The Geo-n Localization Algorithm

Heiko Will, Thomas Hillebrandt, Marcel Kyas
Freie Universität Berlin
AG Computer Systems & Telematics
Berlin, Germany
Email: {heiko.will,thomas.hillebrandt,marcel.kyas}@fu-berlin.de

***Abstract*—We introduce Geo-n, a highly precise, distance-based, general-purpose localization algorithm for use in cluttered and indoor environments, where the estimated distances between an unlocalized node and anchor nodes may contain large errors due to NLOS signal propagation. Geo-n is very resilient to outliers and to a wide range of geometrical constellations of the nodes. Geo-n uses a two stage filtering technique to obtain the most representative intersection points between every pair of circles induced by anchor coordinates and distance measurements and uses these to estimate the position of unlocalized nodes. If no intersection exists between two circles, Geo-n approximates one to further improve localization accuracy.**

We compare Geo-n to LLS, NLLS, AML, Min-Max, and ICLA using simulations and real world experiments, and show analysis of the spatial distribution of position errors. Results demonstrate that Geo-n outperforms the other algorithms. Like NLLS, its spatial error distribution is very homogeneous. However, Geo-n is much more robust and achieves lower average position error while retaining reasonable computational complexity.

***Index Terms*—Indoor Localization, Localization Algorithm, Radio Runtime Measurement, Distance Measurement, Wireless Sensor Networks**

I. Introduction

The precise spatial localization of a signal source is not limited to computer science. A broad variety of other sciences rely on localization algorithms. Psychologists want to detect the precise spatial source of electric impulses in the human brain, biologists want to track the position of birds equipped with small sensor nodes and geologists want to detect the source of an earthquake using seismic waves. Most of these applications use the same principles: based on the measurement of a physical value, the distance between the target and some fixed points (anchors) is estimated and then the position of the target is calculated with a localization algorithm. The main differences between the variety of existing localization algorithms are the handling of distance measurement errors and their robustness to the geometrical constellation of unlocalized nodes and anchors.

The context of our exposition is indoor localization. We assume that anchor nodes know their exact position and the unlocalized node can measure the distances to at least three anchor nodes. These measurements consist of the real distance between the devices and an unknown error. This error may have many different sources, depending on the measurement system and the structure of the building in which the devices are located. We sum up all these different errors and call them *distance measurement errors.* The distance between the real node position and the calculated node position is called *position error.* Previously, we have shown that the position error not only depends on the distance measurement errors but also heavily depends on the geometrical constellation of node and anchors [1]. This observation is important for the evaluation of lateration algorithms because using the average position error as the only metric can lead to misleading conclusions. For example, an algorithm with a remarkably low average position error could perform very bad in real world deployments if the low position error is mainly achieved in regions out of the convex hull of the anchors but the anchors are placed on the walls of a building and so the hull can never be left by the node. We present an evaluation of the average position error resulting in several large real world deployments for all compared algorithms introduced in Section II. We also present the spatial distribution of the position error for all algorithms which we have calculated with the LS^2 [1] simulation engine in Section IV.

The main contribution is the presentation of the Geo-n localization algorithm in Section III, which is based on a clustering like selection scheme of circle intersections. The main design goals for Geo-n were robustness to outliers in distance measurement errors, which mainly result from non-line-of-sight (NLOS) distance estimations, and a homogeneous spatial distribution of the position error. In addition, the computational complexity of the algorithm should be low enough for real time localization for Wireless Sensor Networks (WSNs) with low power nodes to be possible.

II. Related Work

Several algorithms for distance-based position estimation have been published in the last decades. Some have been focused on low computational complexity for use in sensor networks and most recent publications have focused on resilience to measurement errors or filtering for use in indoor scenarios. These algorithms are designed to reduce the effect of distance measurement errors, e.g. ones caused by the complicated indoor multi-path propagation, low signal-to-noise ratio (SNR), severe multi-path effects, reflection and link failures, and to improve the position error [2]–[6]. These algorithms include iterative methods, which use gradient descent or Newton method to calculate an estimated position. Grid-scan methods [7], [8] divide the target field into several cells and are using voting based methods to select a cell as an estimated position. Refined geometry relationship [5], [9] obtains the target relative position rather than actual position, and the method is still based on the range based measurements, in which the distance measurement noise still causes estimation errors.

We compare out Geo-n algorithm to well known and to new algorithms. Three of them are well known algorithms and are often used for performance comparison when proposing a new localization algorithm: Multilateration using non-linear least squares (NLLS) or linear least squares (LLS) [4], [6] and Min-Max algorithm [10], [11]. The two new algorithms also use the intersection points of circles for position estimation: Adapted Multi-Lateration (AML) [12] and Iterative Clustering-based Localization Algorithm (ICLA) [13].

1) Nonlinear Least Squares Multilateration: Given N anchor nodes with fixed positions at $b_i = (x_i, y_i)$ for $i = 1, 2, \ldots, N$ and possibly noisy range measurements r_i from these nodes to a non-anchor node located at $u = (x, y)$, multilateration finds the most likely position of the unknown node, denoted by $\hat{u}$. From this information we write a system of equations:

$$\begin{aligned}(x-x_1)^2+(y-y_1)^2 &= {r_1}^2\\ (x-x_2)^2+(y-y_2)^2 &= {r_2}^2\\ &\vdots\\ (x-x_N)^2+(y-y_N)^2 &= {r_N}^2\end{aligned} \tag{1}$$

This problem is usually solved by using a *least squares* method, that is, minimizing the sum of the squared residuals between the observed ranges r_i and the estimated distance $||u - b_i||$:

$$\hat{u} = \operatorname*{argmin}_u \sum_{i=1}^{N} \big(||u-b_i|| - r_i\big)^2 \tag{2}$$

The minimization problem can be solved by using any of the Newton type optimization algorithms [14]. These start from an initial guess at the solution and then iterate to gradually improve the estimated position until a local minimum of the objective function in Eq. (2) is found. However, there is a non-negligible probability of falling into a local minimum of the error surface when solving Eq. (2). Therefore, to find an estimate close to the global minimum, LS must run several times with different initial starting points, which is expensive in terms of computing overhead. For instance, our reference implementation uses two starting points, the LLS solution and the centroid of all anchor coordinates.

