HAL
open science

MuR : A Distributed Preliminary Method For Location Techniques in Sensor Networks

Clément Saad, Abderrahim Benslimane, Jean-Claude König

To cite this version:

Clément Saad, Abderrahim Benslimane, Jean-Claude König. MuR : A Distributed Preliminary Method For Location Techniques in Sensor Networks. WiMOB'06: Wireless and Mobile Computing, Networking and Communications, Jun 2006, Montréal, Canada. lirmm-00135517

HAL Id: lirmm-00135517

https://hal-lirmm.ccsd.cnrs.fr/lirmm-00135517

Submitted on 8 Mar 2007

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

MuR : A Distributed Preliminary Method For Location Techniques in Sensor Networks

Clément Saad, *LIA /CERI,* Abderrahim Benslimane, *LIA / CERI,* and Jean-Claude König, *LIRMM*

***Abstract*— Wireless sensor nodes need to know their localizations in many control and monitoring applications such as routing, target tracking, etc ... To determine node localization, many techniques have been proposed in the literature. This paper proposes a rule-based method, called MuR (Method using Rules), that allows to locating nodes with high accuracy. The rules are based on information of located nodes (called anchors). They resolve ambiguity when a node can be located at more than one position. MuR is compatible with existing localization techniques; it can be used as a preliminary step before the execution of one of these techniques. Simulation results show the effectiveness of MuR locating a maximum number of nodes.**

***Index Terms*— wireless sensor networks, localization, positionning, accuracy.**

I. Introduction

Ad-hoc wireless sensor networks have been proposed for many applications such as target tracking, intrusion detection, medical applications, climate control, and disaster management. The localization of nodes can be used for routing or others location based services. Some services require high accuracy localization while others do not.

Sensors are small devices, with scarse resources, that can communicate using wireless communication protocols (e.g., 802.11). Each sensor has a perception radius and if another sensor is in its perception then the two sensors are neighbors. An example, often used in ad-hoc wireless sensor network, is the aircraft deployment of sensors in a field. In this network, only some nodes, called **anchors**, know their localizations (either by human intervention or they are equipped with GPS). A maximum number of remaining nodes have to determine their positions based on anchor localizations. The number of anchors has to be the smallest because, for example, nodes which are equipped GPS cost much and consume more energy.

The network has to be self-organizing; the energy being scare, sensors have to minimize their computations and, especially, communications.

Extensive research efforts have been conducted to resolve the localization problem (see Section 3).

The performance, in terms of the number of nodes being located, of all existing solutions is closely related to the number of anchors in the sensor network. The performance improves with the number of anchors. In this paper we propose a rule-based method, called MuR, that increases the number of anchors; we define rules, based on existing anchors, that allows resolving ambiguity when a node has more than one possible position.

Anchors broadcast their positions; each node estimates its distances to anchors. When a rule can be applied, the ambiguity between possible positions for a node is resolved; in this case, the node becomes an anchor and broadcasts its localization. Errors or human intervention (e.g., adverse attack in military context) can impact the efficiency of rules locating nodes; *vote process* to minimize this impact.

It is worth noting that MuR can be used as a first step to existing localization solutions; it allows increasing the number of anchors that are used by existing solutions to locate a maximum of non-anchor nodes.

The rest of the paper is organized as follows. Section 2 describes key existing solutions to the localization problem. Section 3 presents the details of MuR. Section 4 presents experimental results evaluating the performance of MuR. Section 5 concludes the paper and presents future work.

II. Related works

A large number of existing techniques attempts to solve the localization problem.

A detailed survey is provided by Hightower and Borriello in [1].

We can distinguish four categories:

- Infrastructure-based systems: They require infrastructures like RADAR[2] or Cricket[3].
- Robot-based systems. They use robots to locate nodes [4].
- GPS-free methods: They do not require anchors to locate nodes. The authors in [5] propose a method that builds a virtual system of coordinates and the nodes compute their positions in this virtual system.
- GPS-based methods:They use the positions of anchors (equipped with GPS) to determine estimated positions of non-anchor nodes.

Some of the GPS based methods use estimated distance between pair of neighbors. These methods are called Range-Based Localization Schemes (in contrast with Range-Free Localization Schemes [6], [7], [8]).

The most popular methods are RSSI (Received Signal Strength Indicator), ToA/TDoA (Time of arrival / Time difference of arrival)and AoA (Angle of arrival).

Manuscript received January 8, 2006. This work is supported by DGA.

C. Saad is with the Department of Computer Science, Avignon university LIA /CERI, 339 chemin des Meinajaries, BP 1228-84911 Avignon Cedex 9 (phone: +33 467 41 85 85; fax: +33 467 41 85 80, e-mail: saad@lirmm.fr).

