor node is known and develop arguments concerning the probability that the network will detect this location. They use the RSSI error model to analyze the problem of *evaluating* the ability of the sensor network to locate a sensor node. However, they do not describe how their algorithms can be implemented on a sensor node to estimate its own location. Moreover, their method does not take into account the basic parameters of the RSSI model (standard deviation and path loss exponent) and thus gives incorrect results.

In this paper, we formulate the correct probability distribution of the position of a sensor node based on one reading produced with the RSSI model. Due to the errors implicit in the RSSI model, it is unrealistic to try to compute a good estimation of the location of the sensor node based only on a single measurement (or even few measurements) from each beacon. Such an approach would be so inaccurate to make the

*Department of Computer Science, Brown University. Email: {cpap,franco,rt}(at)cs.brown.edu.

estimate practically worthless. Notwithstanding this difficulty, we show that a reliable estimation of the location can be achieved by processing a reasonably small number of readings of the signals.

Especially for indoor positioning systems, this is an assumption that has been extensively used. For example, in [13, 14], the position estimation is based on a *location fingerprint* $\mathbf{t} = [t_1\ t_2 \dots t_N]$, where N is the number of beacons and t_i $(i = 1, \dots, N)$ is the mean value of the received signal strength over a certain time window. Also, in [3, 9, 19, 38], experiments with various sample sizes are presented where the samples are used to compute certain features of the signal strength such as the standard deviation and the path loss exponent. Finally, in [12], simulations are presented that use various number of samples, where it is stated that there is a need for more than 50 samples to *filter out* the errors in the probability distribution.

We show that using only the mean of the measurements is not a correct procedure, due to the lognormal distribution of the distance from the beacon (see Theorem 4.1). Instead of using directly the mean value, we use another value that is adequate according to the specific underlying probability distribution of the distance. The number of samples that are used vary from 20 to 300 and obviously the accuracy of the computed location grows with the number of samples. Finally, once the sampling has been performed, we show how to seek the minimum of a function that approximates the actual location with small computational effort.

1.1 Our Contributions

The main contributions of this paper are as follows:

- We evaluate the probability that a sensor node lies within a certain region, given that the power received from the beacons is modeled with RSSI. To the best of our knowledge, this is the first detailed formulation of the probability distribution of the position of a sensor node. We show that unlike the normal distribution of the received power, the probability distribution of the actual position is lognormal. Thus, we give evidence to the role of the parameters σ and n in the probability distribution of the actual distance, where σ is the standard deviation of the normal variable that models the power received by the sensor and n is a parameter, called *path loss exponent*, that depends on the transmission medium. In previous approaches [16], the probability distributions used did not exhibit dependency on these two variables.
- We present a method for estimating the location of a node from multiple sample power readings from the beacons. Our method computes the *expected* value of the received power and combines it with the mean and the standard deviation of the sample readings using a steepest descent approach [34]. We show that our method is simple and efficient and provides a good estimation of the position. Note that using multiple sample readings is necessary for a reliable location estimation. Indeed, the probability distribution of the location for a single sample implies that the domain within which the sensor lies with high probability has large area.
- We describe an implementation of our location estimation algorithm that is suitable for execution on standard sensor hardware and we analyze the results of an extensive simulation of the execution of the algorithm in TOSSIM [10, 17]. The results of the simulation show that our method has accuracy that is comparable to or better than that of previous methods.

1.2 Organization of the paper

The rest of this paper is organized as follows. In Section 2, we present the definition of the problem and an analysis of the RSSI model adopted in sensor networks. Section 3 focuses on the probability distributions of the relative error, the actual distance and the position. We give results that define the correct probability distributions of the position of a sensor node due to power measurements. In Section 4, we describe the main method that uses sampling to compute an estimate of the position, based on probabilistic facts developed in the preceding sections and give a time-efficient minimization-based method for estimating the actual location. Theorem 4.1 is the main result of this section indicating a way to compute a good estimation of the actual distance given a certain number of samples (which is different than a simple mean and takes into consideration the lognormal distribution). Finally, in Section 5, we give extensive simulation results using the TOSSIM simulator [10, 17] for sensor networks and we also present a comparison (in terms of localization error) of our results with other methods. We conclude in Section 6.

2 Preliminaries

Suppose are given a region of the plane with k beacon nodes $b_1, b_2, \dots, b_k$ (nodes of known location). The coordinates of the beacons are (x_i, y_i) for $i = 1, \dots, k$. The beacons transmit information about their location with a signal of normalized intensity to a sensor node s that does not know its location. Based on

the locations of the beacons and the estimated distances from the beacons (computed from the received signals), the sensor computes its actual location. In the absence of distance estimation errors, the problem could be easily solved by using three beacons. We would have to solve a system of three circles equations and determine their common intersection point. However, errors occur and the distance estimations made by the sensor nodes are not accurate. Thus, even in the unlikely case that the estimated distances yield a single solution to the system of the circles equations, we cannot conclude that the intersection point is the actual location of the sensor node.

Among the several models proposed for estimating the distance between a beacon and a sensor node, the most realistic and commonly used one is the *received signal strength indicator* model (RSSI) [28]. In this model, the beacon broadcasts signal to all sensors and the sensors can estimate the distance between them and the beacons on the basis of the strength of the signals they receive.

Let b_i be a beacon located at (x_i, y_i) and s a sensor node located at (x, y). We define the *relative error* ϵ_i pertaining to b_i as follows. Suppose that s reads a distance $\hat{r_i}$, while the actual distance is r_i. The relative error is

$$\epsilon_i = \frac{\hat{r_i}}{r_i} - 1 \in [-1, +\infty) \tag{1}$$

The commonly accepted transmission model [28] expresses the received power p_i (in dBm) as

$$p_i = p_0 + 10n \log\left(\frac{r_i}{r_0}\right) \tag{2}$$

where p_0 is the received power in dBm at a reference distance r_0 and n is the path loss exponent which is a constant depending on the transmission medium (indoors, outdoors) and ranges typically from 2 to 4. In some environments, such as buildings, stadiums and other indoor environments, the path loss exponent can reach values in the range of 4 to 6. On the other hand, a waveguide type of propagation may occur in tunnels, where the path loss exponent drops below 2.

We recall that if the received power in mW at a point k is P_k, and $P_{k'}$ is the received power at some reference point k' (again in mW), then the received power p_k in dBm at point k is defined as

$$p_k = 10 \log\left(\frac{P_k}{P_{k'}}\right)$$

The measured power, however, differs from that given equation (2); due to channel fading (variation of the received signal power caused by changes in transmission medium or path), the measured power is

$$\hat{p_i} = p_i + x \tag{3}$$

The random variable x represents the medium-scale channel fading and is typically modelled as Gaussian zero-mean with variance σ^2 (in dBm). Typically, σ is as low as 4 and as high as 12 (this implies that the error may be large). Inserting $\hat{p_i}$ and $\hat{r_i}$ into (2), we get

$$\hat{p_i} = p_0 + 10n \log\left(\frac{\hat{r_i}}{r_0}\right) \tag{4}$$

where now the measured power $\hat{p_i}$ in dBm relates to the measured distance $\hat{r_i}$ by the sensor. By combining the above equations, we get that the relation between the measured distance and the actual distance is

$$\hat{r_i} = r_i 10^{\frac{x}{10n}} \tag{5}$$

which gives

$$\epsilon_i = 10^{\frac{x}{10n}} - 1 \tag{6}$$

3 Probability Distributions

As mentioned before, the random variable x is assumed to be Gaussian zero-mean, i.e.,

$$P_x(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x^2}{2\sigma^2}} \tag{7}$$