2) Linear Least Squares Multilateration: The *nonlinear least squares* problem can be linearized by subtracting one of the equations given in Eq. (1) from the remaining $N-1$ equations. In matrix notation, the linear system can be expressed as $Au = b$ and can be solved by the LS method to provide an estimated location, as given by the closed form solution shown in Eq. (3) (i.e., normal equations).

$$\hat{u} = \left(A^T A\right)^{-1} A^T b \tag{3}$$

with:

$$A = \begin{bmatrix} x_1 - x_N & y_1 - y_N \\ x_2 - x_N & y_2 - y_N \\ \vdots & \vdots \\ x_{N-1} - x_N & y_{N-1} - y_N \end{bmatrix}$$

and

$$b = \frac{1}{2}\begin{bmatrix} x_1^2 - x_N^2 + y_1^2 - y_N^2 + r_N^2 - r_1^2 \\ x_2^2 - x_N^2 + y_2^2 - y_N^2 + r_N^2 - r_2^2 \\ \vdots \\ x_{N-1}^2 - x_N^2 + y_{N-1}^2 - y_N^2 + r_N^2 - r_{N-1}^2 \end{bmatrix}$$

3) Adapted Multi-Lateration: Similar to multilateration and Geo-n, AML tries to estimate the position of an unlocalized node using circle intersections. AML consists of three steps: intersection and elimination, first estimation and refinement. In the first step, two intersecting circles are arbitrarily chosen. These circles may intersect at one or two points. If there is more than one point, the point with the larger distance to the third anchor is eliminated. In the first estimation step, the previously computed intersection point is moved to the middle of the line connecting it with the closest point of the third anchor's circle. This is done to compensate the errors introduced by range measurements. The calculation is done using the resemblance of triangles. In the last step, the position can be further refined. Therefore, the anchors that were not used in the previous steps are added to the position estimation process with the same principle utilized in the second step.

4) Min-Max: The Min-Max algorithm, also known as Bounding Box algorithm, is a simple and straightforward method in contrast to the quite expensive number of floating point operations required by LLS or NLLS. The main idea is to build a square (bounding box) given by $[x_i - r_i, y_i - r_i] \times [x_i + r_i, y_i + r_i]$ around each anchor node using its location (x_i, y_i) and distance estimate r_i, and then to calculate the intersection of these squares. The final position of the unlocalized node is approximated by the center of the intersection box, which is computed by taking the maximum of all coordinate minimums and the minimum of all maximums:

$$\begin{aligned}&[\max_{1\le i\le N}(x_i - r_i), \max_{1\le i\le N}(y_i - r_i)]\times\\ &[\min_{1\le i\le N}(x_i + r_i), \min_{1\le i\le N}(y_i + r_i)]\end{aligned} \tag{4}$$

5) Iterative Clustering-based Localization Algorithm: In ICLA, node localization is done by clustering intersection points, which is claimed to be resistant to RSSI errors. The algorithm consists of three main steps. In the first step all intersection points between every two circles centered at the anchors coordinates and with radii equal to the estimated distances are generated. These intersection points cluster around the unlocalized node. In the second step the iterative clustering model (ICM) is applied to get the most representative intersection points for localization. The final step of the algorithm

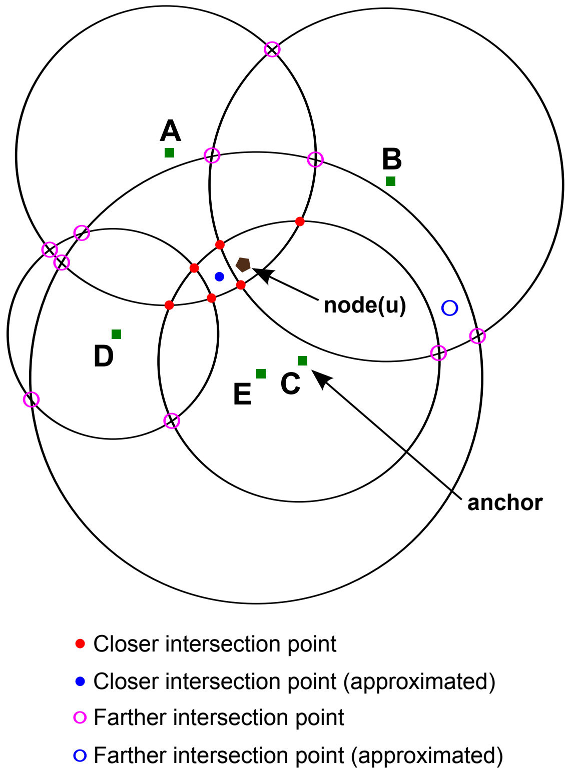
A
B
D
E
C
node(u)
anchor
● Closer intersection point
● Closer intersection point (approximated)
○ Farther intersection point
○ Farther intersection point (approximated)

Fig. 1. Motivation for using clustering like approach.

calculates the position of the unlocalized node by taking the centroid of all intersection points of the biggest group that ICM has produced. ICM is the central part of the algorithm. Here, all intersection points are iteratively moved towards their moving direction and merged if a collision occurs. The collision area is a circular area with the radius equal to the size of the moving step. Points with bigger weight exert a larger attracting force to other points and influence their moving direction. Initially, all points have the same weight. At the end of the procedure, all points are classified into several different clusters according to the left points.

III. The Geo-n Localization Algorithm

Let u be an unlocalized node and $A_u = \{a_1, a_2, \ldots, a_N\}$ the set of $N \geq 3$ anchor nodes to which u estimes its distance. Let $C = \{C_1, C_2, \ldots, C_N\}$ be a set of circles induced by anchor coordinates $a_i \in A_u$ and distance measurement r_i (between u and a_i) as the radius. For each u, Geo-n makes a selection of all pairwise intersection points between the circles in C as detailed later. This way, only intersection points that are probably beneficial for the localization task are kept. We motivate the use of this approach using Fig. 1.

Figure 1 shows the unlocalized node u and all anchor nodes in $A_u = \{A, B, C, D, E\}$. The black circles have a radius equal to error-prone distance measurements. The distance to anchor D, for example, is measured a bit too short (its distance estimate is less than the true distance from u) while the rest of the anchors are measured too long (their distance estimates are greater than their true distances from u). Especially anchor E has a very large positive distance measurement error. Without any measurement error, all circles would intersect at node u. However, we always have measurement errors in the real world that result from the measuring processes using a noisy channel and the NLOS error deriving from obstacles in the direct paths. Thus, it is possible that no circle passes through u. Two distinct circles intersect in at most two points. Due to the geometry, in most cases one intersection point is close to the location of u. Occasionally, both intersection points are equidistant to it. As a consequence, the density of intersection points is usually highest close to node u even when there is a large measurement error. This is indicated by the seven solid red and blue points in Fig. 1, whereas the second largest cluster only consists of three points. Finding a good approximation of this cluster and using it for localization is the central problem solved in the Geo-n localization algorithm.