A. Benslimane is with the Department of Computer Science, Avignon university LIA /CERI, 339 chemin des Meinajaries, BP 1228-84911 Avignon Cedex 9 (e-mail: benslimane@lia.univ-avignon.fr).

J-C. König is with the Department of Computer Science, Montpellier university LIRMM, 161 Rue Ada, F-34392 Montpellier Cedex 5 (e-mail: konig@lirmm.fr).

In RSSI, nodes measure the power of the received signals. With the power transmission information, the effective propagation loss can be calculated; theorical or empirical models are used to translate this loss into distance.

In ToA/TDoA, nodes translate directly the propagation time into distance if the signal propagation speed is known; the most basic localization system using ToA techniques is GPS [9].

In AoA, nodes estimate the angle at which signals are received and use simple geometric relationships to calculate their positions. The accuracy of these measurements is closely related to the network environment; thus, the positions computed by the nodes may contain errors, called errors of measures.

The classical technique that use these measures to compute the node's position is the trilateration; when a node estimates its distances to at least three anchors, it computes its position with high accuracy if, anchors are its neighbors; otherwise, the position is estimated. For example, let X be a node and A, B, C anchors. X wants to compute its position. It knows distances d_{AX}, d_{BX}, d_{CX} and positions of A, B, C which are respectively $(x_A, y_A), (x_B, y_B), (x_C, y_C)$. the resolution of the following system gives to X its position:

$$\begin{cases} d_{AX}=\sqrt{(x_X - x_A)^2 + (y_X - y_A)^2} \\ d_{BX}=\sqrt{(x_X - x_B)^2 + (y_X - y_B)^2} \\ d_{CX}=\sqrt{(x_X - x_C)^2 + (y_X - y_C)^2} \end{cases}$$

Among the most popular GPS based methods, we can quote have the methods of Niculescu and Nath (APS) [10], Savvides & al. [11] and Savarese & al.[12]. These methods use the same execution scheme; each node estimates its distances to the anchors, computes an estimation of its position (e.g., using trilateration technique), and then performs a refinement process in order to improve the accuracy of estimation.

III. Preliminary

In some applications, nodes may be mobile (for example, [13] considers localization of vehicles without GPS equipment). In this paper, we focus on static networks. Moreover, we assume that all the sensors have identical reachability radius r; however, it is easy to adapt MuR with sensors having different reachability radius. We represent a wireless sensor networks as a graph $G(V, E)$ where V is the set of n nodes representing sensors and E is the set of m edges representing communication links. If two nodes u, v are neighbors, then they are linked and the distance between u and v is smaller than r; G is considered connex.

We assume also that some anchors have a priori knowledge of their own positions with respect to some global coordinate system. Note that all identical nodes (anchors or others nodes) have the same capabilities (energy, processing, communication, ...).

We assume that each node can compute its distance to its neighbor when it receives a signal. Thus, if a node receives a signal from transmitter, it will know that it is located on the disk centered on the transmitter.

Figure 1 represents a network with 12 nodes: 3 anchors (black nodes) and 9 not positioned nodes (white nodes).

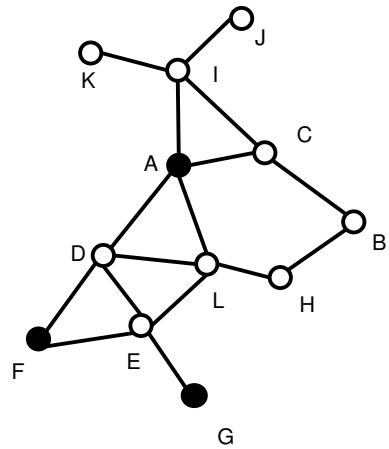

Fig. 1. An example of network

IV. Distance Estimation Techniques

In this section, we describe how a node estimates its distances to anchors. There are three distance estimation techniques: Sum-Dist[11], DV-Hop[12] and Euclidian[10]. In [14], Langendoen and Reijers provide a detailed comparative survey of these techniques; in the three techniques, the anchors start by broadcasting their positions.

A. Sum-Dist

1) Description: This method is the simplest solution for determining distances to anchors. It is adding the ranges encountered at each hop during message propagation. Each anchor sends a message including its identity, coordinates and a path length set of 0. When a node receives this message, it adds the measured range to the path length and broadcasts the message. Thus each node obtains distance estimations and positions of anchors; only the shortest distance, to each anchor, will be conserved. For example, (Figure 2), the estimated distance between S and D is $d_{SY} + d_{YD}$, and we have $d_{SD} \leq d_{SY} + d_{YD}$ due to triangular inequality.

