A Decentralized Location System for Sensor Networks Using Cooperative Calibration and Heuristics

Ricardo Reghelin and Antônio Augusto Fröhlich
Laboratory for Software and Hardware Integration
Federal University of Santa Catarina
PO Box 476 – 88049-900 – Florianópolis, Brazil
reghelin@inf.ufsc.br, guto@lisha.ufsc.br

ABSTRACT

In this paper we study the problem of determining the location of nodes in a wireless sensor network, describing a fully decentralized algorithm called HECOPS, where every node estimates its own position after interacting with other nodes. Only a limited number of nodes have exact knowledge of their position coordinates. Any node can, however, be selected as a reference. We establish 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. We present heuristics that 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 Algorithms, Wireless Sensor Networks

1. INTRODUCTION

Wireless Sensor Networks (WSN) have many attractive applications in data collection, detection and tracking of objects [8], and environmental observation and forecasting [1], such as habitat monitoring [3], health care [1], battlefield surveillance and enemy tracking [10] . Many of these applications, however, depend on an appropriate positioning system. Furthermore, several Wireless Sensor Networks communication protocols now under development take advantage of the geographic positions of nodes [4, 15].

By placing a large number of relatively cheap sensor devices, also called sensors or nodes, in positions close to the phenomenon of interest, it is possible to obtain accurate position estimates. Each sensor device usually is capable of sensing a specific phenomenon, processing data on a small scale, and communicating via an omni-directional radio signal [1]. WSNs typically do not use high-power transmission devices due to data collision, energy consumption, and high cost. This results in simple devices with a limited range of communication to nodes in the neighborhood. Another limitation is that it would be costly to equip all nodes with self-localization devices such as GPS. This problem is approached in general by equipping only a fraction of the population of nodes with self-localization devices, or even manually placing nodes that have a priori knowledge of their position. These nodes are called beacon nodes or anchor nodes. All other nodes determine their own position by interacting with anchors and non-anchor nodes. This determination is the localization problem in a WSN.

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. A node is said to be connected when it is able receive and transmit data to other nodes in the network[1].

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. Several techniques can be used to generate these RMs, including time of arrival (ToA), angle of arrival (AoA), phase measurements, and received signal strength (RSSI). This algorithm is indifferent to which method is used, except that different methods offer different tradeoffs between accuracy, complexity, cost, and power requirements. Some of these methods generate RMs with errors as large as 50%. Range measurements errors can come from multiple sources, including multipath interference, line-of-sight obstruction, and channel inhomogeneity with regard to direction.

[1]The tracking problem is not explained in detail in this article, but is understood to be an extension of the localization problem.

MSWiM'06, October 2–6, 2006, Torremolinos, Malaga, Spain.

This study treats two major challenges in positioning in an ad-hoc space [13]. 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. We present an accurate, robust, and decentralized approach to RF-based localization, called HECOPS (Heuristic Environmental Consideration Over Positioning System). HECOPS can be used with any network of battery-operated wireless nodes to measure, store, and compute location information. It uses a decentralized approach to computing locations that runs on the sensor device rather than on a back-end server.

The objective of this study is to propose a local positioning system for a WSN with the following features:

- self-configurability: the local positioning system should work based on the coordination of the nodes inside the wireless network, without any assistance from other infrastructure;
- robustness: the positioning technique should tolerate RM inaccuracy, e.g., in the estimation of distance between a pair of nodes; also it should tolerate the failure of some nodes;
- high accuracy: the system should provide location information that is accurate enough to support target applications (e.g., tracking mobile objects).

The remainder of this article presents the proposed solution in detail. Section 2 presents 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 evaluate the performance of the algorithm. Finally, section 5 closes the article with the authors' conclusions about the studied problem and the proposed solution.

2. RELATED WORK

There is a substantial body of research addressing localization but we concentrate on algorithms with the following classification:

- range-based algorithms which consider RMs. These are different from range-free algorithms which consider only the proximity of a pair of nodes
- anchor-based algorithms which assume that a certain minimum number or fraction of the nodes know their positions, e.g., by manual configuration or using some other location mechanism
- decentralized algorithms in which all position calculations run on each individual node.

