Evaluation of an RSSI-based Location Algorithm for Wireless Sensor Networks

Rafael Pereira Pires, Lucas Francisco Wanner and Antônio Augusto M. Fröhlich

Laboratory for Software and Hardware Integration - LISHA
Federal University of Santa Catarina - UFSC
P.O.BOX 476, Florianópolis, SC, Brazil

{rafaelpp,lucas,guto}@lisha.ufsc.br

Received 30 december 2006

Abstract Position awareness is a desirable feature for many applications of Wireless Sensor Networks. The Received Signal Strength Indication of a radio channel provides a feasible way of estimating distance between nodes because its use doesn't require any additional hardware but a radio transceiver. The main drawback of using RSSI is its instability and interference susceptibleness noticed in real environments. This work shows an evaluation of an implementation of a location algorithm for wireless sensor networks and presents improvements that substantially improve the trustworthiness in its results. The results show that adequately adjusted RSSI measurements can be successfully used for localization in Wireless Sensor Networks.

Keywords Location Algorithms, Wireless Sensor Networks, RSSI, Ad Hoc Networks, Embedded Systems

§1 Introduction

Many applications for wireless sensor networks take into account the geographic position of the nodes, hence, they need a system that provides them some kind of location information. Examples of such applications include tracking of objects, habitat monitoring, environmental observation and forecasting,

battlefield surveillance and enemy tracking. To deal with that, GPS (Global Positioning System) devices attached to sensor nodes could be used. The GPS system gives coordinates with good precision, but it doesn't work in some indoors environments and it's based on a considerable satellites infrastructure, thus not being suitable for low cost devices in ad hoc networks. For this kind of network, the location systems should be self-contained (to not depend on a fixed infrastructure), and robust to failing nodes and to errors in range measurements. These characteristics must be implemented considering the nodes restricted resources like power, processing and communication.

HECOPS (Heuristic Environmental Consideration Over Positioning System) [2)] is a distributed location algorithm for Wireless Sensor Networks, 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. HECOPS establishes a ranking system to determine the reliability of each estimated position, and uses 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. This work brings an implementation and evaluation of the HECOPS algorithm over the EPOS Operating System [1)] and presents several adaptations that improve the reliability on its results.

The main problem of location algorithms that use RSSI as range measurement for distance estimation is its instability noticed in practice. By observing this characteristic and the algorithm behavior, we've noticed that little alterations in the way that the distances are corrected could considerably improve position estimations reliability, enabling its real deployment.

The next section presents a description of the HECOPS algorithm. Section 3 shows the used infrastructure and some considerations about the implementation. Section 4 presents an evaluation of the results and a discussion about the changes introduced in HECOPS. Finally, section 5 closes the paper.

§2 The HECOPS Location Algorithm

The HECOPS's location algorithm aims at solving 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 node via local node-to-node communication [2)]. Hence, some nodes need to have an a priori knowledge of their

position based on some global coordinates system. These nodes are called *anchors*. Based on its coordinates, all other nodes will estimate their position. The position information of the anchor nodes can came from GPS devices installed in just a small amount of nodes, preestablished in the source code or by methods of self establishment of coordinates [6] executed before the location service.

Another important factor in distributed algorithms of localization is how to estimate distances between pair of nodes directly connected. There are techniques based on time propagation of messages like ToA (Time of Arrival) and TDoA (Time Difference of Arrival), but they need high resolution timers. Other techniques, like AoA (Angle of Arrival) and distance estimation based on optical and ultrasound devices are additional hardware dependent. By the independence of any other devices but a radio transceiver, already required for wireless communication, the HECOPS uses as range measurement the RSSI (Received Signal Strength Indication). The estimated distance is inversely proportional to signal strength received.

The main problem of RSSI is its variation, which is completely non-uniform in all directions and is easily affected by electromagnetic interferences and physical barriers. These characteristics bring much imprecision in this kind of estimating distance and, so, the algorithm must treat them in an efficient way. Figure 1 shows the contour of probability of packet reception in relation to distance to a transmitting node x.