By equation (6), we have that $x = 10n \log(\epsilon + 1)$. It is well known [36], that, given the probability density function $f(v)$ of a variable v, the probability density function of a variable u, related to v by $v = g(u)$ is $f(g(u)) \left|\frac{dg(u)}{du}\right|$. Therefore, we have that the probability distribution of the relative error is log-normal:

$$\begin{aligned} P_\epsilon(\epsilon) &= P_x(10n \log(\epsilon + 1)) \frac{dx}{d\epsilon} \\ &= \frac{10n}{\ln(10)} \frac{1}{\epsilon + 1} \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(10n \log(\epsilon+1))^2}{2\sigma^2}} \end{aligned} \tag{8}$$

To compute now the probability density function $P_{r_i}(r_i)$ of the actual distance r_i, we must use (1) and get

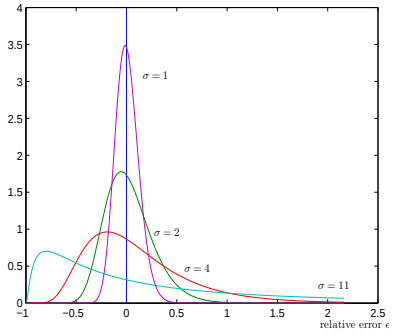
$$\begin{aligned} P_{r_i}(r_i) &= P_\epsilon\left(\frac{\hat{r_i}}{r_i} - 1\right) \frac{\hat{r_i}}{r_i^2} \\ &= \frac{10n}{\sigma\sqrt{2\pi}\ln(10)} \frac{1}{r_i} e^{-\frac{(10n \log(\hat{r_i}/r_i))^2}{2\sigma^2}} \end{aligned} \tag{9}$$

Note that the maximum of this function clearly depends on both n and σ. Based on the above distributions, we can prove the following theorem for the probability distribution of the position (the details are omitted due to space limitations).

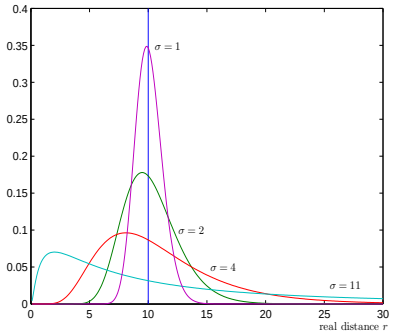
Theorem 3.1 *Let b_i be a beacon node located at (x_i, y_i) sending information to a sensor node under the RSSI model with standard deviation σ and path loss exponent n. Let $\hat{r_i}$ be the measured distance from beacon node b_i at the sensor node. We have that the probability density function of the actual position (x, y) of the sensor node is given by*

$$P^{(i)}_{X,Y}(x,y) = \frac{10n e^{-\frac{\left(10n\log\left(\frac{\hat{r_i}}{\sqrt{(x-x_i)^2+(y-y_i)^2}}\right)\right)^2}{2\sigma^2}}}{2\pi\sigma\sqrt{2\pi}\ln(10)((x-x_i)^2+(y-y_i)^2)}$$

.



(a) The probability density function of the relative error ϵ for $\sigma = 1, 2, 4, 11$ and measured distance $\hat{r_i} = 10$.



(b) The probability density function of the actual distance r for $\sigma = 1, 2, 4, 11$ and measured distance $\hat{r_i} = 10$.

Figure 1: Probability distributions of the measured distance and the relative error.

In Figures 1(a), 1(b) we plot the probability density functions of the relative error ϵ and the actual distance r_i, as computed above. We point out that the maximum of $P_\epsilon(\epsilon)$ is achieved when $\epsilon < 0$, which implies $r_i > \hat{r_i}$. However, the maximum of $P_{r_i}(r_i)$ is achieved when $r_i < \hat{r_i}$. This is due to the term $\frac{\hat{r_i}}{r_i^2}$, appearing in (9). Also, note that, as σ increases, the peak of $P_{r_i}(r_i)$ moves towards the beacon.

3.1 Intuition

In Figure 1(b) we see that the peak of the probability distribution of the actual distance is always between 0 and $\hat{r_i}$, as can be formally proved. At first sight this is counterintuitive. However, we must reflect that the Gaussian zero-mean x is a power in dBm, yielding the result. Since x is normally distributed, we would expect that the points $\hat{r_i} \pm \alpha$ would have equal probability density. This would however be the case if, for example $\hat{r_i} = r_i + x$. In our case, $\hat{r_i} = r_i 10^{\frac{x}{10n}}$.

The plot of the probability density function defined in Theorem 3.1 is shown in Figure 2(a) where the base of the cylinder represents the circle formed by the measured radius $\hat{r_i}$ with respect to the certain beacon b_i. We can see that not only the value but also the location of the maximum of the probability distribution changes for various values of the standard deviation σ. Actually, for $\sigma = 2$ (Figure2(a)), the maximum is larger (and closer to the beacon circle) than the maximum of the distribution of $\sigma = 5$ (Figure 2(b)), where the error is larger. To simplify the notation, we set

$$\Phi_{b_i}(x,y) = P^{(i)}_{X,Y}(x,y)$$

to denote the probability distribution due to beacon b_i. Note that $\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty}\Phi_{b_i}(x,y)dxdy = 1$. Suppose now we have two beacons. We want to compute the corresponding probability distribution. Suppose that the plane is subdivided by a uniform grid into cells $\{(i,j) : i,j \in \mathbf{Z}\}$, of sufficiently small size d. The probability $\Pr(x \in (i,j))$ is very closely approximated by $\Phi_{b_1,b_2}(p_{ij})d^2$, where $p_{ij} \in (i,j)$ (for example p_{ij} is the point at the center of (i,j)) and $\Phi_{b_1,b_2}(.)$ is the density function.

If $\hat{r_1}$ and $\hat{r_2}$ are the measured radii, then with p_{ij} we associate the relative errors $\epsilon_1(i,j)$ and $\epsilon_2(i,j)$. Within this approximation, pairs of relative errors form a countable set

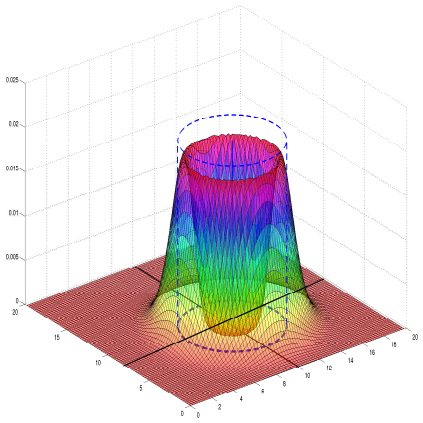
$$E_{\hat{r_1},\hat{r_2}} = \{\epsilon_1(i,j), \epsilon_2(i,j) : i,j \in \mathbf{Z}\}$$

and with (i,j) we associate the probability

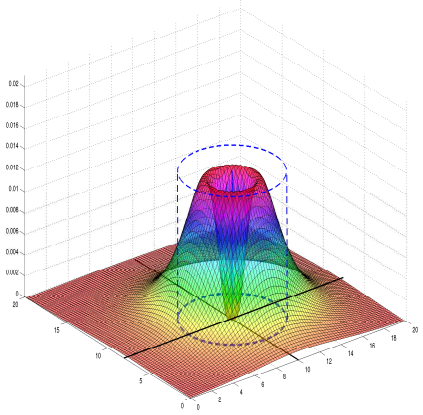
$$\Pr(\epsilon_1(i,j), \epsilon_2(i,j)) = \Phi_{b_1,b_2}(p_{ij})d^2$$

Since the measurements are independent

$$\Pr(\epsilon_1(i,j), \epsilon_2(i,j)) = \Pr(\epsilon_1(i,j))\Pr(\epsilon_2(i,j))$$

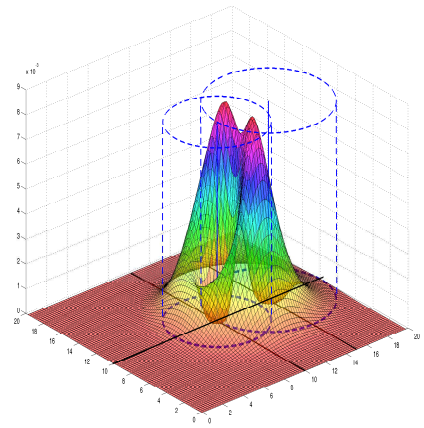


(a) Density function for one beacon and $\sigma = 2$. The circle is centered at (10,8) and $\hat{r} = 2$.

