Simulation Environment for Distance based Location Algorithms in Wireless Sensor Networks

JAGOBA ARIAS, JESÚS LÁZARO, JAIME JIMÉNEZ,
AITZOL ZULOAGA, ARMANDO ASTARLOA
Department of Electronics and Telecommunications
University of the Basque Country
Alda. Urquijo s/n. 48013 Bilbao, Vizcaya
SPAIN

Abstract: - In the last years, a remarkable research effort has been made in the area of wireless sensor networks. One of the main advantages of this kind of distributed system is the ability to determine the position of its nodes and the events sensed. Therefore, a simulation framework is required to compare the different location algorithms that could be used in a certain environment, in order to decide which one is the most suitable method for determining the position of a node or target, as well as to identify the weaknesses and strengths of every algorithm. In this paper, we describe a simulation environment, for distance based location algorithms to be used in wireless sensor networks.

Key-Words: - **wireless sensor networks, DoA, location, simulation, Monte Carlo**

1 Introduction

The evolution of wireless networks has suffered a huge boost in the last years. Due to the availability of low cost and low power RF transceivers, many new applications, where small and inexpensive wireless nodes cooperate to perform complex tasks, have arisen. One of these applications is wireless sensor networks. In this kind of wireless networks, each node senses and takes measurements of a certain physical magnitude, which is geographically distributed. The main objective of this network is to obtain a map of this magnitude, so that it can be monitored (e.g. by a human operator) remotely from a unique location.

However, the cooperation level required for these monitoring tasks involves complex high-level operations, such as packet routing and path finding [1][2], many of which may be terribly simplified if the exact location of all nodes is known. For example, a node could decide if it should forward a packet to one of its neighbors if it is closer to the final destination of the packet than its sender. Furthermore, some of the operations of the wireless sensor network could become unmeaning if the location of the nodes is unknown. Let us suppose that the network must monitor the pressure of a certain gas along a pipe. Just knowing that node 0x3F26 is measuring 2 atm is completely useless if we ignore the position of this node. Some of the applications may even lie strictly on the location of the nodes:

- In a medical facility, a wireless sensor may be useful to locate some members of the medical staff quickly, in order to attend a patient quickly.
- In large storage areas, such as harbors or airports, a location finding scheme could ease the management of the space, by quickly establishing where specific containers are.
- National parks could find the position of a certain specimen quickly within a vast land extension, which would help the authorities protecting the endangered species from furtive hunters and control their demographic evolution.

Although there are some commercial global location systems, such as GPS [3], they are usually based on satellites to provide the service all over the world, which makes the system useless indoors (e.g. hospitals and other facilities) and under foliage, due to the signal absorption produced by roofs and leaves. Therefore, a terrestrial location algorithm needs to be implemented in wireless sensor networks to provide this service.

This paper is organized as follows: section 2 describes the nature of the location problem and the most usual approaches to its solution. In section 3, we describe the requirements of a simulation application for the location process within a wireless sensor network. The conclusions of this work are presented in section 4.