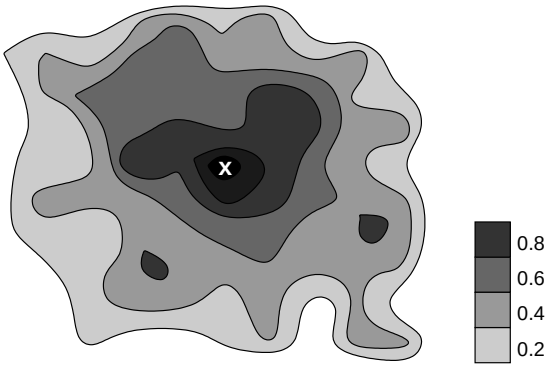

Figure 1 Contour of probability of packet reception from a central node [2]

Even with the instability of RSSI, we find that its variation is directed related to the direction of the radio transmission signal, which means that nodes

positioned in the same direction of a transmitter node will make similar readings, with almost linear variations according to distance. Based on that, in HECOPS, the distances to anchor nodes estimated by nodes with unknown position are calibrated by correction factors obtained by other nodes in the same direction. This factor, called *deviation*, is defined by the ratio between the actual distance, calculated by nodes that already know their positions, and the signal strength of a message sent by one of them. For example, in Figure 2, nodes B and X, by being positioned at the same direction related to A, are supposed to be affected by the same deviation. Considering A and B anchor nodes, the deviation would be given by $dev_{AB} = \frac{d_{AB}}{RSSI_B}$, where dev_{AB} is the deviation, d_{AB} the Euclidean distance between A and B, and $RSSI_B$ the RSSI reading of the node B of a message sent by A. Therefore, the distance between A and X would be given by $d_{AX} = RSSI_X \times dev_{AB}$. Node D wouldn't be affected by dev_{AB} because it's not in the same direction.

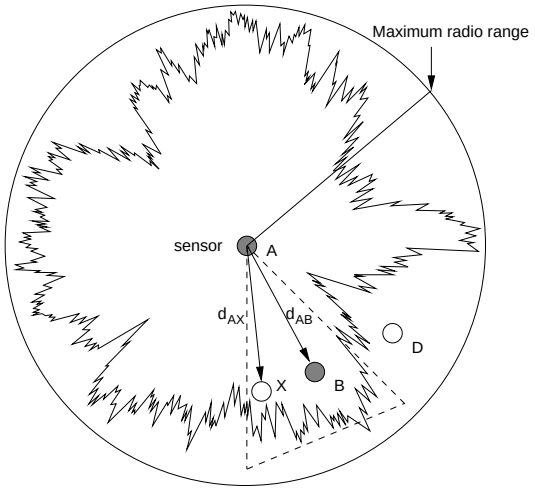

Figure 2 Irregular radio pattern of a sensor and calibration system based on direction[2)]

We expect the required anchor proportion be as small as possible, without compromising the results accuracy. For that reason, the HECOPS allows nodes with estimated positions to be chosen as landmarks. With the aim that the algorithm performance doesn't be compromised by this characteristic, a heuristic scheme that gives a value to the confidence on location information given by nodes was developed to choose the best landmarks. Each node, when calculating its position, defines a confidence value on the result obtained. This value ranks

the nodes that should be chosen by a node that has to estimate its location.

Confidence calculation is based on the confidence value of the nodes chosen as landmarks and on the confidence of the nodes used in distance calibration related to that landmarks. In a scale varying from 0 to 1.0, anchor nodes have maximum confidence on its position, equal to 1.0. The other nodes have confidence limited by 0.8, given by equation (1), where C_x is the confidence on position that is being calculated by a node X, C_i the confidence on each landmark chosen by X (n in total) and C_{ix} the confidence of the node that, together with the node i, have defined the deviation applied to the distance between the nodes i and X, if any (In Figure 2, C_{ix} would be the confidence on node B, considering i the node A).

$$C_x = 0.8 \times \frac{\sum_{i=1}^{n}(C_i \times 0.75 + C_{ix} \times 0.25)}{n} \quad (1)$$

Location information received from anchors is very trustworthy. But, if the distance estimation of that node has been calibrated by another node, the confidence is even greater. For this reason, the weights of 0.75 and 0.25 were attributed for the confidence on a chosen landmark and the confidence of the node used to calibrate the distance between them, respectively. Thus, when a node that has to estimate its position has already chosen its landmarks and have the estimated distances to all of them, it's enough to apply some method to calculate coordinates, like lateration or min-max [3].