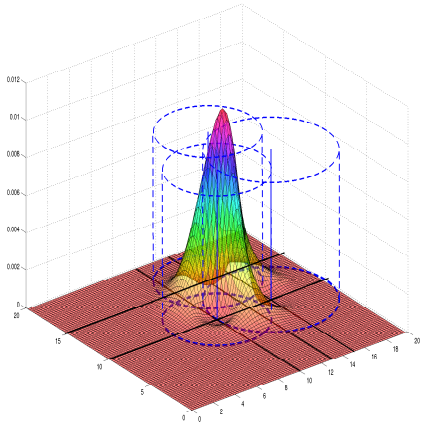


(b) Plot of the density function for the beacon of Figure 2(a) and $\sigma = 5$.

Figure 2: Probability density functions of the position. Note that there is a great level of uncertainty as to which the location of the node is. In both diagrams, the circle is centered at $(10, 8)$.



(a) Plot of the density function for two beacons.



(b) Plot of the density function for three beacons

Figure 3: Probability distributions of the position for more than one beacon.

Since $\Pr(\epsilon_k(i,j)) = \Phi_{b_k}(p_{ij})$ $(k = 1, 2)$, it follows that $\Phi_{b_1 b_2}(p_{ij})$ is proportional to $\Phi_{b_1}(p_{ij})\Phi_{b_2}(p_{ij})$, so that

$$\Phi_{b_1 b_2}(p_{ij}) = \frac{\Phi_{b_1}(p_{ij})\Phi_{b_2}(p_{ij})}{\sum_{s,t} \Phi_{b_1}(p_{st})\Phi_{b_2}(p_{st})}$$

We extend the argument to a finite set of beacons $B = \{b_1, \ldots, b_k\}$. If we then let $d \to 0$, we transition to the continuous case and obtain the following result.

Theorem 3.2 *Let $B = \{b_1, b_2, \ldots, b_k\}$ be a set of beacon nodes that send information to a sensor node under the RSSI model with standard deviation σ and path loss exponent n. If the measured distance from beacon node b_i at the sensor node is $\hat{r}_i$ $(i = 1, \ldots, k)$, then the probability density function (due to all the beacons in B) of the actual position (x, y) of the sen-*

sor node is given by

$$\Phi_{(B)}(x,y) = \frac{\prod_{i=1}^{k} \Phi_{b_i}(x,y)}{\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty}\left(\prod_{i=1}^{k} \Phi_{b_i}(x,y)\right) dydx}$$

where $\Phi_{b_i}(x,y)$ is the probability distribution due to beacon b_i, as defined in Theorem 3.1.

Plot of this density function for multiple beacons is shown in Figure 3(a,b). Note that even with 3 beacons there is great uncertainty about where the point lies. Additionally, if we use only one measurement, we may end up with a density function having more than one maximum. Moreover, the computation of this function on a sensor node is a very difficult task since there is no simple analytical expression for the maximum of the probability distribution computed in Theorem 3.2. Also, there is no analytical expression for the integrals needed to compute the probability for a certain region (so that we can integrate around the maximum of the probability distribution) and the computation of the integrals based on iterative methods are very demanding tasks to be executed on a sensor node. The path loss exponent in all plots is $n = 2$. We can also see that not only the value but also the location of the maximum of the probability distribution changes for various values of the standard deviation σ.

4 Samples of Measurements

In the previous sections we have examined the probability distribution of the sensor's position based on a single measurement. This setting however, can give rise to unacceptable errors for the values of σ ($\sigma = 4, 6, 8$) reported in the literature [28]. A consequence of this situation is that we may be unable to define a disk containing the sensor's location with an acceptable degree γ of confidence (say, $\gamma > 0.9$). Additionally, if our practice is based on only one reading, there is no way for the sensor to estimate the standard deviation σ (this is actually the standard deviation of the Gaussian random variable x introduced in Section 2) of each beacon. This parameter σ—conveniently assumed to be known in Section 2—has to be estimated in practice since it is needed in the computations (see below).

To overcome these difficulties, we show in this section how we can obtain a good estimate of the location of the sensor node based on small number of readings. Especially for indoor positioning systems, this is an assumption that has been extensively used. For example, in [13, 14], the position estimation is based on a *location fingerprint* $\mathbf{t} = [t_1\ t_2 \ldots t_N]$, where N is the number of beacons and t_i ($i = 1, \ldots, N$) is the mean value of the received signal strength from the i-th beacon over a certain time window. Note that t_i denotes the *measured* power $\hat{p_i}$ that appears in Equation 4. Hence the "measured power" location fingerprint $\mathbf{t}$ can easily be transformed to a location fingerprint $\mathbf{r}$ of "measured radii" by using Equation 4, since the reference values p_0 and d_0 are known. The number of samples that are used vary from 40 to 300 and obviously this number affects the accuracy of the computed location. Also, in [3, 9, 19, 38], experiments with various sample sizes are presented where the samples are used to compute certain features of the signal strength such as the standard deviation and the path loss exponent.

Suppose now that we use a sample of k readings from beacon b_i. We have a sequence of radii $\hat{r_{i1}}, \hat{r_{i2}}, \ldots, \hat{r_{ik}}$. Let $\bar{r_i}$, $\bar{s_i}$ denote the *unbiased* estimators of the value $\mathsf{E}[\hat{r_i}]$ and of the standard deviation $\sqrt{\mathsf{Var}(\hat{r_i})}$ of the underlying distribution of the measured radii $\hat{r_i}$ ($i = 1, \ldots, 3$) respectively. Then, it is known [36] that

$$\bar{r_i} = \frac{\sum_{j=1}^{k} \hat{r_{ij}}}{k} \tag{10}$$

and

$$\bar{s_i}^2 = \frac{\sum_{j=1}^{k} (\hat{r_{ij}} - \bar{r_i})^2}{k-1} \tag{11}$$

How can we relate these statistical values to the standard deviation σ_i of each beacon (the standard deviation appearing in Equation 5)? First we compute the mean $\mathsf{E}[\hat{r_i}]$ and the variance $\mathsf{Var}[\hat{r_i}]$ of the sample radii for beacon b_i. By Equation 5 we have

$$\hat{r_i} = r_i 10^{\frac{x}{10n}} \Rightarrow \mathsf{E}[\hat{r_i}] = r_i e^{\frac{c\sigma_i^2}{2}} \tag{12}$$

where $c = \frac{\ln^2(10)}{100n^2}$. Additionally,

$$\mathsf{Var}[\hat{r_i}] = r_i^2 \mathsf{Var}\left[10^{\frac{x}{10n}}\right] = r_i^2 (e^{c\sigma_i^2} - 1) e^{c\sigma_i^2} \tag{13}$$

