RF-Based Location System Using Cooperative Calibration

Ricardo Reghelin and Antonio Augusto Fröhlich
Laboratory for Software/Hardware Integration (LISHA)
Federal University of Santa Catarina (UFSC)
88049-900 Florianopolis - SC - Brazil
Email: {ricardo / guto}@lisha.ufsc.br
http://www.lisha.ufsc.br/~{ricardo / guto}

ABSTRACT
This paper studies the problem of determining the location of nodes in a wireless sensor network. We describe a fully decentralized algorithm called HECOPS where every node estimates its own position after interactions with other nodes. Only a limited fraction of nodes have knowledge of their position coordinates, however any node can be selected as a reference. We propose a ranking system to determine the reliability of each estimated position. This leads to a novel approach for position calculation that uses fewer but more reliable landmarks, thus reducing data communication and limiting error propagation. Heuristics are used to reduce the effects of measurement errors, including a scheme to calibrate range measurements by comparing, whenever possible, the estimated distance with the actual distance between a pair of nodes. Experiments demonstrate that the algorithm is superior to a previously proposed method in terms of its ability to compute correct coordinates under a wider variety of conditions and its robustness to measurement errors.

Categories and Subject Descriptors
C.2.4 [**Computer-Communication Networks**]: Distributed Systems – *Distributed applications.*

General Terms
Algorithms.

Keywords
Localization Algotythm, WSN.

1. INTRODUCTION
Wireless sensor networks have many attractive applications in data collection and detection and tracking of objects

This paper proposes a solution for the following problem: given a set of nodes with unknown position coordinates, and a mechanism by which a node can estimate its distance to a few nearby (neighbor) nodes, determine the position coordinates of every connected node via local node-to-node communication.

The proposed positioning algorithm is based on two elements: node positions and range measurements (RM), that is, the measurements of the distances between pairs of neighboring nodes. This algorithm is indifferent to which method is used. Some methods generate RMs with errors as large as 50%. RM errors can come from multiple sources, including multipath interference, line-of-sight obstruction, and channel inhomogeneity with regard to direction.

This study treats two major challenges in positioning within an ad-hoc space [5]. The first challenge is to reduce RM errors. Our algorithm improves accuracy using several heuristics, the most important being an algorithm to calibrate RMs based on comparisons between the actual distance and the calculated distance obtained by RMs. The second challenge is related to the sparse anchor node problem. The system responds to this problem by considering not only anchors, but all nodes, as potential references for position calculations. We call these references "landmarks".

In Section 2 we discuss related work in this field. Section 3 explains the algorithm and heuristic used to reduce errors. Section 4 reports the experiments carried out in order to characterize the performance of the algorithm. At last, section 5 closes the article.

2. RELATED WORK

2.1 Previous Auto-localization Systems
There is a substantial body of research addressing localization but we concentrate on algorithms with the following classification:

- range-based algorithms which consider RM's.
- anchor-based algorithms which assume that a certain minimum fraction of the nodes know their positions.
- decentralized algorithms in which all position calculations run on each individual node.

We presented some algorithm with these characteristics.

Savarese *et al.* proposed two algorithms in [6]. The idea behind the ABC algorithm is that each node should get RM's from all available neighboring nodes, even if they are not anchors. In the beginning of the algorithm, if no global reference is encountered, the node assumes a hypothetical position that later is corrected if the node connects to the network over multiple hops.

Another algorithm, Terrain, improves on ABC by first flooding the network with information from anchor nodes.

Instead of simply flooding the network with the anchors' positions, some approaches, such as Niculescu *et al.* [3], proposing DV-hop, and Savarese *et al.* [5] proposing Hop-

TERRAIN, improve the initial positions by considering the number of hops between anchors. Once an anchor gets a position from another anchor after a certain number of hops, it estimates an average size for the hops by diving the Euclidean distance between the anchors by the number of hops. This hop size is then deployed as a correction to the entire network. Another similar approach is DV-distance [3], which considers instead of the total numbers of hops, the sum of all RM's between the anchors.