§3 Location on EPOS

In order to evaluate the viability and the results of implanting the HECOPS in real sensors devices, we have built the algorithm in the application level of the EPOS operating system (*Embedded Parallel Operating System*) [1, 5, 4]. The communication infrastructure for sensor networks in EPOS provides applications with the RSSI reading for received messages. The chosen target prototypes of evaluation was the Mica2, with a processor Atmel ATMega128, an 8-bit word AVR architecture with 7.37 MHz clock.

By being an 8-bit architecture, the AVR8 doesn't have a floating point unit, therefore, such operations must be made in software. The problem is that the existing libraries for that purpose are too big and have a considerable computation cost for these small resource devices. As a workaround, we've decided to use integer arithmetic. But, in order to don't loose much precision, the used values in calculations should be raised some magnitude orders, in a

way that, after processed, still have enough precision to not compromise the algorithm results.

In many cases, mathematic libraries are also unavailable, but, for distance calculation, we have to use the square root. With that purpose we've used an iterative numerical analysis method used to find the roots of a function based on its derivatives, the Newton-Raphson method. By the behavior of the quadratic function $f(x) = x^2 - n, n \in \Re$ and $n > 0$, the application of this method has quick convergence (until 20 iterations in order to obtain an one unit precision). The positive root of the function is equal to the square root of n.

In the beginning, only the anchor nodes know their positions. They initialize the process by sending messages to every reachable nodes. This message contain its identification (ID), coordinates (x,y) and confidence (Figure 3(a)). The nodes that will receive this message will store the information together with the RSSI reading. If the receiving node already know its position, it calculates the distance and deviation between them and send these information to every reachable nodes (Figure 3(b)). This information is also stored by the nodes who wish estimate its position.

(a) (b)

Figure 3 Messages exchanged content: (a) Position information message. (b) Deviation information message

When a message with deviation information is received by a node that doesn't know its coordinates, it checks if it's in the same direction than the transmitter, related to the third node described in the message. If it is, it calibrates the RSSI reading of a message sent by that third node with the deviation.

The checking of a node to discover if it's in the same direction of another one related to an sending node is made according to proximity between them. In Figure 4, the node C receives a message from B about the deviation between A and B. So, node C verifies if its distance to node B is lower than the half of the distance to A. In a positive case, node C calibrates the RSSI reading of the last message received from A with the deviation between A and B.

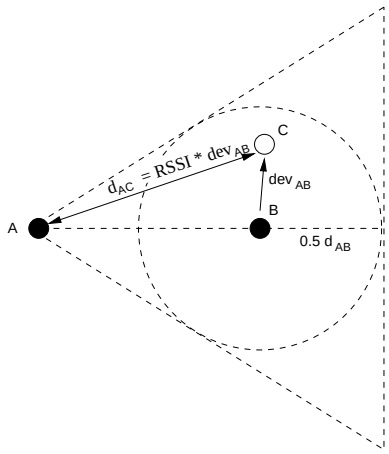

Figure 4 Definition to discover if two nodes are in the same direction, in order to use calibration

Position information messages are stored by the nodes that will estimate its coordinates in a list ordered by the confidence value. When the list size reaches 3 it's already possible to execute the position calculation. The 3 nodes of the list with greatest confidence in their positions are chosen as landmarks.

The used method for coordinates calculation was the min-max, as it has lower computational complexity than lateration (what would require the solving of systems of linear equations). In this method, an area where probably is located the node estimating its position is defined. Therefore, the area is divided in a certain number of points and, for each point, a residue value is calculated, based on (2), where (x_i, y_i) corresponds to the coordinates of each point inside the defined area, (x_j, y_j) are the coordinates of each anchor node (n in total) and d_j the estimated distance from the node that is estimating its position and the anchor j. The point that presents the least residue value is assumed as the node position.

$$Residuo = \sum_{j=1}^{n} \left(\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} - d_j \right) \tag{2}$$

Figure 5 shows how the area where the points used in residue calculation is defined. It's rectangular and is defined by the intersection of the intervals $\bigcap_{j=1}^{n} [x_j - d_j, x_j + d_j]$ and $\bigcap_{j=1}^{n} [y_j - d_j, y_j + d_j]$ in the x-axis and y-axis, respec-

tively. Hence, the area is divided in a customizable number of points, in order to adequate the calculation to the processing power supported by each device. Finally, when node position is defined, it sends a message like the Figure 3(a) to all its neighboring nodes, in a way that it can serve as landmark for another nodes.