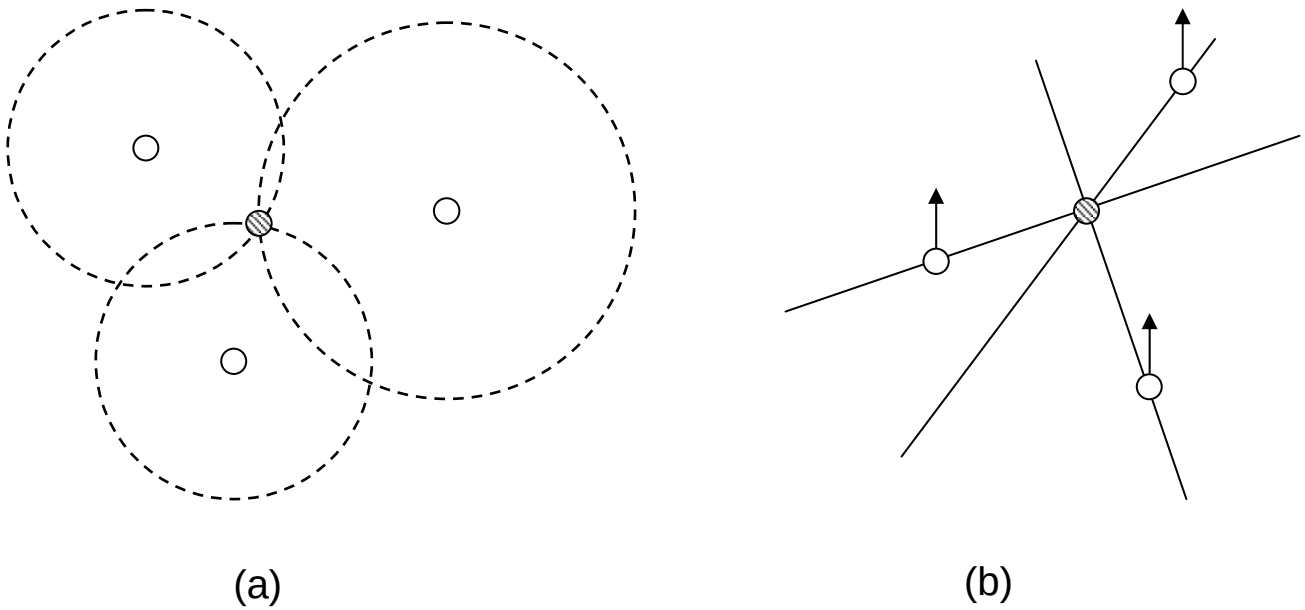

Fig. 1: The two main methods for finding the position of a target. The patterned circles represent the targets and the white ones represent beacons. (a) Distance of Arrival and (b) Angle of Arrival.

2 Problem Formulation

Let us suppose we know the exact position of N nodes (which from here onwards will be referred as beacons) of a certain wireless network and we want to estimate the position of another node (the target). To perform this task, the beacons will emit some kind of signal and the target will measure the characteristics of the received signals. Studying how the environment has affected the propagation of the signal, some information from the geographical location of the target may be obtained:

- If all the nodes have directive emitters and/or receivers, the direction of the target from a certain node may be estimated. If we know these data for a sufficient number of beacons, we may trace lines from each beacon following the correspondent direction. The point where all the lines meet is the position of the target. This location method is known as Angle of Arrival (AoA) [4][5].
- The distance between a beacon and the target affect the reception of the signal in two different ways: it attenuates its power and it produces a delay. Measuring these distances, we may find the position of the target by tracing circles (or spheres) with the center in the position of the beacons and the distances to the target as radii. This location method is known as Distance of Arrival (DoA) [6][7][8].
- If the target is moving respect to the beacons, the frequency measured at the reception of the signal is different from the actual emitted frequency due to the Doppler Effect. This frequency drift depends on the direction of the movement of the target and the relative position of the beacon and the target, which may be used to estimate its position [9][10].

If the wireless network is built from small and inexpensive nodes, the different measurements are expected to contain some error. However, due to the intrinsic nature of wireless sensor networks, where all the nodes cooperate to fulfill their sensing purposes, the effect of this error may be reduced. The simulation platform described in this article will be used to quantify the performance of the different location DoA algorithms for wireless sensor networks, in order to compare them and establish some kind of merit function for them.

The most common schemes to measure the distant between two nodes are the following:

- Measuring the flight time of a signal from one node to the other leads the system to know the distance between the both of them. This requires that the speed of the traveling wave is known and constant, and that the time of flight is large enough to be measured. For example, using RF signals indoors, where the distances are only some meters long, to perform this kind of measurement requires complex receivers, which must be able to measure very short periods of time (of about some nanoseconds) and, in these cases, some other kinds of signals (e.g. sound

signals) are preferred for these kinds of scenarios. These methods are known as ToA (Time of Arrival) [6].