Note that we used the fact that the expected value and the variance of a lognormal distribution are $e^{\sigma^2/2+\mu}$ and $(e^{\sigma^2+\mu} - 1)e^{\sigma^2+\mu}$, respectively, where μ and σ are the mean and the standard deviation of the related normal distribution. We recall that r_i is the actual distance of the sensor node from the beacon and for the Gaussian random variable x, it is $\mu = 0$. Also note that

$$\mathsf{E}[\hat{r_i}] \geq r_i \tag{14}$$

which indicates that the average of the readings is larger than the actual distance (this in turn implies that the circles defined by the beacons are expected to intersect in 6 points). Suppose now we receive signal from three beacons b_1, b_2, b_3 (we can assume

that b_1 is located at (0,0), b_2 is located at $(b, 0)$ and b_3 is located at (c, d)). If the sensor node is located at point (x, y), then (x, y) is a solution of the system of equations:

$$x^2 + y^2 = r_1^2 \quad (15)$$
$$(x-b)^2 + y^2 = r_2^2 \quad (16)$$
$$(x-c)^2 + (y-d)^2 = r_3^2 \quad (17)$$

By assuming now (this assumption will introduce the error of the described method) that the statistical values $\bar{r_i}$ and $\bar{s_i}$ computed by the sensor can substitute the theoretical ones we have the equations

$$\bar{r_i} = r_i\sqrt{e^{c\sigma_i^2}} \Rightarrow \bar{r_i}^2 = r_i^2 e^{c\sigma_i^2}$$

and

$$\bar{s_i} = r_i\sqrt{(e^{c\sigma_i^2}-1)e^{c\sigma_i^2}} \Rightarrow \bar{s_i}^2 = r_i^2(e^{c\sigma_i^2}-1)e^{c\sigma_i^2}$$

From these, we get an estimation $\bar{\sigma_i}^2$ of σ_i^2:

$$\bar{\sigma_i}^2 = \frac{1}{c}\ln\left[1 + \left(\frac{\bar{s_i}}{\bar{r_i}}\right)^2\right] \quad (18)$$

Finally, note that the estimate for the distance r_i^2 can be made by using

$$r_i^2 = \frac{\bar{r_i}^2}{e^{c\bar{\sigma_i}^2}} = \frac{\bar{r_i}^2}{1 + \left(\frac{\bar{s_i}}{\bar{r_i}}\right)^2} = \frac{\bar{r_i}^4}{\bar{r_i}^2 + \bar{s_i}^2} \quad (19)$$

Hence we have the following result that relates estimates of the actual distance and the standard deviation with reference to a beacon b_i with features of the lognormal distribution:

Theorem 4.1 *Suppose a sensor node reads k distance samples $\hat{r_{i1}}, \hat{r_{i2}}, \ldots, \hat{r_{ik}}$ from a beacon b_i that is modelled with the RSSI of path loss exponent n and standard deviation σ. If $\bar{r_i}$ is the sample mean and $\bar{s_i}$ is the sample standard deviation then we have the following:*

1. *The estimate of the square of the actual distance r_i^2 of the sensor node from beacon b_i is given by $\frac{\bar{r_i}^4}{\bar{r_i}^2 + \bar{s_i}^2}$.*
2. *The estimate of the square of the standard deviation σ_i^2 is given by $\frac{1}{c}\ln\left[1 + \left(\frac{\bar{s_i}}{\bar{r_i}}\right)^2\right]$, where $c = \frac{\ln^2(10)}{100n^2}$.*

□

Note that the above theorem indicates that the quality of estimation of the actual distance is heavily dependent on the estimation of the distribution of the measured radii.

4.1 Minimization-Based Estimation

In this section we develop a method for location estimation based on several samples. This method does not involve any complex calculations (such as square roots) which is very important to consider when we develop algorithms to be executed on sensor nodes, due to sensor's modest computing power. As we saw above, after completing sampling procedure, we derive estimates for r_1^2, r_2^2, r_3^2, given by Equation 19. Our aim is to formulate an objective function whose minimum will yield a good approximation of the sensor's location.

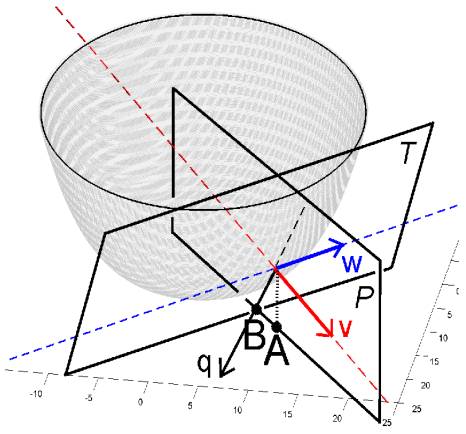


Figure 4: This is the plot of the function f and its tangent plane $\mathcal{T}$ at the point $(10, 10, f(10, 10))$ for $x_1 = y_1 = 0, x_2 = 3, y_2 = 0, x_3 = 1, y_3 = 4, r_1 = 1, r_2 = 2, r_3 = 4$. The vectors (directions) v, w, q are depicted. Vectors w and q belong to the same vertical plane. Point A is the initial point and point B is the new point. Two vertical planes are depicted, $\mathcal{T}$ (tangent plane) and $\mathcal{P}$.

This function should satisfy two conditions:

- It should be convex.
- Its derivatives should not include roots.

Suppose we have 3 beacons located at (x_1, y_1), (x_2, y_2), (x_3, y_3). Let $f(x, y)$ be the function

$$\begin{aligned} f(x,y) &= ((x-x_1)^2 + (y-y_1)^2 - r_1^2)^2 \\ &+ ((x-x_2)^2 + (y-y_2)^2 - r_2^2)^2 \\ &+ ((x-x_3)^2 + (y-y_3)^2 - r_3^2)^2 \end{aligned}$$

Note that if all 3 circles intersect at the same point (x_0, y_0), this function has minimum 0 at (x_0, y_0). Unfortunately, minimizing that function is not an easy task, if we are restricted on the available primitives. Hence we are going to use methods that are based on the gradient of the function. The good feature about such methods is that we can get to a point very close

to the minimum in a small number of computationally simple iterations. Indeed, let

$$\begin{aligned} \alpha(x,y) = \frac{\partial f(x,y)}{\partial x} = \\ = \quad & 4(x-x_1)((x-x_1)^2+(y-y_1)^2-r_1^2) \\ + \quad & 4(x-x_2)((x-x_2)^2+(y-y_2)^2-r_2^2) \\ + \quad & 4(x-x_3)((x-x_3)^2+(y-y_3)^2-r_3^2) \end{aligned} \tag{20}$$

and

$$\begin{aligned} \beta(x,y) = \frac{\partial f(x,y)}{\partial y} = \\ = \quad & 4(y-y_1)((x-x_1)^2+(y-y_1)^2-r_1^2) \\ + \quad & 4(y-y_2)((x-x_2)^2+(y-y_2)^2-r_2^2) \\ + \quad & 4(y-y_3)((x-x_3)^2+(y-y_3)^2-r_3^2) \end{aligned} \tag{21}$$

be the partial derivatives of f. Note that the above expressions are easily computable on a sensor node. The function $z = f(x,y)$ describes a convex solid surface with obvious definitions of "interior" and "exterior". Initially, we make a guess for our point (this is required by all steepest descent methods [34]). Suppose, for uniformity, we choose as our initial point (x_0, y_0) the centroid of the beacon triangle. We compute the vector v which is orthogonal to the tangent plane $\mathcal{T}$ and pointing toward the exterior. Hence

$$\mathsf{v} = \left[\begin{array}{ccc} \alpha(x_0,y_0) & \beta(x_0,y_0) & -1 \end{array} \right]^T$$