Savvides *et al.* [7] propose an iterative multi-lateration scheme (AHLoS), where a node solves a set of over-constrained equations relating the distances among a set of anchors and a set of non-anchor nodes (including itself). Savarese *et al.* [5] propose a rank of confidence to assign to each node. Nodes with low confidence are not be considered in multi-lateration, which reduces error. The algorithm improves accuracy but does not result in a relationship between confidence and position error.

In the MDS algorithm [4] a single defined starting node first initializes flooding to communicate its position to three or more anchor nodes, which are called ending anchors. The ending anchors send their locations and the flooding routes to each of them. The starting sensor first simply estimates its physical position with a tri-lateration based on its hop distances to the ending anchors, which is similar to the distance vector exchange-based method [3]. Then, it estimates the positions of those sensors that are on these routes or one hop away from it. Probably MDS is the most similar research to our work. The main similarity is that MDS proposes a heuristic calibration obtained by comparing an estimated position with the physical position.

None of the above related work meets all of our design objectives discussed in Section I. All of them are potentially inaccurate and also several possess intolerable computational complexity .

In order to obtain more accurate position estimation in anisotropic networks and to avoid error propagation with the problem of cumulative errors, we propose a heuristic-based distributed method based.

3. HECOPS LOCALIZATION ALGORITHM

3.1 Overview

The main idea of this algorithm is to use fewer but more reliable landmarks for position calculation. A node is a landmark for another node if any RM is established and also if it can transmit its position coordinates. In order to best select landmarks for calculation, we create a ranking system based on the confidence of node position and the confidence of landmarks. The confidence of landmarks is complementary to the confidence of node position and depends on the network configuration and topology.

3.2 Range measurements (RM) improvements

Distance estimation is crucial to reduce errors when calculating position. HECOPS employs a heuristic approach to improve accuracy on distance estimation between nodes. The most important aspect is an algorithm to offset errors caused by RF propagation. We suggest three other heuristics.

3.2.1 Deviation

The ideal radio range of a node is a circle centered on the node. In real world, however, a node usually has an irregular radio pattern as shown in Figure 1. This means that the radio range of a node is different in different directions [4]. Many conditions affect RF propagation, which causes differences in measurements. The signal strength received by one node can be different from the expected value.

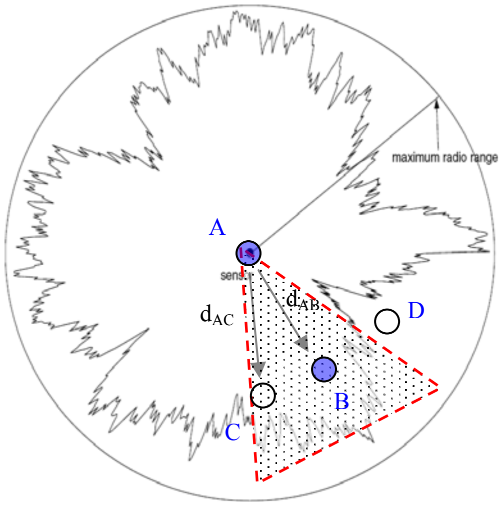

Figure 1 Irregular radio pattern of a sensor [4]

The packet reception rate is directly related to RSSI. An empirical study [1] indicates that the distribution of the packet reception rate over distance is quite non-uniform, as Figure 2 shows. In fact, individual contours clearly exhibit directionality, with propagation being better at some directions than others. But the most important conclusion, which is fundamental for the understanding of this algorithm, is that although irregular, RSSI levels can be seen as concentric circles. This means that there exist a difference between the estimated RM's and the physical distance. This difference, which we call "deviation", is an error caused by environmental conditions.

Considering the RF propagation in one specific direction, the idea here is first to identify eventual deviations in this specific direction. Then we consider this deviation valid for all nodes within the area along this direction. This last consideration makes it possible to determine a calibration factor for the related RM.