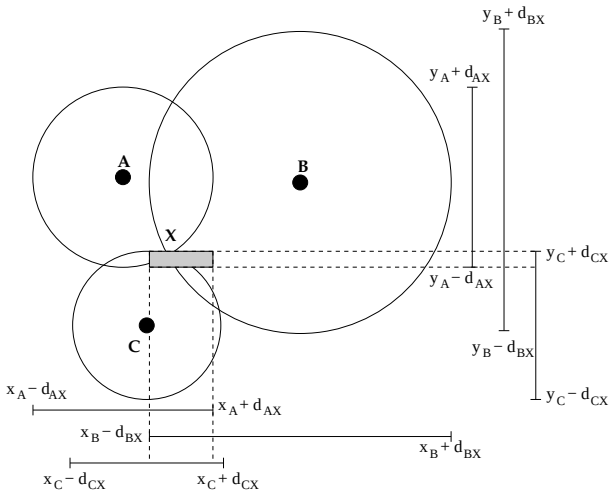

Figure 5 Min-Max

Figure 6 shows the area partitioning and the points to be considered on residue calculation. In (3), the parameters that define the partitioning parameters are shown. The loop that calculates the residues would have N iterations, nx columns and ny rows ($nx \times ny = N$).

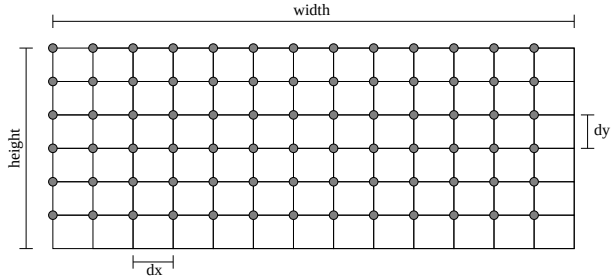

Figure 6 Area partitioning based on the number of testing points

$$nx = \sqrt{\frac{N \times width}{height}} \quad ny = \sqrt{\frac{N \times height}{width}} \quad dx = \frac{width}{nx} \quad dy = \frac{height}{ny} \quad (3)$$

§4 Evaluation

In order to evaluate the location system implemented in this work, and to allow tweaking of parameters in the location algorithm, RSSI measurements were collected in the field, and stored for offline execution. For this, a wrapper was developed to allow running the same code developed for the sensor platforms with the EPOS system in UNIX workstations. Through this wrapper, the same code that would run in a sensor node runs in a thread, and message exchanging is performed through memory copies, using the data collected in the field. Additionally, a graphical tool was developed to allow observing the behavior of the location system.

In order to allow this execution scenario, RSSI measurements were collected between every pair of nodes in a 3×3 sensor grid, with nodes $5m$ apart. These measurements served as input for the workstation wrapper. Three nodes from the nine in total were selected as anchors, and three other nodes estimated their positions. Figure 7 illustrates the geographical disposition of the nodes after stabilization of the estimated positions. The arrows indicate the distance between the estimated positions of nodes A, B and C, and the actual position were they were located. This figure shows that the relative disposition of the nodes' estimated position was maintained from the actual position, which encourages the use of this system in applications which require this characteristic, such as geographical routing.