Geo-n uses different filtering methods to remove intersection points that do not contribute to the localization or are suspected to increase the positioning error. Thus, it can locate u with high accuracy and is resilient to a large number of measurement error. We describe Geo-n in Algorithm 1. We write C_i for the circle centered at point a_i with radius r_i and $\overline{D}_i$ to describe the closed disk bounded by C_i. The concatenation of two lists is denoted by $\oplus$.

In lines 1 to 2 the variables IP (the list of all intersection points) and AIP (the list of all approximated intersection points) are initialized to empty lists. Lines 3 to 12 populate IP with all intersection points between every pair of circles induced by the N anchor coordinates a_i and distance measurements r_i and AIP with approximated intersection points if the circles do not intersect or one circle is contained within the other.

Approximating Intersections

In line 9 an intersection point is approximated if there is no real intersection point. The two possibilities are displayed in Fig. 2a and Fig. 2b. In both cases, it is still possible to approximate one intersection point: it is the middle of the line connecting points P_1 and P_2 (highlighted in blue). Points P_1 and P_2 are the closest intersection points of the two circles and the line connecting anchors A and B and where P_1 and P_2 lie on different circles.

As a result of lines 3 to 12, the number of pairwise intersections between the corresponding circles is not greater than N^2. For each pair of circles there exist at most two intersection points and at least one (approximated) intersection point, hence $\frac{N\cdot(N-1)}{2} \leq |IP \oplus AIP| \leq N \cdot (N-1)$.

Line 13 and lines 22 to 31 specify a two stage filter mechanism to the generated intersection points of the previous step. Line 13 only keeps those intersection points of IP contained in at least $N-2$ closed disks $\overline{D}_i$ ($i \in \{1, \ldots, N\}$). From now on, we refer to this as Filter 1 (F1). Most of the farther intersection points (see Fig. 1) are removed as a result of applying F1, which may also reduce the total runtime of the algorithm.

Algorithm 1 Geo-n

```
Require: ranges $r_i$ and anchor positions $a_i$, $i = 1, \dots, N$,
    $N \geq 3$, $\forall i, j : a_i \neq a_j \lor i = j$;
Ensure: the estimated position $(\hat{x}, \hat{y})$
 1: $IP \leftarrow nil$;                         ▷ Intersection points
 2: $AIP \leftarrow nil$;            ▷ Approximated intersection points
 3: for $i \leftarrow 1$ to $N - 1$ do
 4:     for $j \leftarrow i + 1$ to $N$ do
 5:         $I \leftarrow C_i \cap C_j$;
 6:         if $I \neq nil$ then
 7:             $IP \leftarrow IP \oplus I$;
 8:         else
 9:             $AIP \leftarrow AIP \oplus$ [Approximate an intersection
            point between $C_i$ and $C_j$];
10:         end if
11:     end for
12: end for
13: $IP \leftarrow [x \in IP \mid \left|\{i \mid x \in \overline{D}_i\}\right| \geq N - 2]$;
14: $WIP \leftarrow nil$;               ▷ Weighted intersection points
15: for all $x \in IP$ do
16:     $WIP \leftarrow WIP \oplus [(x, W_{IP})]$;
17: end for
18: for all $x \in AIP$ do
19:     $WIP \leftarrow WIP \oplus [(x, W_{AIP})]$;
20: end for
21: if $|WIP| \geq 3$ then
22:     $distance[\,|WIP|\,] \leftarrow 0$;      ▷ Set all distances to zero
23:     for $i \leftarrow 1$ to $|WIP|$ do
24:         for $j \leftarrow 1$ to $|WIP|$ do
25:             if $i \neq j$ then
26:                 $distance[i] \leftarrow distance[i] + \|WIP_i -$
                $WIP_j\|$;
27:             end if
28:         end for
29:     end for
30:     $median \leftarrow$ calculate median of $distance$;
31:     $WIP \leftarrow [x \in WIP \mid distance$ of $x \leq median]$;
32: end if
33: $(\hat{x}, \hat{y}) \leftarrow$ weighted centroid of $WIP$;
34: return $(\hat{x}, \hat{y})$
```

In our test runs described in Section V, around 60% of the intersection points are removed on average. This leads to a significant performance increase in terms of execution time (between 20% and 60% depending on the number of anchors N). Note, that approximated intersection points are not affected by this filter. Keeping them will almost always result in higher localization accuracy, especially when the number of real intersection points tends to be small.

Before the second stage of the filter is applied, lines 14 to 20 merge IP and AIP into a new list named WIP containing weighted intersection points for the final step of Geo-n. Real intersection points are weighted with weight W_{IP} and approximated ones with weight W_{AIP}. In our experiments, assigning

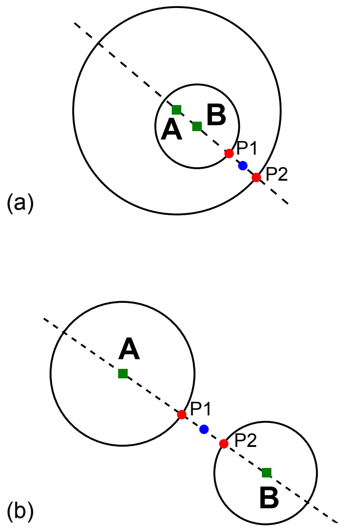

Fig. 2. Approximated intersection points (drawn in blue) when there is no intersection of the circles.

equal weight showed the best results whereas in simulations weighting real intersection points three times higher than approximated ones gave the best performance.

Lines 22 to 31 implement a median filter based on the inter-point distances to further remove useless intersection points not improving localization accuracy. First, for each point in WIP, the sum of distances to all other points is calculated and stored in the array $distance$. Next, line 30 obtains the median of the $distance$ array. Then the points whose value in $distance$ is larger than the $median$ are eliminated in line 31. Note that this median filter is only applied if the number of intersection points in WIP is not smaller than 3 because otherwise filtering would only leave one weighted intersection point that is probably farther from the true position than the result using the whole list. From now on, we refer to this as Filter 2 (F2). In total, the combination of both filters in this order is beneficial for localization accuracy. Applying the first filter leads to a performance increase of 13%-20% in our test runs and simulations compared to using the second filter only.