- ToA algorithms usually provide good performance for measuring distances. However, when the distances are short they normally require two different signal processing units: one for the communication (usually RF) and a second one for measuring distances (typically ultrasounds). In these cases, a simple (yet inaccurate) way of estimating distances is to measure the power of the received signal. The longer the distance between the two nodes is, the greater the attenuation will be. This kind of algorithms is called PoA (Power of Arrival) [11][12][13].

DoA schemes are the most commonly used location algorithms, because they require relatively simple hardware: for instance, AoA schemes are based on rotating antennas or electronically configurable antenna arrays.

3 Simulation platform

Every DoA location algorithm needs the same kind of data to compute the position of the target: the number of beacons that have detected the target, their positions and the estimated distances to the target. However, depending on the characteristics of the location algorithms, these data will be corrupted with errors, whose magnitude and features may vary.

The general procedure for estimating the behavior of different location algorithms is the following:

1. Deploy the beacons and the target in the scenario.
2. Calculate the real distances between the target and each node.
3. Corrupt the distances with error, to simulate a real environment.
4. Corrupt the position of the beacons with error.
5. Estimate the position of the target and calculate the location error made.

This procedure must be repeated a sufficiently large number of times, in order to eliminate any kind of effect, due to certain beacon alignment or to the appearance of an unusually high error in distance estimations.

3.1 Scenario description

The chose scenario is a three-dimensional cube, whose side is a simulation parameter. The default value for this dimension is 10 m, which defines quite a big area: if the typical height of a floor in a building is 2.5 m, it corresponds to a 4 floor building, where each floor is100 m^2

The nodes (both beacons and target) are deployed in this scenario at random, following a uniform distribution. No other kind of probability distribution is allowed here, for the following reasons:

- The side of the cube determines exactly the volume of the zone of interest. A normal distribution, for example, could allow that a node (beacon or target) would be placed outside our cube.
- The expected distances among nodes are directly related to the side of the cube. Thus, the relation between the distance estimation error (see section 3.2) and the expected distance between nodes is simple to obtain. This ratio gives an idea of how good a result is, i.e. the relationship between the relative error in distance and the three-dimensional location error.
- There is no *a priori* knowledge of the relative positions of the beacons and the target. In the real world, they could be placed anywhere, so there is no reason for giving a greater preference to a certain zone (e.g. someone could find placing the nodes close to the center of the region reasonable, whilst in some applications, the corners of the regions could give better practical results, due to the effect of environmental factors, such as obstacles, which could reduce the lengths of the wireless links).

3.2 Distance estimation error

The distance estimation error measures the uncertainty in the measure of distance. In the real world, this error has two different components: one of them varies with time and represents the probability of getting two different results when measuring the same magnitude twice using the same method. This kind of error can be reduced simply by performing a large number of measurements and calculating their average value. However, there is a second source of error, which cannot be completely wiped out with repetition: the systematic error. Examples of this kind of error can be found both in ToA and DoA algorithms and are usually produced by the the presence of an obstacle in the surroundings of both bodes. This interference may fake the measurement in two different ways: the wave may reflect against the obstacle increasing the time of flight (in ToA schemes) or the obstacle may attenuate the power of

the signal, making the target seem that it is further than it really is (in PoA systems). In general, PoA schemes are more sensitive than ToA algorithms, because both effects (reflection and attenuation) have a great impact in distance estimation (the reflected wave has a longer path, which implies larger free space losses, some power is lost in the reflection and it will interfere with the LOS wave, altering the measured power).

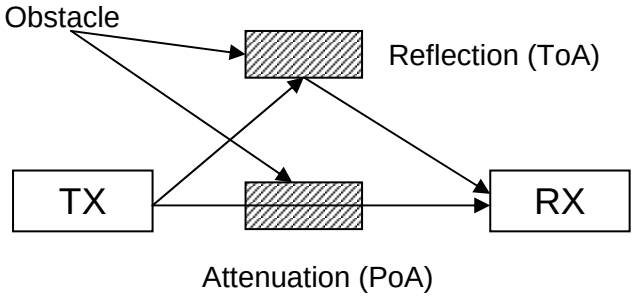

Fig. 2: Possible distance measurement perturbations both for ToA and PoA schemes.