In other words, the system calibrates RM's by comparing, in some cases, the estimated distance with the actual distance between nodes. The estimated distance is dependent on the communication device used, such as RSSI. The actual distance is calculated as the Euclidean distance when the coordinates of a pair of nodes is known. In Figure 1, we can see an example where nodes A and B are anchors and can identify eventual deviation. Node C is along the distance AB (d_{AB}) so it is presumed to be affected by the same deviation on AB

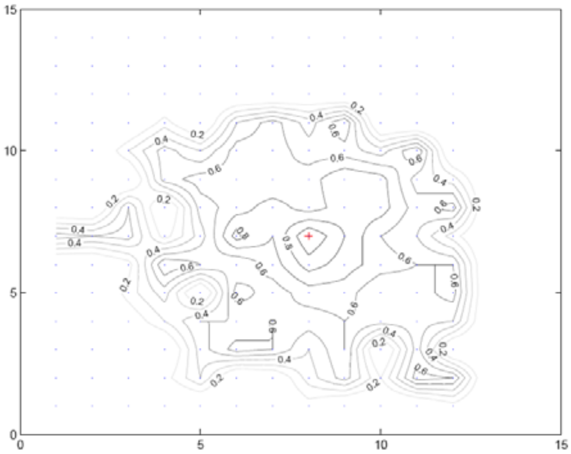

Figure 2 Contour of probability of packet reception from a central node [1]

The formula for deviation (dev) is:

$$dev_{AB} = L_{real} / d_{AB} \qquad (1)$$

Where: dev_{AB}= deviation from node A to node B

Lreal = real distance between nodes A and B

d_{AB}= estimated distance AB, calculated by RM

The area in which the deviation should be applied is suggested in Figure 3 and is called the "tri function area". It is an isosceles triangle as tall as 1.5 times the distance between the pair of reference points (i.e., one node, A, that sends a signal and one node, B, that receives it).

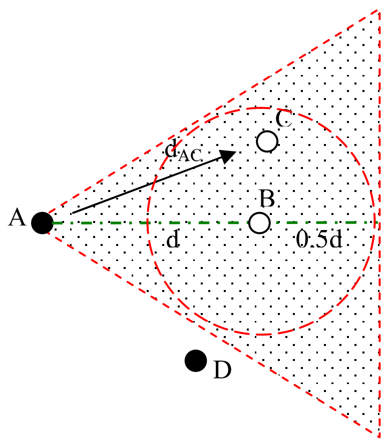

Figure 3 Area interested on the A-B distance deviation defined by function: triangle (A,B)

We define here a function called tri (A,B). A node is said to have a tri function if its position is inside any tri function area where nodes A and B have individual confidence values bigger than 0.80. In the example on Figure 3, node B receives a beacon from node A and eventually identifies a deviation, dev_{AB}. Node C has tri (A,B) and will receive this deviation information and consider it when it calculates position. When d_{AC} is calibrated by B, we can represent it as d_{ACB}. Node D is out of range and considered too far to be affected by dev_{AB}.

Due to the high computational effort necessary to calculate a triangular area, the area affected could be changed to a circle centered on receiving node. An example can be seen in Figure 3, where node B is the center.

3.2.2 Permanent deviation

This would be an extended application of the tri function. Suppose one node detects many deviations around itself for 360 degrees. We can suppose that the reason is an error related to the transmitter/receiver hardware device. The insight here is to establish a permanent deviation for this node to compensate for this error in all directions or in a specific angle of transmission. Note that this kind of information received by neighboring nodes should be sent to the receiving node, so there must be an additional protocol. This is not demonstrated in this paper.

3.2.3 Beacons from all nodes

Considering two nodes, A and B, we probably have different RM's for A to B and B to A. The algorithm is design to consider each RM independently. Depending on topology and environmental conditions, the algorithm will select automatically the most reliable RM's to be considered.