Finally the position of the unlocalized node is estimated as the weighted centroid of the remaining intersection points, as expressed in Eq. (5).

$$(\hat{x}, \hat{y}) = \left(\frac{\sum_{i=1}^{|WIP|} w_i \cdot x_i}{\sum_{i=1}^{|WIP|} w_i}, \frac{\sum_{i=1}^{|WIP|} w_i \cdot y_i}{\sum_{i=1}^{|WIP|} w_i} \right) \qquad (5)$$

The run-time complexity of Geo-n is in $\mathcal{O}(N^4)$.

Proof: The calculation of all intersection points in lines 3 to 12 has a runtime in $\mathcal{O}(N^2)$. Line 13 removing farther intersection points has runtime in $\mathcal{O}(N^3)$. Lines 14 to 20 combining both lists of intersection points again have runtime in $\mathcal{O}(N^2)$. Median filtering of the remaining intersection points clearly dominates the runtime of Geo-n. Lines 22 to 29 calculating the sum of distances to all other points have runtime in $\mathcal{O}(N^4)$. Finding the median in line 30 has runtime in $\mathcal{O}(N^2)$, if a method like BFPRT [15] is used. The runtime of eliminating points with a distance sum larger than the median in line 31 is again $\mathcal{O}(N^2)$. The runtime of estimating the final position by Eq. (5) is also in $\mathcal{O}(N^2)$. ■

We note that in general the number of anchors from whom a unlocalized node receives location references is low in most scenarios, limited by technical limitations of radio communication and the distance intervals, hence the running time of Geo-n is reasonable. This can also be seen in Fig. 3 where the average execution time needed for a single localization with three, six, nine, and twelve anchors is shown for Geo-n and three of the reference algorithms. The LLS algorithm has the smallest execution time because it has a closed form solution. AML and Min-Max would have even smaller execution times than LLS in this comparison, and therefore are not shown. However, the execution time of NLLS is greater than the execution time of Geo-n as long as the anchor count is below 12, although the run-time complexity of NLLS is in $\mathcal{O}(N^3)$. The execution time of ICLA, an algorithm that also uses a clustering approach, is extremely large compared to all other algorithms.

The memory requirement of Geo-n is in $\Theta(N^2)$. The memory required to store the two coordinates of the anchor nodes and range measurements is linear, namely $3N$ registers. Most memory is required to store the generated intersection points of the circles of whom $N \cdot (N-1)$ exist at most as described earlier. The $distance$ array has the same maximum size. Thus the memory requirements of Geo-n are still modest.

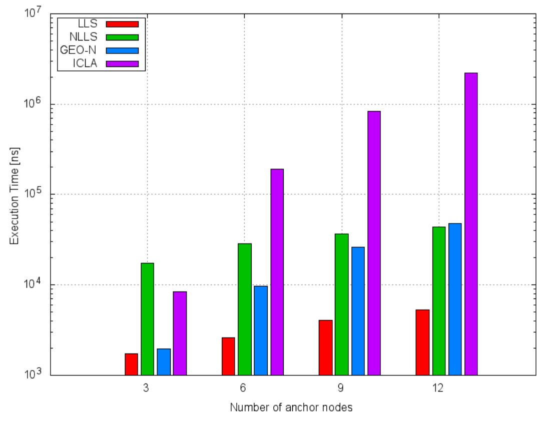

Fig. 3. Average execution times of various localization algorithms computed on a workstation.

IV. Spatial Distribution

To evaluate the spatial distribution of the position error we executed every algorithm 1000 times in the LS^2 [1], [16] simulation engine. LS^2 calculates the position error for every discrete point in the simulated area using an error model and an algorithm selected by the user. In the first scenario we chose a very basic anchor setup with four anchors placed in the four corners of the playing field. The inter-anchor distance is much higher than in most real world scenarios and shows the performance of the evaluated algorithms in borderline situations. The resulting image consists of up to three differently colored areas. The grey area indicates a position error between 100% and 250% of the expected distance measurement error value; the darker the area, the higher the error. The green area (if present) indicates a position error lower than the expected distance measurement error; the darker the area, the lower the error. In the blue area the error is higher than 500% of the distance error and is cropped to achieve a better image contrast. The anchors are represented by the small red squares.

The green area is very important for cooperative localization strategies in WSNs, because the position error stays in a reasonable range as long as the node remains in the green area. Otherwise the position error tends to grow much faster than expected because for each step of the recursive cooperation strategy the resulting position error is added to the average distance error. If the resulting position error is larger than the average distance error this error function grows very fast.

For this simulation we chose a Gaussian distributed error for the general noise simulation and an exponential distributed error to simulate NLOS situations. The expected value of the distance measurement error is 5% of the playing field width, the standard deviation is 1.5%. A NLOS error occurs with a probability of 10% and adds an exponential error with rate 2.

In Fig. 4 we present the results of the first simulation run. Four anchors have been placed in the simulation area on the edges of a quadrilateral. This setup simulates a very sparse anchor distribution which often can be found in smaller rooms in real world deployments. The Geo-n algorithm performs very well in this setup. Nearly in the whole convex hull of the anchors the position error is lower than the expected value of the distance error. Even large areas outside the convex hull show this behavior, which is important for real world deployments for the handover to the anchors of adjacent rooms. The average position error for the whole simulation area is 101% of the expected distance error, which is a very low value. NLLS (142% avg. error), LLS (144% avg. error), and AML (182% avg. error) show good performance in the center of the convex hull of the anchors and lose performance in the corners of the simulation area. Min-Max (332% avg. error) shows the typical spatial error distribution of bounding-box algorithms [17]: inside the convex hull the performance is very good but drops really fast to unusable results outside. For most indoor scenarios Min-Max performs very well because anchors are mounted at walls and so the hull cannot be left by the nodes. The performance of ICLA (132% avg. error) is approximately comparable with Geo-n, which is not very surprising because it uses the same approach of clustering circle intersections but a different clustering scheme. For this anchor setup all algorithms should perform well in real world deployments.