Let now $\mathcal{P}$ be the **vertical** plane containing v applied to $(x_0, y_0, f(x_0, y_0))$. Since $\mathcal{P}$ is a vertical plane, any normal vector w of $\mathcal{P}$ will have a zero z-component. Additionally, w is orthogonal to v and therefore may be chosen as

$$\mathsf{w} = \left[\begin{array}{ccc} -\beta(x_0,y_0) & \alpha(x_0,y_0) & 0 \end{array} \right]^T$$

We seek the vector q pointing towards the minimum of the function. Such vector belongs to $\mathcal{P}$ and is orthogonal to v (q is orthogonal both to v and w (see Figure 4)), i.e.,

$$\mathsf{q} = \left[\begin{array}{ccc} \alpha(x_0,y_0) & \beta(x_0,y_0) & \alpha^2(x_0,y_0)+\beta^2(x_0,y_0) \end{array} \right]^T$$

Now we compute the intersection point (x_0', y_0') of the line passing by $(x_0, y_0, f(x_0, y_0))$ which is collinear with the direction q and the xy-plane. The parametric equation of this line is

$$(x,y,z) = (x_0 + t\mathsf{q}_x, y_0 + t\mathsf{q}_y, f(x_0,y_0) + t\mathsf{q}_z)$$

for all $t \in \mathbf{R}$.

Then we have that the new point (x_0', y_0') is given by

$$\begin{aligned} x_0' &= x_0 - \frac{f(x_0,y_0)\alpha(x_0,y_0)}{\alpha^2(x_0,y_0)+\beta^2(x_0,y_0)} \\ y_0' &= y_0 - \frac{f(x_0,y_0)\beta(x_0,y_0)}{\alpha^2(x_0,y_0)+\beta^2(x_0,y_0)} \end{aligned} \tag{22}$$

program LOCALIZE

1: read $\hat{r_{1j}}, \hat{r_{2j}}, \hat{r_{3j}}$ for $j = 1, \ldots, k$ from the three beacons;
2: compute $\bar{r_i} = \frac{\sum_{j=1}^{k} \hat{r_{ij}}}{k}$ $(i = 1, 2, 3)$;
3: compute $\bar{s_i}^2 = \frac{\sum_{j=1}^{k} (\hat{r_{ij}} - \bar{r_i})^2}{k-1}$ $(i = 1, 2, 3)$;
4: $r_i^2 \leftarrow \frac{\bar{r_i}^4}{\bar{r_i}^2 + \bar{s_i}^2}$ $(i = 1, 2, 3)$;
5: choose an initial point (x_0, y_0);
6: **repeat**
7: $\quad (x, y) \leftarrow (x_0, y_0)$;
8: $\quad x_0' \leftarrow x_0 - \frac{f(x_0,y_0)\alpha(x_0,y_0)}{\alpha^2(x_0,y_0)+\beta^2(x_0,y_0)}$;
9: $\quad y_0' \leftarrow y_0 - \frac{f(x_0,y_0)\beta(x_0,y_0)}{\alpha^2(x_0,y_0)+\beta^2(x_0,y_0)}$;
10: $\quad (x_0, y_0) \leftarrow (x_0', y_0')$;
11: **until** $(\alpha(x_0', y_0')\alpha(x, y) < 0 \mid \beta(x_0', y_0')\beta(x, y) < 0)$
12: $\lambda \leftarrow 1000^{-m/100}$; ($m$ is the dimension of the grid)
13: **repeat**
14: $\quad (X, Y) \leftarrow (x, y)$;
15: $\quad (x', y') \leftarrow (x - \lambda\alpha(x, y), y - \lambda\beta(x, y))$;
16: $\quad (x, y) \leftarrow (x', y')$;
17: **until** $(\alpha(x', y')\alpha(X, Y) < 0 \mid \beta(x', y')\beta(X, Y) < 0)$
18: return (X, Y) as the final estimation;

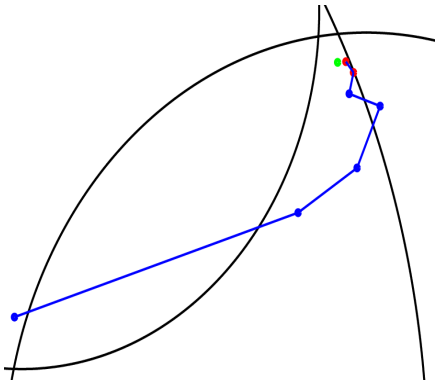
Figure 5: The pseudocode of the program to be executed on the sensor node. Note that the computations do not involve any complex operations, such as square roots, which decreases the computational effort on the sensor node and is also power efficient. The main computational work of this programs lies in gathering and processing the samples.

The relation between all these vectors can be seen in Figure 4. The described process gives a new point (x_0', y_0'). This point is expectedly closer to the point that corresponds to the minimum of f as we follow the direction of the gradient as long as the products $\alpha(x_0, y_0)\alpha(x_0', y_0') > 0$ and $\beta(x_0, y_0)\beta(x_0', y_0') > 0$. When this condition no longer holds, we have "overshot"; to remedy, we backtrack to the previous point referred here as (x, y) and apply a typical steepest descent method with very small rate λ. We therefore compute our new point (x', y') by setting

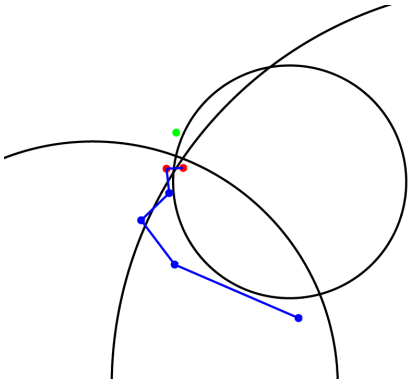
$$(x', y') = (x - \lambda\alpha(x, y), y - \lambda\beta(x, y)) \tag{23}$$

We continue this process until the gradients $\alpha(x, y)$, $\beta(x, y)$ change sign. At that point we stop and we report the final point as our estimation. Here we should

emphasize the fact that it is very important to take samples of adequate size. Taking samples implies a better behavior for function f, meaning that there would be only one minimum and therefore the algorithm will quickly converge to the minimum.



(a)



(b)

Figure 6: This is the path followed by the given program. The radii of the depicted circles are computed after the sampling using Equation 19. The first point of the path (bottom left in (a) and bottom right in (b)) is the centroid of the beacon triangle. Finally, the procedure stops after 6 iterations in (a) and 5 iterations in (b). The last two points of the paths are computed with the second repeat loop (steepest descent method). Note that the actual point (lightly shaded point) is very close to the final point of the paths.

As far as the value of the variable λ is concerned, this variable is chosen to be small enough and inversely proportional to the size of the grid since these features of λ force the second repeat loop of the algorithm to converge quickly. This has been observed in the experiments. For the experiments, the value of λ is equal to $1000^{-m/100}$.

In Figure 5 we give a short program for execution at the sensor node for location estimation. Note that this program only involves elementary primitives.

Illustrations of two executions of this program are given in Figure 6.

4.2 Complexity and Limitations

As far as the time complexity of the algorithm is concerned, the most expensive part is the sampling procedure and the computation of the estimates $\bar{r_i}^2$ and $\bar{s_i}^2$. These steps take time $O(k)$, so the running time of the algorithm is proportional to the number of samples. There is an obvious trade-off between accuracy and power consumption.

Also, the computation executed on the sensor nodes depends on the time the gradient methods take to converge, which is generally small for a well-behaved function. For the other parts of the algorithm there are closed formulas, so we can assume that they take time $O(1)$. We can also easily see that the exact number of multiplications needed by the presented program is $(7k + 8) + 10n_1 + 4n_2$, where is k is the size of the sample, n_1 is the number of iterations of the first repeat loop and n_2 is the number of iterations of the second repeat loop. Finally, the size of the code of the program (ROM) written in NesC [10] is 47K whereas the amount of memory (RAM) needed to execute this program in TOSSIM [17] (see Section 5) is 637K.