3.2.4 Transmissions at multiple power levels

As demonstrated in [2] and [1], transmitting at various power levels, the RF signal propagates at various levels in its medium and it is possible to collect different results at the receiver. According to [1], varying transmission power therefore diversifies the set of measurements obtained by receiving nodes and in fact increases the accuracy of tracking by several meters in our experiments. HECOPS can accept nodes broadcasting beacon messages at various transmission power levels.

3.3 Coordinates System

3.3.1 Position Calculation

Lateration is most common method but requires a high computational effort [7]. Min–max method [8] is simple but not accurate enough. As we use no more than four landmarks in our work, we choose to use the following function:

$$\text{Min } f(x,y) = \sum_{i=1 \text{ to } 4} [(x_i - x_n)^2 + (y_i - y_n)^2]^{0.5} - d_i \qquad (2)$$

Where *n* are the four selected landmarks, *x* and *y* are the coordinates, and *d* are the RM's from nodes selected as landmarks.

Initial numbers and limits can be obtained by the min-max method or results from previous calculations. The closer the initial numbers are to the optimum value, the faster the algorithm converges. The algorithm increases and decreases coordinates within limits in such way the function always minimize. Process ends when function result value or iteration count value reaches values previously defined.

3.3.2 Weighting information to select landmarks

A weight value is assign to each node as a measure of the reliability of its position and so it is possible to rank all nodes. The Hop-Terrain method [5] has a similar definition, so we maintain two aspects: we use the same term, "confidence", to represent both the weight and rank value, varying from 0 to 1. Here similarity between methods ends. An anchor node has top reliability defined as 1.0. Other nodes have confidence rated from 0 to 0.8, depending how much we can trust its estimated position.

Confidence is calculated by the following formula:

$$C_n = \sum_{i=1 \text{ to } 3} (C_i * 0.75 + tri_{in} * 0.25) / 3 * 0.8 \quad (3)$$

Where: C_n= confidence for node n

i= top 3 results of $(Ci * 0.75 + tri_{in} * 0.25)$

C_i= confidence of node i

tri_{in} = confidence value of the node that composes a tri function with nodes i and n; if the value is less than 0.7, we set it to 0.

Note that a non-anchor node will never receive C=1, no matter how much reliable information is available. This is because only anchors know their positions exactly; other nodes will always estimate.

An RM from an anchor is quite reliable. But it is even more reliable if this RM is calibrated by a deviation (dev) as explained in Section 3.2.1. Therefore, the tri factor on formula (3) makes one RM much more reliable and likely to be chosen for position calculation.

Only reliable nodes can help in a tri function, otherwise uncertainty would increase. So only tri values bigger than 0.7 are considered when using the formula (3).

The constant numbers suggested (0.75 and 0.25) ensure that a final calculated C depends mostly on the C value of the neighbor nodes, as we expect. The tri function alone helps to increase C, however it cannot by itself push the C value very high, otherwise a tri function with a low initial C value would be chosen instead of an anchor.

3.4 Main algorithm

Each node stores a certain number of its most reliable landmarks in a table with basic information for position calculation. Each line is related to one landmark. This table is transmitted line by line to other nodes. Receiving nodes try to store all lines and find a tri function.

3.4.1 Table structure

The first line of table is reserved to for the current node parameters. These are: ID of the receiving node (ID_n), x-coordinate of node position (X_n), y-coordinate (Y_n), permanent deviation estimate (D_n), confidence value of ID_n position (C_n), number max of lines in current table (L_n), and value of the minimum confidence value in the current table (C_{min}). All other lines in the table refer to selected landmarks. Each line contains information related to a single landmark: ID of beacon node (ID_1), ID of receiving node (ID_2), x-coordinate of beacon node position (X_1), y-coordinate (Y_1), signal strength RM value (S_{12}), deviation estimate (*D12*), confidence value of *ID1* position (C_1), ID of the node forming a tri function (ID_{tri}), and confidence value of ID_1 (C_{tri}).