In Fig. 5 we present the results of the second simulation run. Four anchors have been placed in the simulation area nearly on a line. This setup simulates a very sparse anchor distribution which often can be found in hallways in real world deployments. This setup is very challenging for most

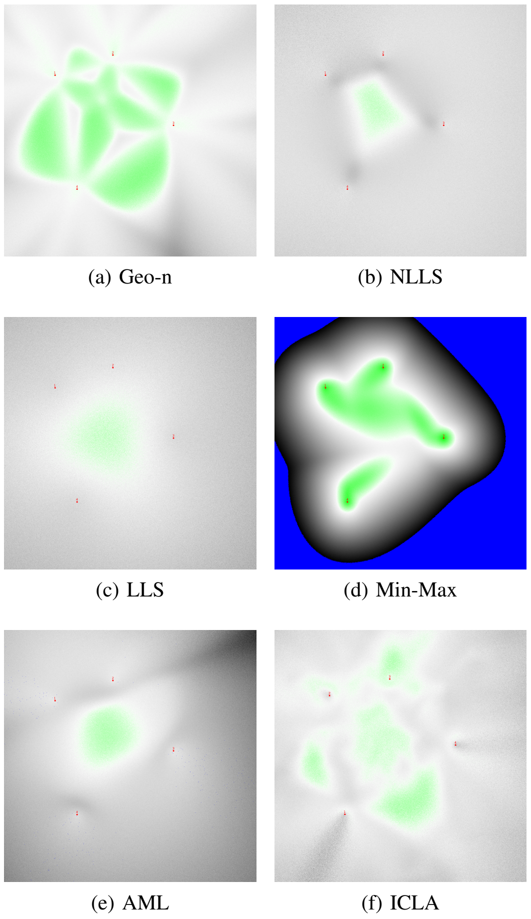

Fig. 4. Spatial distribution of the average position error with four anchors positioned as a small quadrilateral.

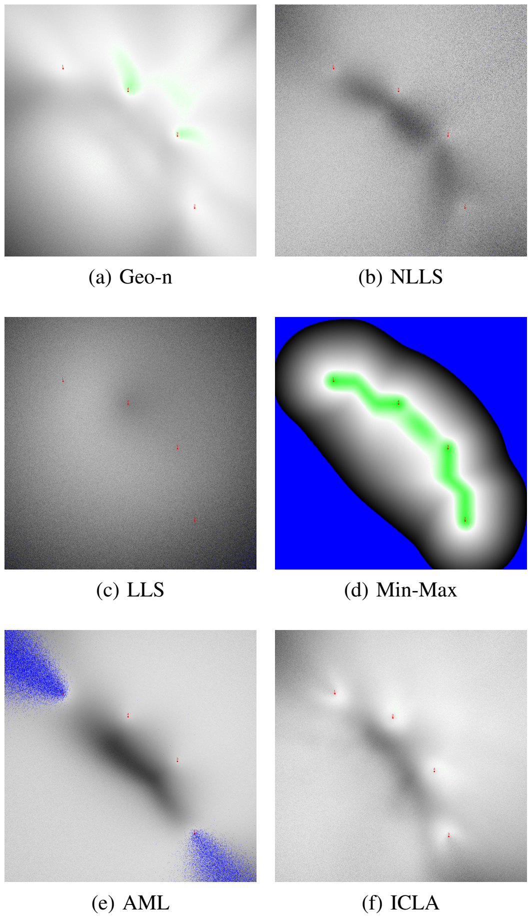

Fig. 5. Spatial distribution of the average position error with four anchors positioned nearly on a line.

algorithms because the anchors are nearly collinear and small distance errors could lead to very large position errors. The Geo-n algorithm shows a quite homogeneous error distribution compared to the other algorithms. There are only two smaller spots where it outperforms the distance error and some smaller areas where the position error grows larger than 200%. The average position error for the whole simulation area is 160% of the expected distance error. NLLS (226% avg. error) shows a very problematic spatial distribution: the weaknesses of NLLS are inside the convex hull of the anchors and the strengths lie outside. For most real world indoor deployments NLLS will perform worse than the average error. LLS (290% avg. error) shows a nearly homogeneous result but on a low level. AML (198% avg. error) has a good average performance but suffers the same weaknesses than NLLS for real world deployments. Min-Max (358% avg. error) again performs as expected: inside the convex hull the performance is very good but drops really fast to unusable results outside. The results show that Min-Max could again outperform any other algorithm but Geo-n in many likely indoor deployments. ICLA (180% avg. error) shows the same spatial distribution as AML but on a slightly lower level.

The last simulation run is shown in Fig. 6. In this setup we simulated a very dense anchor network with nine anchors. Such a setup can be found in large buildings with drywalls or in department stores, museums, etc.. In this scenario the Geo-n algorithm shows its main advantages: if enough anchors with no NLOS errors are present, it steadily filters out outliers and computes very precise results. The spatial distribution of the position error is very homogeneous on a very low level. The average position error is only 66% of the expected distance error. NLLS (124% avg. error), Min-Max (284% avg. error), and LLS (136% avg. error) perform comparably to the discussion of simulation run 1. AML (338% avg. error) drops in performance if many anchors can be reached. We have observed and discussed this behavior in recent publications [18]. ICLA again performs very good with an average error of 88% but stays behind Geo-n. Especially the spatial distribution of the position error is inhomogeneous compared to Geo-n and so the algorithm will probably only compete with Geo-n in very limited indoor scenarios with a dense anchor setup mounted on all four walls of a building.

The presented results show that the Geo-n algorithm performs very well in all three simulated situations and should also perform well in real world indoor deployments. Geo-n outperforms all other algorithms we have evaluated and shows

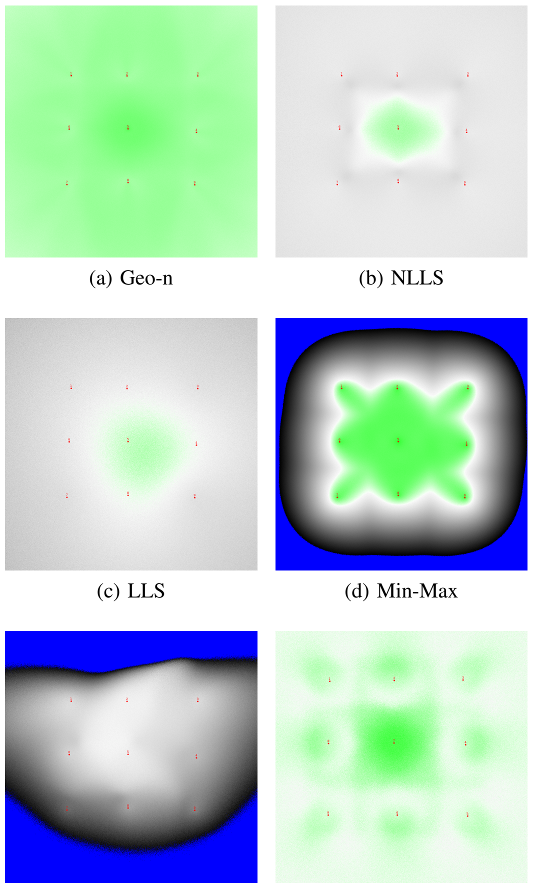
(a) Geo-n (b) NLLS (c) LLS (d) Min-Max (e) AML (f) ICLA