Savarese et al. proposed two algorithms in [14]. The idea behind the ABC algorithm is that each node should get RMs 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. The author suggests a rule apply to nodes with three or fewer RMs. With four or more RMs, the node can calculate its position by multi-lateration using the least min square method. As more nodes come into the range, position estimation improves. The authors report that a 5% RM error results in about 60% average position error. This is a consequence of the propagation of errors.

The Terrain algorithm improves on ABC by first flooding the network with information from anchor nodes. The algorithm provides an initial solution for each node in the network by multi-hop forwarding of the anchor positions. The authors report that a 5% RM error results in about 35% average position error.

Once an initial estimation is obtained, location accuracy can be improved through an iterative refinement process, that is, a new cycle of calculation using the first estimated numbers. Instead of simply flooding the network with the anchors' positions, some approaches, such as Niculescu et al. [12], proposing DV-hop, and Savarese et al. [13] 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 dividing the Euclidean distance between the anchors by the number of hops. This hop size is then deployed as a correction to the entire network. The advantages of these approaches are their simplicity and the fact that they do not depend on RM. The drawbacks are that they will only work for isotropic networks, that is, when RM errors are the same in all directions, so that the corrections that are deployed reasonably estimate the distances between hops.

Another similar approach is DV-distance [12], which considers the sum of all RMs between the anchors instead of the total numbers of hops. This approach is more accurate because it considers different hop sizes, but on the other hand it is sensitive to RM errors.

Niculescu et al. [11] propose a distributed algorithm that uses angle-of-arrival (APS-AoA) for localization. The nodes collect the angle information from neighbors and derive coordinates by using angle-based triangulation.

Also based on angle of arrival (AoA) technique, Nasipuri et al. [9] propose a localization scheme based on a set of reference points deployed with known coordinates. The reference points transmit high power signals to cover the entire network area. The nodes receive the signals from at least three reference points and determine their coordinates by triangulation according to the angle bearings of the incoming signals. A drawback to using angle-of-arrival is that it is expensive compared to RSSI and also it is difficult to obtain precise angle estimates.

Savvides et al. [15] 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). When a non-anchor node estimates its location, it becomes an anchor. This process repeats until it calculates the positions of all the nodes that eventually can locate three or more anchors. A drawback of iterative multi-lateration is the error accumulation that results from the use as beacons of unknown nodes which estimate their own positions. Fortunately, this error accumulation is not very high because of the high precision of the ranging method.

Savarese et al. [13] 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.

Doherty et al. [5], propose a centralized convex position estimation system. This is an algorithm using connectivity constraints among anchors. The connectivity of the network is represented by a set of convex position constraints and a centralized linear-programming algorithm is used to obtain the coordinates of unknown nodes. The main drawback of this approach is the high computational complexity involved in solving the linear program problem. In the MDS algorithm [7] 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 [12]. 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.

A recursive position estimation approach that requires fewer reference nodes is proposed by Albowicz in [2] for sensor networks. A node that is close to at least three reference points estimates its position through nonlinear regression. After the node obtains a reasonable position estimate, it may serve as a new reference point. This process can be applied recursively until all nodes in the network have obtained their coordinates. While the recursive approach saves on hardware cost, it sacrifices accuracy, especially for nodes far away from the original reference points. This algorithm also ranks all nodes in terms of confidence, based on a residual error value from a position calculation. This paper also has similarity to our work in the sense that it establishes a ranking system to select references. None of the above related work meets all of our design objectives discussed in Section 1. All of them are potentially inaccurate and also several possess intolerable computational complexity [2, 5].

Most of the methods fail to perform well in the anisotropic topology of the sensor networks and complex terrain where the sensor networks are deployed [7]. Also, some algorithms enable error propagation [12, 14]. 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.

3. HECOPS

In this section, we describe HECOPS, our novel localization algorithm. 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.