3.4.2 Updating tables

We assume that there must be a sequence of beacons for all nodes. Each receiving node (B) updates its table at every beacon node (A) transmission according to the algorithm summarized by following pseudo code:

```
read packets and define beacon ID and RM(d_AB) value
create line in table, storing beacon ID and RM
if table is full
then replace line with less confidence
else include line in the table
repeat read line n until last line
    check if there is a tri formed by the 3 nodes: beacon,
    receiving and line n
    if tri = true
    and Confidence n >C_min (min C accepted)
    then include or replace(if full) line in table
    else read next line
select lines from table with top confidence
if only one landmark
then assume position is X Y+ d_AB from landmark
else calculate position X Y
calculate confidence C_B of nodes position
```

Note that receiving node not only reads RM's from beacon node, but also reads all table lines transmitted, in order to check if this beacon node can be an ID_{tri}.

After the limit for the number of lines in the table is surpassed, new lines enter the table by replacing the line with lowest confidence value.

4. SIMULATION RESULTS

We simulated the performance of HECOPS varying topology, node connectivity, and ranging. We evaluated its performance against the Hop-Terrain algorithm. We wrote a Matlab-based simulator to experiment, analyze, and visualize the performance and behavior of the different localization algorithms.

All data points represent averages over 20 trials in networks containing 100 nodes. The nodes are randomly placed, with a uniform distribution, within a square area. The population of anchors is defined and randomly positioned. The RM between connected nodes is blurred in the following way: for each beacon node, all receiving nodes have a certain error value added to RM, with the true range as the minimum. This error is randomly defined from a normal distribution and limited to a maximum error value specified. In order to simulate the irregular RF propagation, all receiving nodes on right side of the beacon will have one random error value different from the receiving nodes from the left side. The connectivity (average number of neighbors) is controlled by specifying the radio range.

To allow comparison between different scenarios, range errors as well as errors on position estimates are normalized to the radio range (i.e. 50% position error means half the range of the radio).

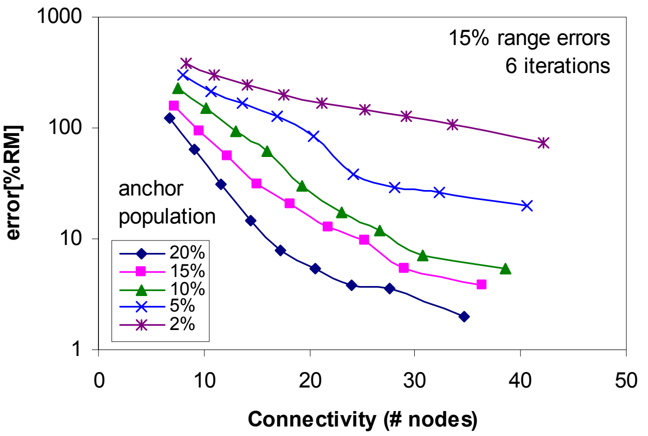

Figure 4 Average position error after HECOPS

Figure 4 shows the average performance of the HECOPS algorithm as a function of connectivity and anchor population in the presence of 15% range errors. As seen in this plot, position estimates have an average accuracy of under 100% error in scenarios with at least 5% anchor population and an average connectivity level of 10 or greater. In extreme situations where very few anchors exist (2%), errors reach above 100% only after level 35 of connectivity.

Figure 5 is similar to Figure 4 but adds the results of the Hop-Terrain algorithm. For certain connectivity between level 10 to 28, HECOPS always gets around 10% less error than Hop-Terrain. HECOPS needs 15 level connectivity to reach 30% error, Hop-terrain needs 22, almost 50% more. Therefore HECOPS can be considered more robust in terms of a low density population network.

In every iteration, all connected nodes get more information from references, and so they can make a better selection. HECOPS get better accuracy after 5 iterations, as can be see in Figure 6. Hop-Terrain called this iteration process, “refinement”. Hop-Terrain initially performs better, but stabilizes at a larger error value.