Fig. 6. Spatial distribution of the average position error with a dense anchor distribution of nine anchors.

very homogeneous results which is very important for cooperative localization algorithms [10], [19], [20]. For non-general cases, bounding box algorithms like Min-Max might provide better or equal performance. For all simulated scenarios only ICLA can nearly reach the low error results of Geo-n but for the price of a much higher computation time as shown in Section III.

V. Real World Evaluation

In order to measure the effectiveness of the six algorithms with real sensor network data and to be able to compare the results with the executed simulations, we recorded the data of a series of different test runs. The experiments were carried out using a modified version of the Modular Sensor Board (MSB) A2 [21] node which is equipped with a Nanotron nanoPAN 5375 [22] transceiver. This hardware enables the sensor nodes to measure inter-node ranges using time-of-flight (TOF) in the 2.4 GHz frequency band. The experiments took place on the first and second floor of our Computer Science Department during daytime. We conducted several test runs with different paths on every floor to get representative samples of indoor distance measurement conditions using varying anchor counts and inter-anchor distances.

Fig. 7 shows one exemplary campaign of measurements following a route among offices, laboratories and with a few people walking around. For the reason of clarity, we plotted only the results of Min-Max, NLLS and Geo-n. The starting point is denoted by “S”, the endpoint is denoted by “E” and the total length of the path was about 100 meters. In this run, we used 17 anchors which were deployed throughout the building. Most of the anchors were placed in office rooms with doors closed. Only a small fraction of nodes was placed on the hallway, in case of Fig. 7, there were four nodes. Ground truth was measured with the aid of a robot system developed at our Department using a Microsoft Kinect. This reference system provides about 10 cm positioning accuracy. The robot also carried the unlocalized node and followed a predefined path with a predefined speed. We used the maximum movement speed of the robot, which is 0.5 m/s. In total, we performed over 21,700 localizations when adding up all test runs. The nanoPAN achieves ranging precision of around 2.56 m on average and the RMSE is 4.15 m. However, the distance error can be as large as 30 m. We even encountered measurement errors up to 75 m in rare cases. Fig. 8 shows the distribution of the distance measurement error using all anchor nodes and all runs. Less than 3% of all measurement errors are below -1.57 m and also less than 3% are larger than 10.58 m.

The quantitative results of the four localization algorithms are shown in Table I. The average anchor degree, which is the average number of anchors seen at each location, was 9.14 throughout all experiments. As it can be seen, Geo-n outperforms the other algorithms in terms of localization accuracy with achieving an average error of 1.55 m. This is about twice as good as ICLA, a more recent algorithm following a similar approach of clustering intersection points and the third-best algorithm with an average error of 2.80 m. Only Min-Max shows a similarly good performance with an average error of 2.01 m but the localization accuracy of Geo-n is still 22.9% better. This relatively good performance of Min-Max is not surprising because the inter-anchor distances were relatively short (between 5 and 10 meters) and the mobile node took mainly positions within the bounds of the network. As we know from Section IV this is the optimal situation for Min-Max algorithm. This fact is also stated by Savvides et al. [10] and proved by Langendoen et al. [11]. Looking at a challenging scenario where the mobile node is moving outside the perimeter of the anchor nodes (Fig. 9) shows the strengths of Geo-n when compared to Min-Max. The results of Geo-n are plotted in blue and the results of Min-Max in orange. The average anchor count in this run was 8.0 though only five anchors are displayed in Fig. 9 and the distance error was 2.45 m on average. Geo-n achieves localization accuracy of 1.70 m while Min-Max only achieves 3.35 m. In this scenario Geo-n clearly outperforms Min-Max by a factor of 2.

The fact that the RMSE of Geo-n and also Min-Max is much smaller than the RMSE of the distance measurements tells us that these algorithms performed very well relative to the quality of the distance measurements available. ICLA, NLLS, and AML, with an RMSE equal or only slightly larger

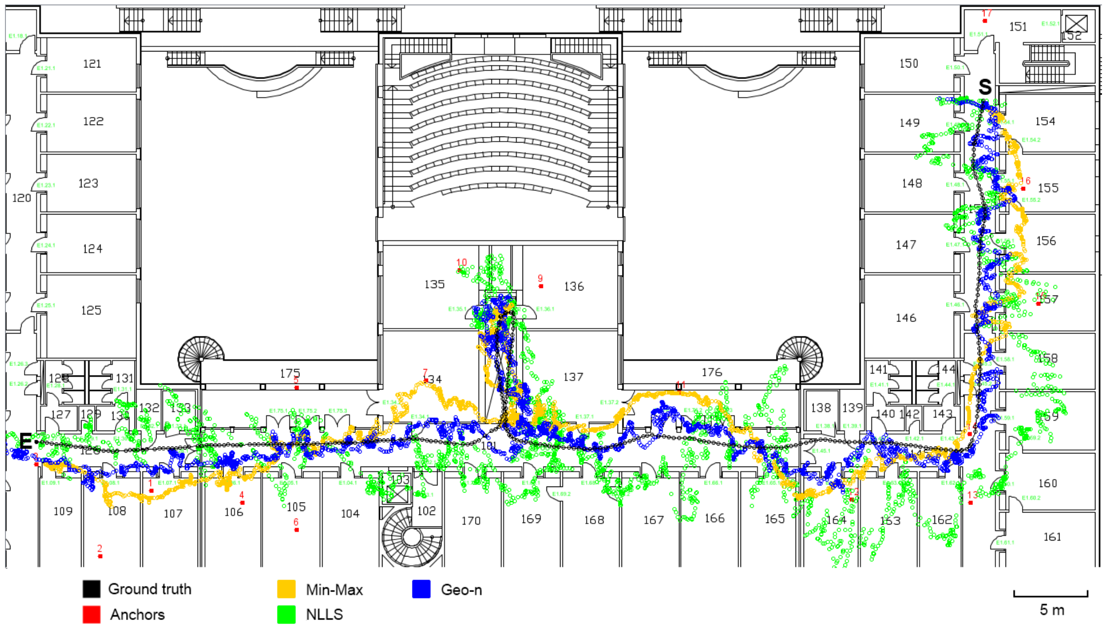

Fig. 7. Position estimates on the second floor of our Computer Science Department. For reasons of better illustration and clarity the results of all algorithms are Kalman filtered.