As far as the complexity of the closed formulas computation is concerned it is realistic to assume that the specified operations can be executed on a sensor node (essentially floating point operations). For example, there are micro-controllers, such as the ATMMega128L [2] (using a Harvard architecture with three 16-bit indirect memory access registers, accessed directly or using a constant offset) and MSP430 [35], which have very rich instruction sets (they support a wide range of of arithmetic instructions and many addressing modes). Finally, a hardware multiplier allows floating-point arithmetic to be carried out [20].

5 Simulation Results

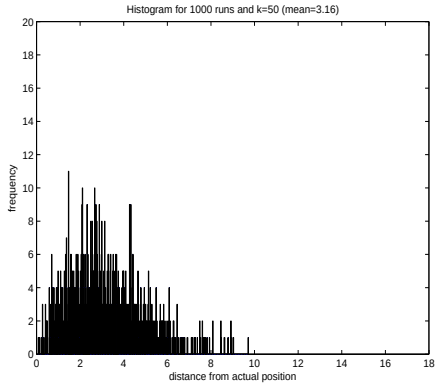
In this section we present extensive simulation results of our method. We have used TOSSIM [10, 17], a simulator of the TinyOS operating system for sensor networks. We executed our simulations in a square of area $m \times m$ cells, where $m = 50, 100, 200$. The 3 beacons are placed in positions that form a well conditioned triangle (well-conditioning is synonymous with the fact that the function $f(x, y)$ has a

Table 1: Simulation in TOSSIM for a 50×50 square: Execution time, average number of iterations of the program (n_1, n_2), localization error (d) and ratio $\frac{d}{m}$ for various samples sizes over 1000 runs.

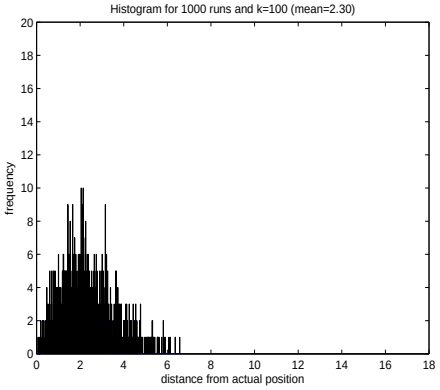
k	time (ms)	n_1	n_2	d	d/m
20	0.140	4.045	1.081	5.018	0.1003
40	0.230	4.226	1.061	3.774	0.0745
60	0.340	4.419	1.072	3.042	0.0608
80	0.450	4.592	1.051	2.554	0.0510
100	0.570	4.697	1.082	2.300	0.0460
120	0.650	4.781	1.056	2.181	0.0436
140	0.761	4.854	1.048	2.040	0.0408
160	0.901	4.825	1.056	1.890	0.0378
180	1.021	4.984	1.047	1.818	0.0363
200	1.111	4.953	1.055	1.766	0.0353
220	1.221	4.983	1.058	1.665	0.0333
240	1.331	4.990	1.056	1.574	0.0314
260	1.432	4.960	1.062	1.566	0.0313
280	1.532	4.991	1.066	1.533	0.0306
300	1.682	5.052	1.056	1.310	0.0262

single global minimum). Namely, the first beacon is placed at $(0, 0)$, the second beacon is placed at $(m, 0)$ and the third beacon is placed at $(m/2, 3m/4)$. The standard deviations of the three beacons $\sigma_1, \sigma_2, \sigma_3$ are set to 4 and the path loss exponent n is set to 2. We also recall that we set the variable λ that appears in Equation 23 equal to $1000^{-\frac{m}{100}}$, where m is the dimension of the grid. Finally the measured distance is computed using Equation 5. We count the execution time of the algorithm implemented in NesC [10] (that runs in TinyOS) and the average number of iterations of the repeat loops over 1000 runs. We also show the mean of the distance d between the actual point and the computed point. We use the ratio $\frac{d}{m}$ as a measure of the quality of the solution of the algorithm. Note that this metric was proposed in [37]. All the results for different number of samples and different sizes of grids are shown in Tables 1 ($m = 50$), 2 ($m = 100$), 3 ($m = 200$) (in Tables 1,2,3, k is the number of the samples, the time is counted in milliseconds (we count the exact time that the simulated processor in TOSSIM takes to execute this program), n_1 is the number of iterations of the first repeat loop, n_2 is the number of iterations of the second repeat loop and d is the mean of the distance between the actual point and the computed point).

The simulation results with TOSSIM (Tables 1,2,3) show that the sensor node can execute the algorithm in a small amount of time. This time is proportional to the number of samples we use each time which indicates that the sampling procedure dominates the execution time on the sensor node. Only up to 6 iterations (n_1+n_2) are enough to compute an estimation of the actual point and the quality of the estimation is dependent on the number of the samples. Additionally, note that for various grid sizes, the algorithm has a uniform behavior, since the ratio $\frac{d}{m}$ is similar for different sizes of the grid. Also, in all cases, the solution we get is better for larger sizes of samples.



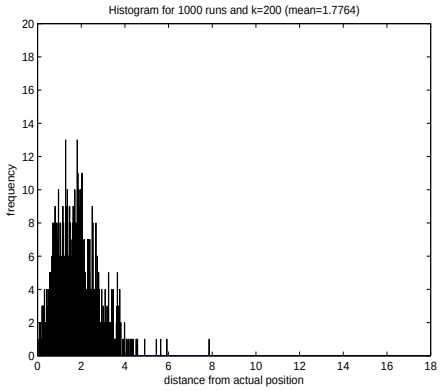
(a) Histogram for the distance from the actual position for $k = 50$.



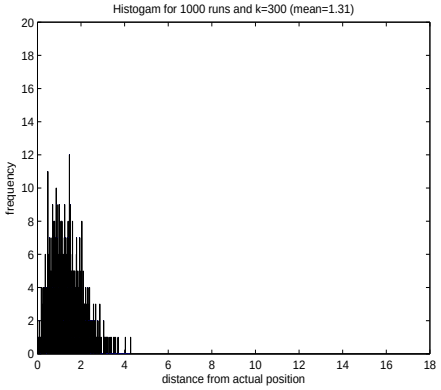
(b) Histogram for the distance from the actual position for $k = 100$.

Figure 7: Histograms for 1000 runs. Note that the mean distance from the actual position decreases as we increase the size of the sample.

We also show the probability distribution of the distance of the computed point from the actual point derived for 1000 runs (Figures 7,8) and for $m = 50$. We can see the plot of the distribution of the error of the estimation (which we define as the distance of

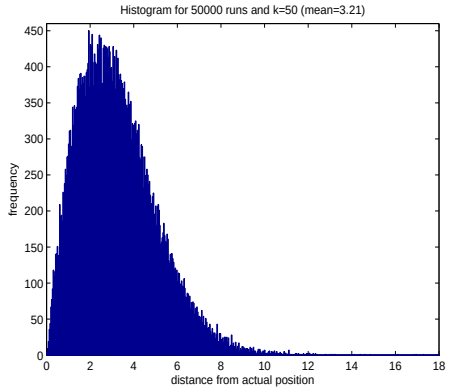


(a) Histogram for the distance from the actual position for $k = 200$.