The different sources of error in the estimation of the distance between two nodes produce different stochastic behaviors for this parameter. Depending on the nature of the location algorithm under test, the accuracy of the position estimation may vary according to the probability function of the error. Therefore, the simulation platform offers the possibility of using three different types of error distribution: Gaussian, uniform and lognormal.

The process of polluting the distance estimations with error is simple: in the very first place, the actual distances between the beacons and the target are computed. These values would be the exact result of the estimation process. However, as the presence of a certain amount of error is needed, the distances, which the algorithm under test will use, will be calculated as random numbers, whose average values are the actual distances and their standard deviation may be introduced by the user. Thus, depending on the scenario, different amounts of error rates may be introduced. This allows the user to decide which algorithm is most suitable for an application, where coarse errors are expected.

3.3 Beacon position error

In all location systems, the position of the beacons must be known *a priori*. As distance based location algorithms require that the beacons are placed at known positions, they may be sensitive to some error made in the location of these nodes. If the location algorithm is going to be used recursively in a wireless sensor network, so that after the location of a node that node becomes a beacon to extend the location service throughout a geographically large area, the location error made at some point of the algorithm will make that all the subsequent estimation procedures will use an erroneous datum as premise, thus propagating the error.

The proposed simulation platform allows the user to include two different kinds of error distribution (normal and uniform), which are used to corrupt the real values in an additive way. Thus, the user may test how sensitive an algorithm is to the presence of this kind of error, and decide if it is suitable for a multihop location service without human interaction.

3.4 Location algorithms

The different location algorithms are written independently in C, sharing the same interface. Thus, the simulation platform may be indefinitely updated to suit the needs of users, which want to test their own algorithms or even keep the platform up to date. The prototype of these function should follow this scheme:

```
result algorithm(beacons *b, int
NumberofNodes, double *distances)
```

The implementation of the algorithms must provide two different data: the estimation of the position of the target and an estimation of the required computational cost of that algorithm. The definition of this cost varies depending on the user's interests: for complex and varied algorithms, it may simply be the time required to compute the position or it may be something simpler to extend to other platforms, such as number of iterations, etc.

3.5 Gathering results

The results of all position estimations are gathered and summarized using first and second order statistics. Thus, the mean value of the location estimation error represents the bias of the expected result. Ideally, the average should tend to the null vector, showing that algorithm does not make systematic errors. However, the presence of some systematic error is not really a big problem, because it can be solved easily by correcting artificially the position (by subtracting the mean value of the error vector).

$$\vec{m}_e = \sum_{i=1}^{L} \vec{r}^{(i)}_{est} - \vec{r}^{(i)}_{act} \qquad (1)$$

The second order statistic obtained from the collected results is the standard deviation of the

estimation error. This measures the expected distance of the error vector to its average value. High values of this parameter mean that the location algorithm does not provide accurate results. As a matter of fact, this parameter measures the accuracy of a certain algorithm, which may be used for comparing two different algorithms.

$$d_e = \frac{\sqrt{\sum_{i=1}^{L} \left| \vec{r}_{est} - \vec{m}_e \right|^2}}{L} \qquad (2)$$

Quite often, the behavior of an algorithm as some simulation parameters vary must be stated. This short of simulation capability involves running a large number of simple simulations, where most parameters remain unchanged, and one of them is altered orderly to obtain a function. The simulation results are saved in a file, using an easy to read ASCII format, so that they can be post-processed for plotting using any standard tool.