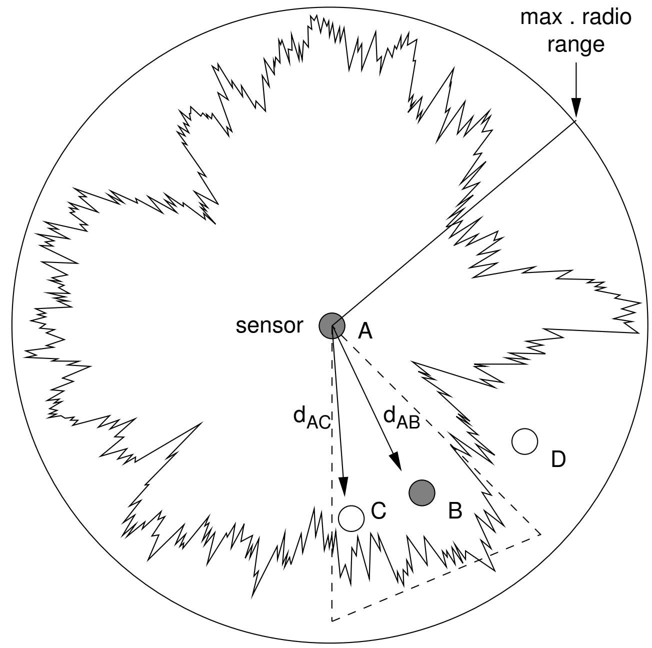

Figure 1: Irregular radio pattern of a sensor

3.1 Range measurements 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.

3.1.1 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 [7]. This means that the radio range of a node is different in different directions [7]. Many conditions affect RF propagation, which causes differences in measurements. The signal strength received by one node can be different from the expected value.

The packet reception rate is directly related to RSSI. An empirical study [6] indicates that the distribution of the packet reception rate over distance is quite non-uniform, as Figure 2 shows [6]. 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 RMs 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 RMs by comparing, in some cases, the estimated distance with the actual dis-

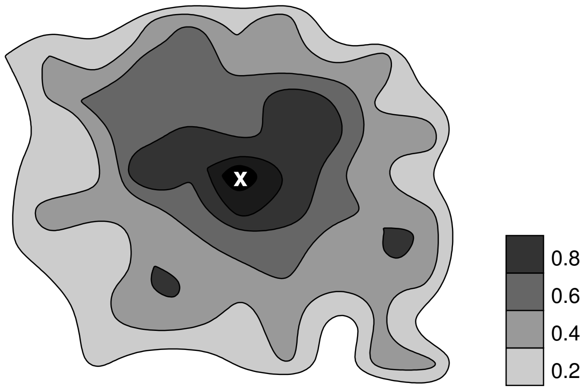

Figure 2: Contour of probability of packet reception from a central node

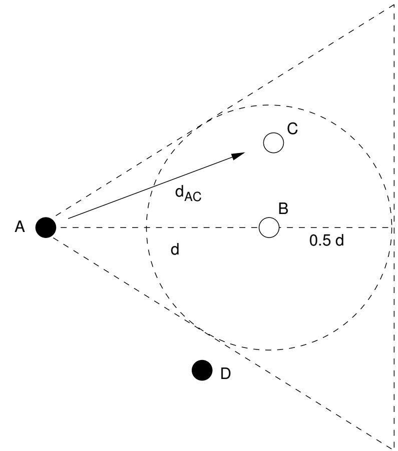

Figure 3: Area interested on the A-B distance deviation defined by function: tri (A,B)

tance 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.

The equation for deviation (dev) is:

$$dev_{AB} = \frac{L_{real}}{d_{AB}} \tag{1}$$

where:

dev_{AB} is the deviation from node A to node B;

L_{real} is the real distance between nodes A and B;

d_{AB} is the estimated distance between A and B, using RM ($RM_{A,B}$).

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).

We define here a function called tri (A,B). The name comes from the triangle shape. 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. It is interesting to note that for node C, d_{AC} , although farther, is a much reliable information than d_{BC}. 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. The circle ratio is easily determined dividing distance AB by distance BC.