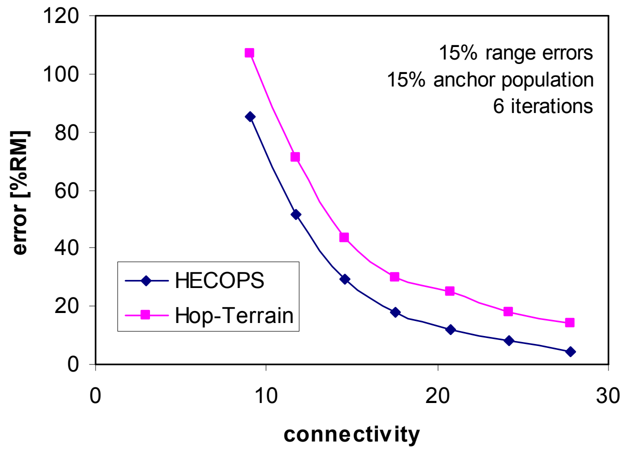

Figure 5 Comparing average position errors: HECOPS and Hop-Terrain

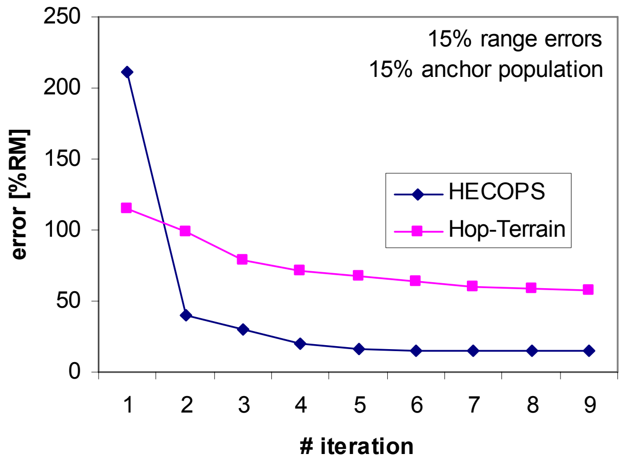

Figure 6 Comparing results along iterations (HECOPS and Hop-Terrain)

Although HECOPS uses only four references, each node stores many RM's. The objective is to increase the probability of detecting tri functions. One drawback of having large tables is the increased communication of unnecessary data. Another problem is limited storage hardware. In Figure 7, we state the maximum limitation for the table of references. For HECOPS we discovered there is no significant penalty. For Hop-Terrain, however, errors only drop under 10% with the limit set to 22 nodes. That means a reference table could have a size of 10 lines and maintain the same accuracy as a table with more than 10 lines.