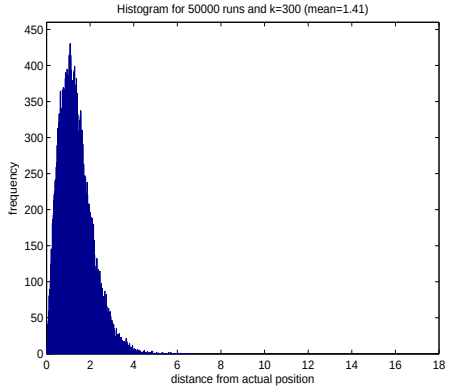


(b) Histogram for the distance from the actual position for $k = 300$.

Figure 8: Histograms for 1000 runs. Note that the mean distance from the actual position decreases as we increase the size of the sample.



(a) Histogram for the distance from the actual position for $k = 50$.



(b) Histogram for the distance from the actual position for $k = 300$.

Figure 9: Histograms for 50000 runs. Note that the resulting probability distribution appears to be log-normal.

the computed point from the actual point) for various sample sizes ($k = 50, 100, 200, 300$). It is clear that as the sample size increases we get an increasingly better approximation of the actual location. The mean of the error for these measurements is no more than 3.16 (this is achieved for $k = 50$ (Figure 7(a))) and it goes down to 1.31 for $k = 300$ (Figure 8(b)). Also, it is interesting that the distribution of the error (for a much larger number of runs) seems to follow a log-normal distribution (Figure 9).

Finally, we give a table (Table 4) that compares existing work on location estimation algorithms. We see that our method is competitive with previous work. We present the average localization error d, the area A of the field where the experiments are executed and the ratio $\frac{d}{\sqrt{A}}$. We use as a comparison measure the quantity $\frac{d}{\sqrt{A}}$, which for our algorithm is bounded by 0.1 (this is what we get for the smallest number of samples $k = 20$). However, for a number of samples $k = 300$ we can get even smaller values (for example for $k = 300$ and for an area 1000×1000 we get $\frac{d}{\sqrt{A}} = 0.026$). From Table 4, we see that our method gives always better or as good results as the results obtained by the existing methods. Also, if we slightly increase the number of the samples we use, we get very good results and the ratio drops substantially (for example for $k = 60$ we get a ratio $\frac{d}{\sqrt{A}} = 0.06$).

Table 2: Simulation in TOSSIM for a 100×100 square: Execution time, average number of iterations of the program (n_1, n_2), localization error (d) and ratio $\frac{d}{m}$ for various samples sizes over 1000 runs.

k	time (ms)	n_1	n_2	d	d/m
20	0.130	3.670	1.142	9.986	0.0986
40	0.240	3.817	1.046	7.634	0.0763
60	0.360	3.884	1.010	6.760	0.0676
80	0.470	3.910	1.010	6.140	0.0614
100	0.570	3.950	1.008	5.740	0.0574
120	0.691	3.960	1.006	5.352	0.0535
140	0.801	3.969	1.002	5.310	0.0531
160	0.921	3.975	1.002	5.002	0.0500
180	1.001	3.985	1.000	4.802	0.0480
200	1.131	3.984	1.001	4.689	0.0468
220	1.221	3.933	1.058	4.680	0.0468
240	1.362	3.983	1.000	4.503	0.0450
260	1.462	3.994	1.000	4.454	0.0445
280	1.592	3.994	1.000	4.441	0.0444
300	1.692	3.996	1.001	4.360	0.0436

Table 3: Simulation in TOSSIM for a 200×200 square: Execution time, average number of iterations of the program (n_1, n_2), localization error (d) and ratio $\frac{d}{m}$ for various samples sizes over 1000 runs.

k	time (ms)	n_1	n_2	d	d/m
20	0.120	3.640	2.736	19.977	0.0998
40	0.240	3.820	2.323	14.957	0.0747
60	0.360	3.878	2.219	13.093	0.0654
80	0.450	3.909	2.120	11.575	0.0568
100	0.560	3.938	2.075	10.821	0.0541
120	0.670	3.953	2.006	10.030	0.0501
140	0.791	3.955	2.029	9.317	0.0460
160	0.911	3.960	2.024	8.979	0.0440
180	1.001	3.977	1.946	8.564	0.0420
200	1.111	3.983	1.946	8.383	0.0410
220	1.201	3.984	1.999	8.347	0.0410
240	1.361	3.983	1.963	7.998	0.0399
260	1.462	3.994	1.937	7.894	0.0394
280	1.582	3.994	1.921	7.852	0.0392
300	1.712	3.996	1.919	7.774	0.0387

Note that the previous methods use sometimes more than three beacon nodes (see for example [25]) where $O(m)$ beacons are placed around the area of localization for an $m \times m$ grid, whereas we use only 3 beacon nodes.

Table 4: Comparison of existing work in location estimation based on RSSI. In each row, we display the bibliographic reference and the respective average localization error (d), the size of the area of the experiments A and finally the ratio $\frac{d}{\sqrt{A}}$.

reference	d (m)	**area** A (m^2)	$d/\sqrt{A}$
[3]	3	22.5×45.5	0.090
[29]	3	16×40	0.118
[4]	4	35×40	0.107
[26]	7.62	13.71×32	0.360
[1]	3	500	0.130
[27]	6	60×60	0.100
[5]	1.83	10×10	0.183
[32]	0.8	6×6	0.130
[18]	13	18751	0.094
[39]	10	26×49	0.280
20 samples	5.018	50×50	$\simeq 0.1$
60 samples	3.042	50×50	$\simeq 0.06$

6 Conclusions and Future Work

In this paper, we have analyzed the most commonly used measurement model for location estimation in sensor networks. Given a normal distribution for the error in dBm, we show how to derive the correct probability distribution for the position of the sensor node. The computation of the probability distribution is based only on one measurement. We verify that if we try to estimate the unknown position based only on this probability distribution, the resulting error may be unacceptably large.

We have therefore presented a theoretical analysis of sampling the measurements for location estimation. By computing statistics on the gathered samples by only three beacons, we can produce a solution based on a minimization method. We propose a simple algorithm that can easily be executed on the sensor node; its complexity, for a constant number of beacons, is proportional to the size of the sample. It is evident that there are a lot of trade-offs in location estimation in sensor networks. If more accuracy is desired, one has to deploy more beacons or to use more samples. This has an impact on the energy consumption of the sensor. The energy-optimal case is the case where only 3 beacons are deployed and make an estimation of the actual point based on the probability distribution computed by taking into consideration only one measurement. This, however, gives unacceptable errors. Additionally, executing computations with the computed probability function is unrealistic, since it involves extremely complex formulas.

Hence were we to depend on few measurements, offline computations have to be made to provide the sensor with the necessary data; this immediately raises a storage problem, since the storage capacity of a sensor node is limited. On the other hand, one can use more samples, an approach that, however, increases energy consumption.

As future work, we envision an extended study of these trade-offs and the implementation of the proposed algorithm on real motes. Additionally, we can certainly extend the analysis to more than three beacons and evaluate the corresponding improvements.

References

[1] C. Alippi and G. Vanini. A RSSI-based and calibrated centralized localization technique for wireless sensor networks. In *Proc. IEEE Int. Conference on Pervasive Computing and Communications Workshops (PERCOMW)*, pages 301–306, 2006.

[2] Atmel corporation. ATM128 datasheet, revised 2461-09/03, 2003.

[3] P. Bahl and V. N. Padmanabhan. RADAR: An in-building RF-based user location and tracking system. In *Proc. IEEE Conference on Computer Communications (INFOCOM)*, pages 775–784, 2000.

[4] M. Brunato and R. Battiti. Statistical learning theory for location fingerprinting in wireless LANs. *Computer Networks*, 47(6):825–845, 2005.