3.1.2 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.

3.1.3 Beacons from all nodes

RF propagation is different in an anisotropic environment. Considering two nodes, A and B, we probably have different RMs 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 RMs to be considered. The problem with increasing the number of RMs is an equivalent increase in computational complexity. In our experiment we do not consider this feature due to complexity of the algorithm. However is clear that this feature can improve accuracy by multiplying the potential tri functions, which improve accuracy.

3.1.4 Transmission at multiple power levels

As demonstrated in [6] and [8], 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 [8], 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. Increasing the number of transmission power levels involves a trade-off in terms of higher storage space for landmarks. No experiments were carried out with this feature due the need for actual physical conditions. In any case, the implementation of the algorithm would not change in this regard.

3.2 Coordinates System

All nodes can exchange information and establish a coordinate system by themselves without central support. The basic idea is to select a number of nodes serving as landmarks. This approach is similar to [18] and to other past work, but here all nodes in the network can serve as landmarks and each node selects its own landmarks. In this section, we first introduce a Euclidean distance estimation model. Then we present an algorithm to calculate position and select landmarks. For simplicity, we illustrate the proposed approach in a two-dimensional space, but naturally this could be applied to three dimensions as well.

3.2.1 Euclidean Distance Estimation

A pair of nodes can estimate their distance if they are within transmission range of each other. Various techniques, such as RSS, ToA, or TDoA (Time Difference of Arrival, a variation of ToA), can be used [15]. However, when nodes are not adjacent or few in number, the distance estimation becomes nontrivial and requires nodes to exchange and process data. We propose a solution in the next section.

3.2.2 Position Calculation

Lateration is most common method but requires a high computational effort [15, 17]. Min-max method [16] is simple but not accurate enough. As we use no more than four landmarks in our work, we choose to use the following equation:

$$Min\ f(x,y) = \sum_{i=1}^{4} \sqrt{(x_i - x)^2 + (y_i - y)^2]} - d_i \qquad (2)$$

where i are the selected landmarks, x and y are the coordinates, and d are the RMs from nodes selected as landmarks.

Initial numbers 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 until this results in a function result value or iteration count value previously defined.

The minimum number of landmarks is three, but due to errors already described in Section 1, most systems use as many landmarks as possible. HECOPS instead uses four landmarks due to uncertainty, in case three nodes are in a straight line. In this last case, the algorithm would lead to two possible results.

3.2.3 Weighting information to select landmarks

Most of positioning systems consider all nodes available to calculate position, not only anchors, but also non-anchors that have estimated their positions. In HECOPS, however, each node selects just a small number of the nodes which are most reliable in terms of position accuracy. 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 [13] 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 equation:

$$C_n = \frac{\sum_{i=1}^{3} C_i \times 0.75 + tri_{i,n} \times 0.25}{3} \times 0.8 \qquad (3)$$

where:

C_n is the confidence for node n

i are the landmarks, top 3 values obtained from (C_i * 0.75 + $tri_{i,n}$ * 0.25)

C_i is the confidence of node i

$tri_{i,n}$ is the confidence value of the node that composes a tri function with nodes i and n; if the value is less than 0.8, 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 equation (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. Thus only tri values bigger than 0.8 are considered when using equation 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.

3.3 Main algorithm

In this section, we explain how nodes store and share information. Each node stores a certain number of its most reliable landmarks in a table. Each line is related to one landmark. This table is transmitted line by line to other nodes. Besides the RM, the receiving nodes store all lines and try to identify a tri function. If a tri function is identified, the information is added to the respective line in the table.

3.3.1 Table structure

Each node of the network maintains a data table with basic information for position calculation.

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. Examples of the structure can be seen in Table 1. 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 ($S_{1,2}$), deviation estimate ($D_{1,2}$), confidence value of ID1 position (C_1), ID of the node forming a tri function (ID_t), and confidence value of ID1 (C_t).