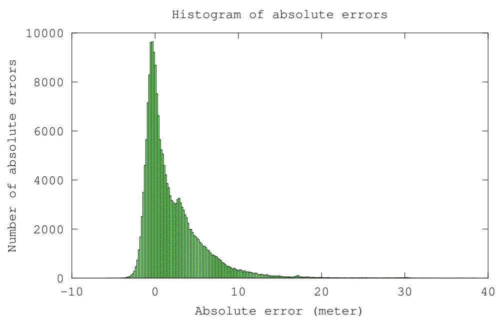

Fig. 8. Histogram of distance measurement error (all runs and anchors). Negative values are a result of too short measurements, positive values of too long measurements.

TABLE I
QUANTITATIVE RESULTS FOR THE LOCALIZATION TASK

Algorithm	MAE [m]	RMSE [m]	MAX [m]
LLS	6.54	11.09	461.63
NLLS	3.68	4.40	32.55
AML	4.30	5.23	36.76
Min-Max	2.01	2.43	20.50
ICLA	2.80	4.03	45.52
Geo-n	1.55	1.91	30.51

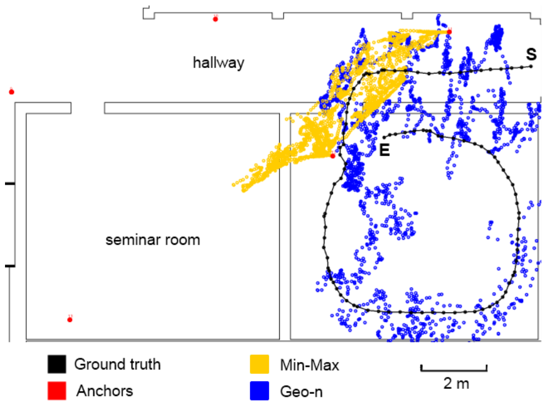

Fig. 9. Comparison between Geo-n and Min-Max on the first floor of our Computer Science Department when being outside the perimeter of the anchors.

than the RMSE of the distance measurements, also showed acceptable performance. The distribution of the localization error of all algorithms is shown in Fig. 10. It can be seen that Geo-n has the smallest spread among all algorithms and also the lowest median of the error. Furthermore, the interquartile range of Geo-n is very small compared to all other algorithms and even the upper quartile lies below the median error of all other algorithms. Obviously, the position accuracy could be improved using some filtering techniques, such as Kalman or particle filters, but the aim of this paper is to show and compare the performance of the used localization algorithms without using any of those filtering techniques.

Table II shows an analysis of the individual steps of Geo-n when compared to the centroid method of all intersection points [23] which is the starting situation after line 12 when not approximating intersections. Thus, the centroid method serves as a reference for the improvements achieved by the different steps contained in Geo-n. Approximating intersection points raises localization accuracy from 4.10 m to 3.52 m which is an improvement of over 14%. Additionally applying F2 gives an average accuracy of 1.78 m which is already a very good value when compared to the other algorithms.

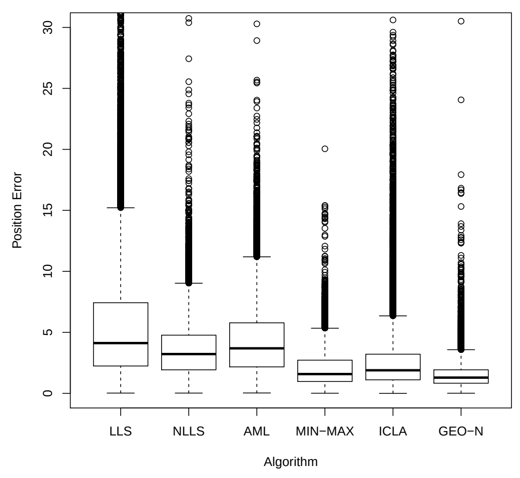

Fig. 10. The distribution of the position error for the selected algorithms. Outliers are cropped at 30 m for better readability.

TABLE II
ANALYZING GEO-N

Algorithm	MAE [m]	Improvement [m]
Centroid	4.10	-
Geo-n (approx.)	3.52	0.58 (14.1%)
Geo-n (approx. + F1)	2.15	1.95 (47.6%)
Geo-n (approx. + F2)	1.78	2.32 (56.6%)
Geo-n (F1 + F2)	2.69	1.41 (34.4%)
Geo-n (full)	1.55	2.55 (62.2%)

Also the simple filter F1 leads to good results with 2.15 m when applied additionally. Combining both filters, first F1 and then F2, finally gives localization accuracy of 1.55 m which is an improvement of 62.2% when compared to the centroid method. Using F1 removes intersection points not beneficial for the localization which helps F2 to better identify the set of intersection points close to the unlocalized node u because the median value gets lowered. Furthermore, it is remarkable that the filtering procedure seems to profit disproportionately from the approximation of intersections, in many cases more than expected. This becomes evident when looking at the improvement of 0.58 m from Centroid to Geo-n (approx.) compared to the improvement of Geo-n (F1 + F2) to Geo-n (full) which is 1.14 m. As mentioned earlier, this disproportional profit is present in about 50% of the runs conducted. Without approximation Geo-n would achieve localization accuracy of 2.69 m which is slightly better than the results of ICLA but in the same dimension.

Summarizing the presented results, it can be stated that the Geo-n algorithm showed the overall best performance of all algorithms tested under real world conditions as indicated by the simulations presented in Section IV. Geo-n's performance is very homogeneous in nearly all situations which is also supported by the fact of having only very little outliers as seen in Fig. 10.