4 Results

This simulation platform has been tested using three different location algorithms:

- The random algorithm: this trivial location algorithm tries to find the position of the target without using the measured distances. Its behavior is quite simple: it builds the smallest parallelepiped which contains all the beacons. Then, it chooses a position within this region at random, which will be the selected position for the target. As this algorithm does not use the measured distances, its accuracy is independent from the distance error. Although its performance is very poor (selecting a random position for the target is not a terribly sophisticated way of finding its actual position), it is used for comparison purposes: if under certain conditions an algorithm behaves worse than this one, it is not worth implementing it.
- The LSQ algorithm [14]: the relationship between the coordinates of the nodes and the distances among them is non-linear, due to the presence of square roots and some powers. However, for *N* nodes, a set of *N*-1 equations may be obtained, where all the unknown variables (i.e. the position of the target) are related using linear equations. In this case, the system may be solved using a Least Squares algorithm. This algorithm requires a low computational cost, and behaves quite well when the number of beacons is greater than the minimum needed (in a three-dimensional space, 4), but presents a low accuracy when the number of beacons is close to the minimum and the distance estimation error is high.
- The Malguki algorithm [15]: this is a more complex location algorithm, which requires a higher computational cost that the previous ones but it offers better accuracies for low number of beacons and high distance estimation errors.

Fig. 3 shows the results of the simulation of the three test algorithms. The side of the scenario cube is 10 m long, which means that the expected distance between two nodes is about 6.6 m. There are four beacons and the distance estimation error has been varied from 0 m to 10 m. Taking into account the expected distances, the relative error may be as big as 50%. At this point, we can observe the foreseen behavior for every algorithm: the horizontal curve about 6 m is the expected accuracy for the random algorithm. As this algorithm does not use the distances to compute the location estimation, its accuracy does not depend on the quality of the measurements. On the other hand, the LSQ algorithm gives accurate position estimations for low values of the distance estimation error, but its performance is quickly degraded as this error grows. However, the Malguki algorithm degrades gracefully with distance error, reaching the uncertainty level of the random algorithm when the distance estimation error is about 4 m. From that point onwards, the contradictory information puzzles the algorithm, which produces less accurate estimations than choosing the position randomly.

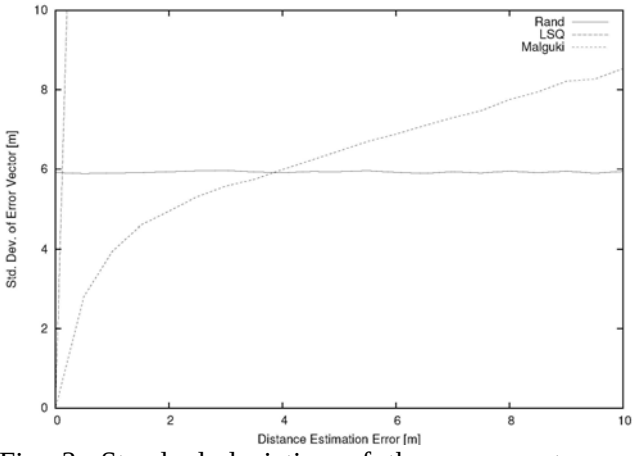

Fig. 3: Standard deviation of the error vector, as distance estimation error grows for the three test algorithms.

Fig. 4 shows the evolution of the computational cost of the same algorithms as the distance estimation error grows. Due to the substantial difference among the test algorithms, the computational cost is represented as the average time to produce a position estimation. Of course, this time varies with the processing power of the computer running the simulation, its load, the amount of memory it has, etc. In this case, the Malguki algorithm, being far more complex than the other two, needs more time to find a solution – but, as we have just seen in the previous graph, it is more accurate.

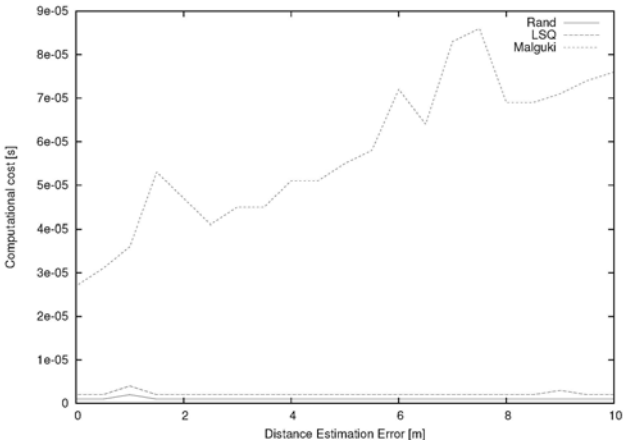

Fig. 4: Computational cost of the test algorithms as the distance estimation error grows.