ID_1	ID_2	X_1	Y_1	$S_{1,2}$	$D_{1,2}$	C_1	ID_t	C_t
		200	300			0.73		
b	a	30	400	197	1.02	1	c	1
c	a	150	220	94		1		
d	a	50	50	292		0.51		
e	a	80	20	305	1.01	0.75	c	1

Table 1: Example of table of reference structure

The quantity of lines stored in each node is determined by capacity of the node's hardware, transmission data, and maximum error desired. The maximum number proposed is 10 and we will demonstrate this later. Each node starts with the table empty and starts storing lines as soon as it receives data from the network.

3.3.2 Updating tables

We assume that there must be a sequence of beacons for all nodes. In the example, the signal transmission sequence will be ABCD. Beacon nodes transmit their respective tables, line by line. Each receiving node (B) updates its table at every beacon node (A) transmission according to the algorithm summarized the pseudo code presented in Figure 4.

```
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;
end
for line = first ... line <= last do
    read line;
    check if there is a tri formed by the 3 nodes:
    beacon, receiving and line n;
    if tri == true and C_n > C_min then
        include or replace (if full) line in table;
    else
        read next line;
    end
end
select lines from table with top confidence;
if only one landmark then
    assume position is (x, y + d_A,B);
else
    calculate position (x, y);
end
calculate confidence C_B of nodes' position;
```

Figure 4: Pseudo-code for updating tables

If the receiving line indicates a new RM, then a new line is included in the table. A receiving node not only reads RMs 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. This ensures that the table maintains only lines with the most reliable landmarks. In case only one reference is available, the node just assumes position (x , $y + d_{A,B}$) from beacon. The first estimation is poor, but may be improved in the next iterations.

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 5 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 6 is similar to Figure 5 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 2 iterations, as can be see in Figure 7. Hop-Terrain called this iteration process, "refinement". Hop-Terrain initially performs better, but stabilizes at a larger error value.

Although HECOPS uses only four references, each node stores many RMs. 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 8, 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 9 presents the 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 [16], Hop-Terrain confidence has high standard deviation and so does not allow further analysis. HECOPS, by contrast, has