VI. CONCLUSION

We presented Geo-n as a precise, distance-based localization algorithm for use in indoor scenarios. Using LS^2 we have shown that Geo-n has a very homogeneous spatial distribution of position errors, especially when the anchor count is large. In both simulations and real experiments, Geo-n outperforms all other tested algorithms. The algorithm ICLA is similar to Geo-n and is outperformed by 44.6% and even Min-Max, an algorithm working very well in common indoor deployments, is still outperformed by 22.9%. The commonly used NLLS algorithm is outperformed by 57.9%. Compared to ICLA, the runtime of Geo-n is reasonable and low enough for use on low power WSN nodes although having asymptotic run-time in $\mathcal{O}(N^4)$. It even outperforms NLLS which has run-time in $\mathcal{O}(N^3)$ for typical 3 to 12 anchors found in the transmission range in indoor deployments.

We propose Geo-n as a new general purpose localization algorithm, especially for indoor deployments. The source code of the algorithm is freely available together with the LS^2 package [16]. Future work should address the integration of a weighting component for the generated intersection points of the last step based on the distance error distribution into the algorithm to gain even higher performance improvements.

REFERENCES

[1] H. Will, T. Hillebrandt, and M. Kyas, "The fu berlin parallel lateration-algorithm simulation and visualization engine," in *Proc. 9 Workshop on Positioning, Navigation and Communication 2012 (WPNC12)*, 2012.

[2] K. Whitehouse, C. Karlof, A. Woo, F. Jiang, and D. Culler, "The effects of ranging noise on multihop localization: an empirical study," in *Proc. 4th international symposium on Information processing in sensor networks*. IEEE Press, 2005, p. 10.

[3] K. Whitehouse, C. Karlof, and D. Culler, "A practical evaluation of radio signal strength for ranging-based localization," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 11, no. 1, pp. 41–52, 2007.

[4] I. Guvenc, C. Chong, and F. Watanabe, "Analysis of a linear least-squares localization technique in los and nlos environments," in *Vehicular Technology Conference, 2007. VTC2007-Spring. IEEE 65th*. IEEE, 2007, pp. 1886–1890.

[5] S. Venkatraman, J. Caffery Jr, and H. You, "A novel toa location algorithm using los range estimation for nlos environments," *IEEE Trans. Vehicular Technology*, vol. 53, no. 5, pp. 1515–1524, 2004.

[6] S. Venkatesh and R. Buehrer, "A linear programming approach to nlos error mitigation in sensor networks," in *Proc. 5th international conference on Information processing in sensor networks*. ACM, 2006, pp. 301–308.

[7] L. Lazos and R. Poovendran, "Serloc: Robust localization for wireless sensor networks," *ACM Transactions on Sensor Networks (TOSN)*, vol. 1, no. 1, pp. 73–100, 2005.

[8] A. Srinivasan and J. Wu, "A survey on secure localization in wireless sensor networks," *Encyclopedia of wireless and mobile communications*, 2007.

[9] D. Moore, J. Leonard, D. Rus, and S. Teller, "Robust distributed network localization with noisy range measurements," in *Proc. 2nd international conference on Embedded networked sensor systems*. ACM, 2004, pp. 50–61.

[10] A. Savvides, H. Park, and M. B. Srivastava, "The bits and flops of the n-hop multilateration primitive for node localization problems," in *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*, ser. WSNA '02. New York, NY, USA: ACM, 2002, pp. 112–121. [Online]. Available: http://doi.acm.org/10.1145/570738.570755

[11] K. Langendoen and N. Reijers, "Distributed localization in wireless sensor networks: a quantitative comparison," *Comput. Netw.*, vol. 43, no. 4, pp. 499–518, Nov. 2003. [Online]. Available: http://dx.doi.org/10.1016/S1389-1286(03)00356-6

[12] G. S. Kuruoglu, M. Erol, and S. Oktug, "Localization in wireless sensor networks with range measurement errors," *Advanced International Conference on Telecommunications*, vol. 0, pp. 261–266, 2009.

[13] L. Haiyong, L. Hui, Z. Fang, and P. Jinghua, "An iterative clustering-based localization algorithm for wireless sensor networks," *China Communications*, vol. 8, no. 1, pp. 58–64, 2011.

[14] J. E. Dennis, Jr. and R. B. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations (Classics in Applied Mathematics, 16)*. Soc for Industrial & Applied Math, 1996.

[15] M. Blum, R. W. Floyd, V. Pratt, R. L. Rivest, and R. E. Tarjan, "Time bounds for selection," *J. Comput. Syst. Sci.*, vol. 7, no. 4, pp. 448–461, Aug. 1973. [Online]. Available: http://dx.doi.org/10.1016/S0022-0000(73)80033-9

[16] "LS^2 - lateration simulator." [Online]. Available: http://inf.fu-berlin.de/groups/ag-tech/projects/ls2

[17] H. Will, T. Hillebrandt, Y. Yuan, Z. Yubin, and M. Kyas, "The membership degree min-max localization algorithm," in *2nd International Conference and Exhibition on Ubiquitous Positioning, Indoor Navigation, and Location Based Service (UPINLBS12)*. IEEE, 2012.

[18] T. Hillebrandt, H. Will, and M. Kyas, "Quantitative and spatial evaluation of distance-based localization algorithms," *Lecture Notes in Geoinformation and Cartography, Proc. of LBS 2012*, 2012.

[19] J. Albowicz, A. Chen, and L. Zhang, "Recursive position estimation in sensor networks," in *Network Protocols Ninth International Conference on ICNP 2001*. IEEE, 2001, pp. 35–41.

[20] A. Savvides, C. Han, and M. Strivastava, "Dynamic fine-grained localization in ad-hoc networks of sensors," in *Proc. 7th annual international conference on Mobile computing and networking*. ACM, 2001, pp. 166–179.

[21] M. Baar, H. Will, B. Blywis, T. Hillebrandt, A. Liers, G. Wittenburg, and J. Schiller, "The scatterweb msb-a2 platform for wireless sensor networks," no. TR-B-08-15, 09 2008. [Online]. Available: ftp://ftp.inf.fu-berlin.de/pub/reports/tr-b-08-15.pdf

[22] "nanopan 5375 rf module datasheet, berlin, germany, 2009. [online], available: http://www.nanotron.com."

[23] N. Bulusu, J. Heidemann, and D. Estrin, "GPS-less low-cost outdoor localization for very small devices," *IEEE Personal Communications*, vol. 7, no. 5, pp. 28–34, Oct. 2000. [Online]. Available: http://dx.doi.org/10.1109/98.878533