5 Conclusion

A simulation platform for location algorithms in wireless sensor networks has been presented. This platform provides a flexible and powerful method for comparing different algorithms and their performance in a number of situations: new algorithms may be easily added, which enables its adaptability to future needs, it allows different kinds of distance estimation error distribution, which helps simulating a wide range of distance estimation techniques, etc. The simulation platform has been verified using three different algorithms.

This simulation platform may be used to find the minimum number of beacons required to obtain a certain location accuracy, to test new location algorithms or to evaluate the computational cost of these kinds of algorithms.

References:

[1] J. Li, J. Janotti, D. De Couto, D. Karger, R. Morris, A Scalable Location Service for Geographic Ad-Hoc Routing, *Proceedings of the 6th ACM International Conference on Mobile Computing and Networking (MobiCom)*, 2000, pp. 120-130.

[2] Y. Ko, N. Vaidya, Location Aided Routing (LAR) in Mobile Ad-Hoc Networks, *Mobile Computing and Networking*, 1998, pp.66-75.

[3] B. Hoffmann-Wellenhof, H. Lichtenegger, J. Collins, *Global Positioning System: Theory and Practice*, Springer-Verlag, 1997.

[4] A. Nasipuri, K. Li, A Directionality based Location Discovery Scheme for Wireless Sensor Networks, *First ACM International Workshop on Wireless Sensor Networks and Applications (WSNA)*, 2002.

[5] C. D. McGillem, T.S. Rappaport, A beacon navigation method for autonomous vehicles. *IEEE Transactions on Vehicular Technology*, Vol. 38, No. 3, 1989, pp. 132–139.

[6] N. B. Priyantha, A. Chakraborty, H. Balakrishnan, The Cricket Location-Support System. Mobile Computing and Networking, 2000, pp. 32–43.

[7] A. Harter, A. Hopper, P. Steggles, A. Ward, P. Webster, The Anatomy of a Context-Aware Application. *Mobile Computing and Networking*, 1999, pp. 59–68.

[8] A. Ward, A. Jones, A. Hopper. A New Location Technique for the Active Office, IEEE Personnal Communications, Vol. 4, No 5, 1997, pp. 42–47.

[9] NOAA Satellites and Information. Argos data collection system. *http://noaasis.noaa.gov/ARGOS*.

[10] J. Arias, J. Lázaro, A. Zuloaga, J. Jiménez. Doppler Location Algorithm for Wireless Sensor Networks. *International Conference on Wireless Networks (ICWN)*, 2004, pp. 509–514.

[11] Y. Gwon, R. Jain, T. Kawahara, Robust Indoor Location Estimation of Stationary and Mobile Users. *IEEE InfoCOM*, 2004.

[12] P. Bahl, V. N. Padmanabhan, RADAR: An In–Building RF–Based User Location and Tracking System. *IEEE INFOCOM*, 2000, pp. 775–784.

[13] R. Want, A. Hopper, V. Falcao, J. Gibbons. The Active Badge Location System, *ACM Transactions on Information Systems*, Vol. 10, 1992, pp. 91–102.

[14] L. Doherty, K. S. J. Pister, L. El Ghaoui, Convex Position Estimation in Wireless Sensor Networks, *International Conference of the IEEE Computer and Communications Societies (INFOCOM)*, 2001, pp. 1655-1633.

[15] J. Arias, A. Zuloaga, J. Lázaro, J. Andreu, A. Astarloa, Malguki: an RSSI based ad hoc location algorithm, *Microprocessors and Microsystems*, No. 28, 2004, pp. 403–409.