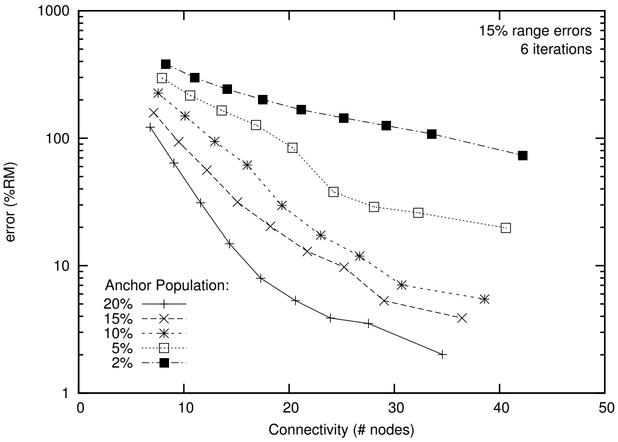

Figure 5: Average position error for different anchor populations after HECOPS

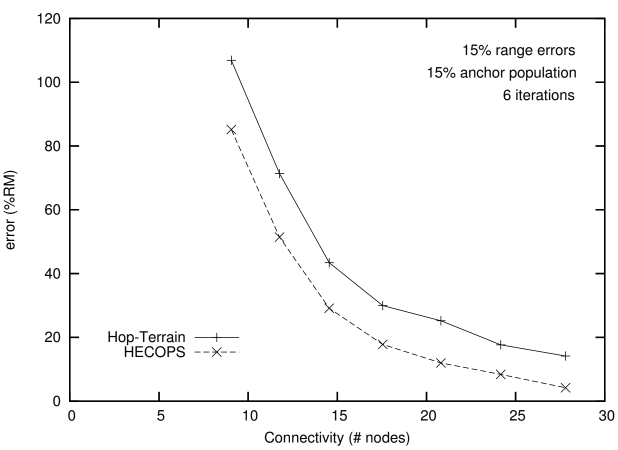

Figure 6: Average position errors: HECOPS and Hop-Terrain

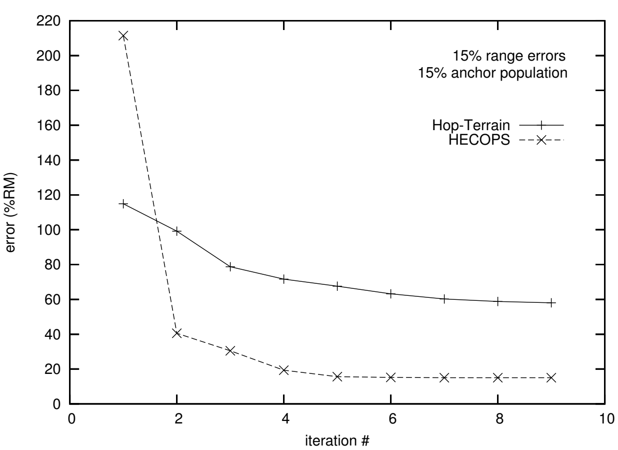

Figure 7: Results along iterations: HECOPS and Hop-Terrain

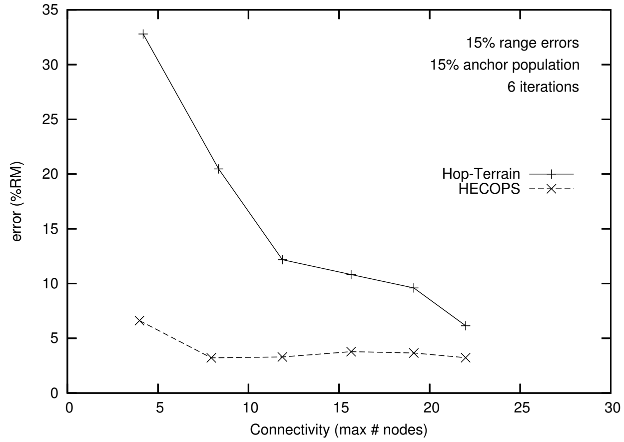

Figure 8: Average position error restricting number of references

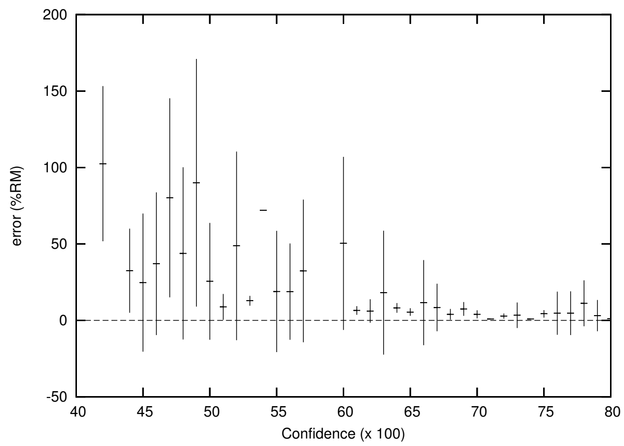

Figure 9: Relation between confidence and positioning error (average and standard deviation)

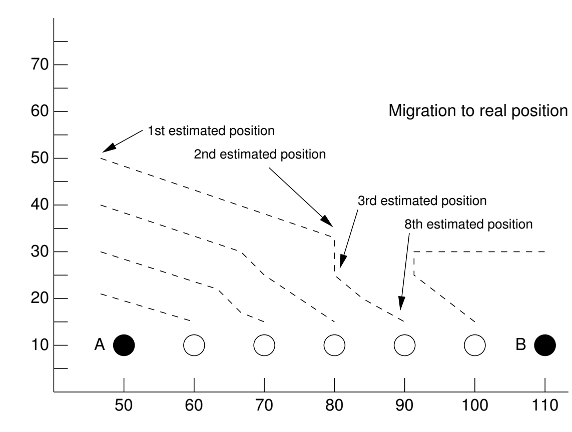

Figure 10: Evaluating performance on a toy-problem

a significant relationship as can be seen in Figure 9. Nodes with a confidence value higher than 0.65 have average position error below 10% and also low standard deviation. Confidence lower than 0.65 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.