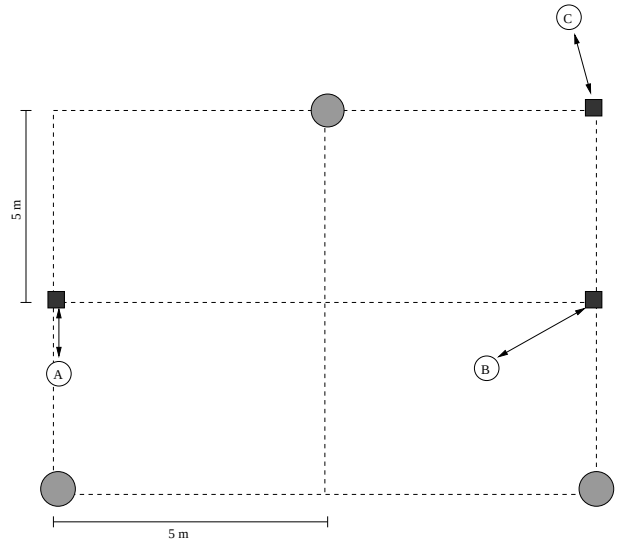

Figure 7 Graphic disposition of the nodes after stabilization

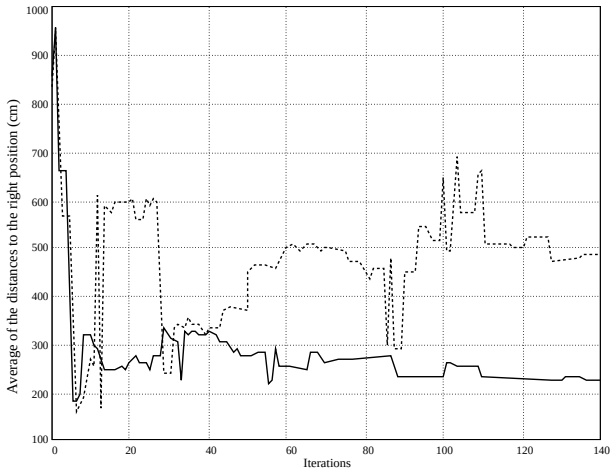

Figure 8 Average distance to the actual positions along iterations

Figure 8 shows the average distance between estimated positions and actual positions along iterations, i.e. the number of times each node has calculated its position. The dotted line shows the behavior of the algorithm in relation to the highly irregular RSSI measurements, which hinder stabilization of the estimated position. In the second curve, a historical average of the RSSI was kept by each node, minimizing the effect of momentary RSSI fluctuations.

The introduction of this historical average may have, as a collateral effect, the invariability of the results along the time, and thus may not be adequate for situations of high mobility, where high variations in RSSI are expected. An alternative for this problem is to keep, in addition to the historical average, an average of the latest measurements and, when the two averages become too far apart, to discard the historical average for the recent average. As the graphic shows, the estimated position stabilizes in the initial iterations, and even if nodes were moving, we would have quick convergence to the new position.

§5 Conclusion

This paper described the implementation and evaluation of the HECOPS location algorithm for Wireless Sensor Networks, using the infrastructure provided by the EPOS operating system. We observed that, even with the restrictions of low cost and low computational power devices, it is possible to use such a location algorithm, as long as some adaptations for performance and accuracy are introduced in the implementation of the location system for real world deployment.

Although RSSI measurements have been discouraged as a reliable distance estimation tool for wireless networks, due to its instability and susceptible to noise and interference, we found that with calibration, cooperative position exchanging, and heuristics, we may obtain good location results based on such measurements without the need for additional hardware.

References

1) Antônio Augusto Fröhlich. *Application-Oriented Operating Systems*. Number 17 in GMD Research Series. GMD - Forschungszentrum Informationstechnik, Sankt Augustin, August 2001.

2) Ricardo Reghelin and Antônio Augusto Fröhlich. A decentralized location system for sensor networks using cooperative calibration and heuristics. In *MSWiM '06: Proceedings of the 9th ACM international symposium on Modeling analysis and simulation of wireless and mobile systems*, pages 139–146, Terromolinos, Spain, 2006. ACM Press.

3) Andreas Savvides, Heemin Park, and Mani B. Srivastava. The bits and flops of the n-hop multilateration primitive for node localization problems. In *Proceedings of the First ACM International Workshop on Wireless Sensor Networks and Applications (WSNA-02)*, pages 112–121, New York, September 28 2002. ACM Press.

4) L.F. Wanner, A.S.H. Junior, A. B. de Oliveira, and A.A. Fröhlich. Operating system support for data acquisition in wireless sensor networks. In *Proceedings of 11th IEEE Conference on Emerging Technologies an d Factory Automation*, pages 582–585, September 2006.

5) L.F. Wanner, A.S.H. Junior, F.V. Polpeta, and A.A. Fröhlich. Operating system support for handling heterogeneity in wireless sensor networks. In *Proceedings of 10th IEEE Conference on Emerging Technologies and Factory Automation*, volume 2, September 2005.

6) Hongyi Wu, Chong Wang, and Nian-Feng Tzeng. Novel self-configurable positioning technique for multihop wireless networks. *IEEE/ACM Trans. Netw.*, 13(3):609–621, 2005.