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

[6] S. Capkun and J.-P. Hubaux. Secure positioning of wireless devices with application to sensor networks. In *Proc. IEEE Conference on Computer Communications (INFOCOM)*, pages 1917–1928, 2005.

[7] B. Dil, S. Dulman, and P. Havinga. Range-based localization in mobile sensor networks. In *Proc. European Wireless Sensor Networks (EWSN)*, pages 164–179, 2006.

[8] L. Doherty, K. S. J. Pister, and L. E. Ghaoui. Convex optimization methods for sensor node position estimation. In *Proc. IEEE Conference on Computer Communications (INFOCOM)*, pages 1655–1663, 2001.

[9] D. B. Faria. Modeling signal attenuation in IEEE 802.11 wireless LANs - vol. 1. Technical Report TR-KP06-0118, Stanford University, 2005.

[10] D. Gay, P. Levis, R. von Behren, M. Welsh, E. Brewer, and D. Culler. The nesC language: A holistic approach to networked embedded systems. In *Proc. ACM Conference on Programming Language Design and Implementation (PLDI)*, pages 1–11, 2003.

[11] T. He, C. Huang, B. M. Blum, J. A. Stankovic, and T. Abdelzaher. Range-free localization schemes for large scale sensor networks. In *Proc. of the Int. Conference on Mobile Computing and Networking (MOBICOM)*, pages 81–95, 2003.

[12] L. Hu and D. Evans. Localization for mobile sensor networks. In *Proc. of the Int. Conference on Mobile Computing and Networking (MOBICOM)*, pages 45–57, 2004.

[13] K. Kaemarungsi and P. Krishnamurthy. Modeling of indoor positioning systems based on location fingerprinting. In *Proc. IEEE Conference on Computer Communications (INFOCOM)*, 2004.

[14] K. Kaemarungsi and P. Krishnamurthy. Properties of indoor received signal strength for WLAN location fingerprinting. In *Proc. Int. Conference on Mobile and Ubiquitous Systems (MOBIQUITOUS)*, pages 14–23, 2004.

[15] B. Karp and H. T. Kung. GPSR: Greedy perimeter stateless routing for wireless networks. In *Proc. of the Int. Conference on Mobile Computing and Networking (MOBICOM)*, pages 243–254, 2000.

[16] S.-P. Kuo, Y.-C. Tseng, F.-J. Wu, and C.-Y. Lin. A probabilistic signal-strength-based evaluation methodology for sensor network deployment. In *Proc. Int. Conference on Advanced Information Networking and Applications (AINA)*, pages 319–324, 2005.

[17] P. Levis, N. Lee, M. Welsh, and D. Culler. TOSSIM: accurate and scalable simulation of entire TinyOS applications. In *Proc. Int. Conference on Embedded Networked Sensor Systems (SENSYS)*, pages 126–137, 2003.

[18] K. Lorincz and M. Welsh. MoteTrack: A robust, decentralized approach to RF-based location tracking. In *Proc. Int. Workshop on Location and Context-Awareness (LOCA)*, 2005.

[19] D. Lymberopoulos, Q. Lindsey, and A. Savvides. An empirical characterization of radio signal strength variability in 3-d IEEE 802.15.4 networks using monopole antennas. In *Proc. European Wireless Sensor Networks (EWSN)*, pages

326–341, 2006.

[20] C. Lynch and F. O. Reilly. Processor choice for wireless sensor networks. In *Proc. ACM Workshop on Real-World Wireless Sensor Networks (REALWSN)*, pages 52–68, 2005.

[21] R. Nagpal, H. E. Shrobe, and J. Bachrach. Organizing a global coordinate system from local information on an ad hoc sensor network. In *Proc. Int. Conference on Information Processing in Sensor Networks (IPSN)*, pages 333–348, 2003.

[22] A. Nasipuri and K. Li. A directionality based location discovery scheme for wireless sensor networks. In *Proc. ACM Int. Workshop on Wireless Sensor Networks and Applications (WSNA)*, pages 105–111, 2002.

[23] D. Niculescu and B. R. Badrinath. Ad hoc positioning system (APS) using AOA. In *Proc. IEEE Conference on Computer Communications (INFOCOM)*, 2003.

[24] D. Niculescu and B. Nath. DV based positioning in Ad Hoc networks. *Telecommunication Systems*, 22:267–280, 2003.

[25] H. Ochi, S. Tagashira, and S. Fujita. A localization scheme for sensor networks based on wireless communication with anchor groups. In *Proc. Int. Conference on Parallel and Distributed Systems (ICPADS)*, pages 299–305, 2005.

[26] P. Prasithsangaree, P. Krishnamurthi, and P. K. Chrysanthis. On indoor position location with wireless LANs. In *Proc. IEEE Int. Symposium on Personal, Indoor, and Mobile Radio Communications*, September 2002.

[27] V. Ramadurai and M. L. Sichitiu. Localization in wireless sensor networks: A probabilistic approach. In *Proc. Int. Conference on Wireless Networks (ICWN)*, pages 275–281, 2003.

[28] T. S. Rappaport and T. Rappaport. *Wireless Communications: Principles and Practice*. Prentice Hall, 2nd edition, 2001.

[29] T. Roos, P. Myllymaki, H. Tirri, P. Misikangas, and J. Sievanen. A probabilistic approach to WLAN user location estimation. *International Journal of Wireless Information Networks*, 9(3):155–166, 2002.

[30] A. Savvides, C.-C. Han, and M. B. Strivastava. Dynamic fine-grained localization in Ad-Hoc networks of sensors. In *Proc. of the Int. Conference on Mobile Computing and Networking (MOBICOM)*, pages 166–179, 2001.

[31] A. Savvides, H. Park, and M. B. Srivastava. The bits and flops of the n-hop multilateration primitive for node localization problems. In *Proc. ACM Int. Workshop on Wireless Sensor Networks and Applications (WSNA)*, pages 112–121, 2002.

[32] X. Shen, Z. Wang, P. Jiang, R. Lin, and Y. Sun. Connectivity and RSSI based localization scheme for wireless sensor networks. In *Advances in Intelligent Computing*, pages 578–587, 2005.

[33] M. Sichitiu and V. Ramadurai. Localization of wireless sensor networks with a mobile beacon. In *Proc. IEEE Int. Conference on Mobile Ad-Hoc and Sensor Systems (MASS)*, pages 177–183, 2004.

[34] J. A. Snyman. *Practical Mathematical Optimization: An Introduction to Basic Optimization Theory and Classical and New Gradient-Based Algorithms*. Springer Publishing, 2005.

[35] TEXAS INSTRUMENTS. MSP430C13x1 datasheet, revised sept. 04., 2004.

[36] D. Wackerly, W. Mendenhall, and R. Scheaffer. *Mathematical Statistics with Applications*. Duxbury Advanced Series, 6th edition edition, 2002.

[37] K. Whitehouse, C. Karlof, and D. Culler. A practical evaluation of radio signal strength for ranging-based localization. In *ACM Mobile Computing and Communications Review (MC2R)*, 2007.

[38] Z. Xiang, S. Song, J. Chen, H. Wang, J. Huang, and X. Gao. A wireless LAN-based indoor positioning technology. *IBM J. Res. Dev.*, 48(5/6):617–626, 2004.

[39] K. Yedavalli, B. Krishnamachari, S. Ravula, and B. Srinivasan. Ecolocation: a sequence based technique for rf localization in wireless sensor networks. In *Proc. Int. Conference on Information Processing in Sensor Networks (IPSN)*, page 38, 2005.

[40] Y. Yu, R. Govindan, and D. Estrin. Geographical and energy aware routing: A recursive data dissemination protocol for wireless sensor networks. Technical Report UCLA/CSD-TR-01-0023, UCLA Computer Science Department, 2001.