Finally, we created a toy problem to evaluate the algorithm for extreme multi-hop performance, as can be seen in Figure 10. Between two anchor nodes (A and B) there are 5 nodes in a straight line with RMs limited to 10. Each node can see just two neighbors. The node begins with a rough estimation close in value to the anchor node's position. In every iteration, the node's calculated position becomes closer to its real position. The dotted lines indicate various estimates of each node. This result proves that HECOPS works in a multi-hop fashion and with a minimum number of references.

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] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. A survey on sensor networks. *IEEE Communications Magazine*, 40(8):102–114, Aug. 2002.

[2] J. Albowicz, A. Chen, and L. Zhang. Recursive position estimation in sensor networks. In *Proc. IEEE Int. Conf. Network Protocols (ICNP'01)*, pages 35–41. IEEE Computer Society, 2001.

[3] E. S. Biagioni and K. W. Bridges. The application of remote sensor technology to assist the recovery of rare and endangered species. *The International Journal of High Performance Computing Applications*, 16(3):315–324, Fall 2002.

[4] 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, October 2000.

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

[6] 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 CSD-TR 02-0013, UCLA, 2002.

[7] X. Ji and H. Zhaa. Sensor positioning in wireless ad-hoc sensor networks with multidimensional scaling. In *Proc. IEEE Conf. on Computer Communications (INFOCOM)*, 2004.

[8] K. Lorincz and M. Welsh. Motetrack: A robust, decentralized approach to RF-based location tracking. In T. Strang and C. Linnhoff-Popien, editors, *Location- and Context-Awareness: First International Workshop, Oberpfaffenhofen, Germany. Proceedings*, volume 3479 of *Lecture Notes in Computer Science*, pages 63–82. Springer, 2005.

[9] 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'02)*, pages 105–111, New York, Sept. 28 2002. ACM Press.

[10] J. L. Nemeroff, L. Garcia, D. Hampel, and S. Di Pierro. Networked sensor communications for the objective force. In R. Suresh and W. E. Roper, editors, *Proc. SPIE Vol. 4741*, pages 29–35, Aug. 2002.

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

[12] D. Niculescu and B. Nath. Ad hoc positioning system (APS). In *Proc. IEEE Global Telecommunications Conf. (GLOBECOM)*, pages 2926–2931, 2001.

[13] C. Savarese, K. Langendoen, and J. Rabaey. Robust positioning algorithms for distributed ad-hoc wireless sensor networks. In *USENIX Technical Annual Conference*, pages 317–328, Monterey, CA, June 2001.

[14] C. Savarese, J. Rabaey, and J. Beutel. Locationing in distributed ad hoc wireless sensor networks. In *Proc. 2001 Int. Conf. Acoustics, Speech, and Signal Processing (ICASSP 2001)*, volume 4, pages 2037–2040. IEEE, May 2001.

[15] A. Savvides, C.-C. Han, and M. B. Srivastava. Dynamic fine-grained localization in ad-hoc networks of sensors. In *Proc. Int. Conf. on Mobile Computing and Networking (MOBICOM)*, pages 166–179, 2001.

[16] 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'02)*, pages 112–121, New York, Sept. 28 2002. ACM Press.

[17] K. Whitehouse, C. Karlof, A. Woo, F. Jiang, and D. E. Culler. The effects of ranging noise on multihop localization: an empirical study. In *Proc. Int. Conf. on Information Processing in Sensor Networks (IPSN)*, pages 73–80. IEEE, 2005.

[18] H. Wu, C. Wang, and N.-F. Tzeng. Novel self-configurable positioning technique for multihop wireless networks. *IEEE/ACM Trans. Netw.*, 13(3):609–621, 2005.