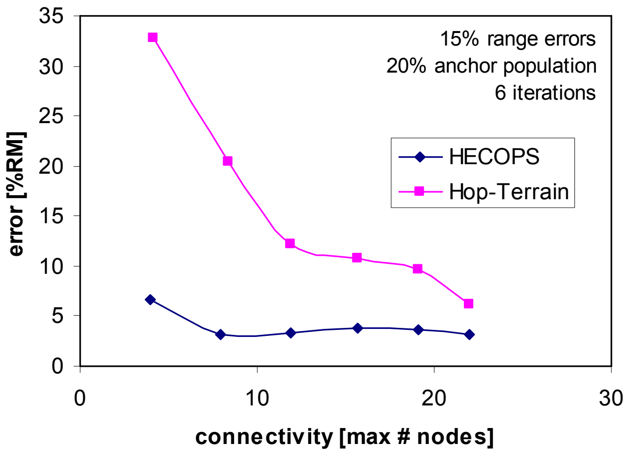

Figure 7 Average position error restricting number of references

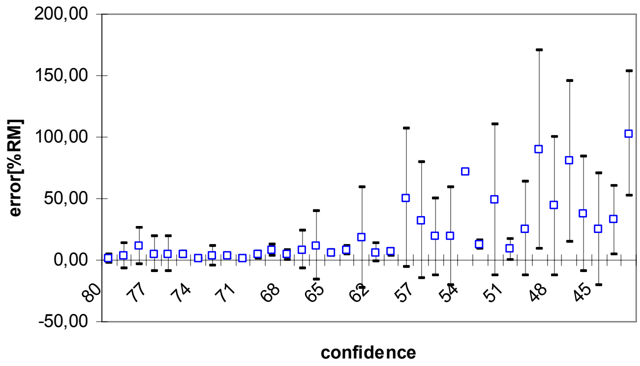

Figure 8 Relation between confidence and positioning error (average and standard deviation)

Figure 8 Relationship between confidence and positioning error (average and standard deviation). The relationship between confidence and positioning error (average and standard deviation) is interesting. According to [5], Hop-Terrain confidence has high standard deviation and so does not allow further analysis. HECOPS, by contrast, has a significant relationship as can be seen in Figure 8. Nodes with a confidence value higher than 65 have average position error below 10% and also low standard deviation. Confidence lower than 65 gives errors around 50% and high standard deviation. These results prove the concept of the algorithm - that only nodes that have high confidence in their own position can serve as a reference for other nodes to calculate their position.

5. CONCLUSIONS AND FUTURE WORK

In this paper we have presented a completely distributed algorithm for solving the problem of positioning nodes within an ad-hoc, wireless network of sensor nodes.

The algorithm features two main improvements on past research: it uses fewer reference nodes to calculate position and uses heuristics to calibrate range measurements. We have described each scheme in detail.

We have explained the simulation environment used to evaluate the algorithm, including details about the specific implementation. We applied the simulation to another algorithm (Hop-Terrain) and compared the two algorithms on the same basis data. We have documented many experiments for each algorithm, showing several aspects of the performance achieved under different scenarios.

The results show that we are able to achieve position errors of less than 10% in a scenario with 15% range measurement error, 10% anchor population, and an average connectivity of 30 nodes.

In the near future we plan to implement experimental tests using nodes in the real world. We also plan to carry out analysis of the communication costs of our methods.

6. REFERENCES

[1] K. Lorincz and M. Welsh, A Robust, Decentralized Approach to RF-Based Location Tracking, tech. report TR-19-04, Division of Eng. and Applied Sciences, Harvard Univ., 2004.

[2] D. Ganesan, B. Krishnamachari, A. Woo, D. Culler, D.Estrin and S. Wicker, Complex Behavior at Scale: An Experimental Study of Low-Power Wireless Sensor Networks, Technical Report UCLA/CSD-TR 02-0013, 2002.

[3] D. Niculescu, D., AND Nath, B, Ad-hoc positioning system, In Proceeding of IEEE Global Communications Conference (GLOBECOM), pp.2926-2931, November 2001.

[4] X. Ji and H. Zha. Sensor positioning in wireless ad-hoc sensor networks with multidimensional scaling. In Proceedings of the 23rd Conf. of the IEEE Communications Society (INFOCOM), 2004.

[5] C. Savarese, C., Rabaey, J., And Langendoen, K. Robust Positioning Algorithms for Distributed Ad-Hoc Wireless Sensor Networks. In USENIX Annual Technical Conference,

[6] C. Savarese, J. Rabaey, and J. Beutel, Locationing in distributed ad-hoc wireless sensor networks, In IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP), pages 2037-2040, Salt Lake City, UT, May 2001.

[7] A. Savvides, A., Han, C., and Srivastava, M. Dynamic Fine-Grained Localization in Ad-Hoc Networks of Sensors. In Proc. 7th ACM MOBICOM Conf. (July 2001), pp. 166–179

[8] A. Savvides, H. Park, M. Srivastava, The bits and flops of the N-hop multilateration primitive for node localization problems, in: First ACM International Workshop on WSNA, Atlanta, GA, 2002, pp. 112–121.