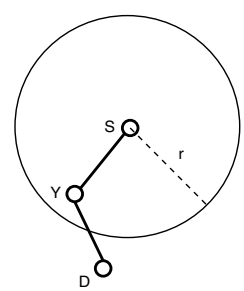

Fig. 2. Sum-Dist

2) Advantage and Drawback: Sum-dist is very simple and fast; it requires few computations. A drawback of Sum-dist is that range errors are accumulated when distance information is propagated over multiple hops.

B. DV-Hop

1) Description: DV-hop consists of two flood waves. Similarly to Sum-Dist, after the first wave, nodes obtain the positions and minimum hop counts to anchors. The second (calibration) wave allows converting hop counts into distances; the conversion is performed by multiplying the hop count with an average hop distance. When an anchor A receives the position of another anchor B during the first wave, it computes the distance between them, and divides it by the number of hops in order to obtain the average hop distance between A and B. A calibrates its distance when it receives the position of anchor. Nodes forward calibration messages (only from the first anchor that calibrates them in order to reduce the total number of messages in the network).

Figure 3 shows an example where A estimates the average of hop distance. There are 3 hops between A and B, and 4 between A and C. A computes Euclidean distance between AB (75m) and AC (125m). The average of hop distance is equal to $\frac{125+75}{3+4} = 28.57m$. Node X estimates distances with B and C as following: $d_{XB} = 2\times28.57$ and $d_{XC} = 3\times28.57$.

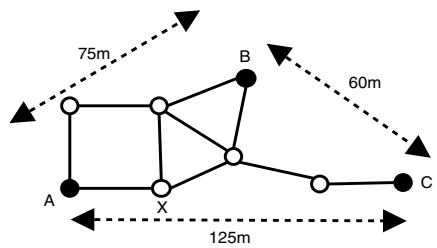

Fig. 3. DV-Hop

.

2) Advantage and Drawback: DV-hop is a stable and predictable method. Since it does not use range measurements, it is completely insensitive to this source of errors. However, DV-hop fails for highly irregular network topologies where the variance in actual hop distances is very large.

C. Euclidian

1) Description: Euclidean is based on the local geometry of nodes around an anchor. When a node contains in its neighborood two nodes having estimated their distances with an anchor then it uses the *neighbor vote* method or *common neighbor* method in order to estimate its distance to the anchor.

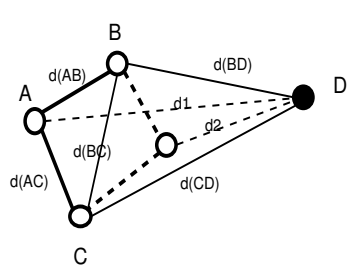

Fig. 4. Euclidean propagation method

Let A, B, C be nodes and D be an anchor (see Figure 4). B and C are neighbors to A. B and C estimate their distances to D. A wants to estimate its distance to D. It knows distances $(d_{AB}, d_{AC}, d_{BC}, d_{BD}, d_{CD})$. So, all the sides and one of the diagonals of quadrilateral $ABCD$ are known; the second diagonal corresponds to d_{AD}. In this case, there are two solutions d_1, d_2. The *neighbor vote* method or *common neighbor* method allows selecting the distance d_1 or d_2; more details can be found in cf. [10].

2) Advantage and Drawback: Euclidian provides an exact distance with anchor whenever possible. However, Euclidean is sensible to range errors and is efficient only in highly connected networks; otherwise, Euclidean's performance rapidly degrades.

Rules of MuR uses lower and higher bounds for each real distance between nodes which are not neighbors. The lower bound is equal to r since for nodes that are not neighbors the real distance between them is higher than r. The higher bound is computed using Sum-Dist. Indeed, the sum of distances between two nodes is higher than the real distance between them using triangular inequality. Therefore, we chose for MuR to use Sum-Dist to estimate distances.

V. MuR (Method using Rules)

In MuR, each anchor broadcasts its position. When a node receives the position of an anchor it estimates distance to this anchor using Sum-Dist. We define three rules that allow resolving the ambiguity when there are two possible positions for a node. For example, in Figure 5, X does not know its position and B, C are anchors belonging to X's neighborhood. X knows positions of B and C and distances between XB and XC. So X can be located at node A or node A'. When one of the rules can be applied, X will know whether it is located in A (ie. (x_A, y_A)) or in A' (ie. $(x_{A'}, y_{A'})$).

However, range errors or human interventions (for example, in military context during an attack) can perturb the efficiency of rules and imply errors. The *voting process* allows to manage errors. We describe this process in this section.

We assume that the real position of X is A. Hereafter, d_{XY} represents the real distance between nodes X and Y, and $\hat{d}_{XY}$ represents the estimated distance between nodes X and Y computed using Sum-Dist.

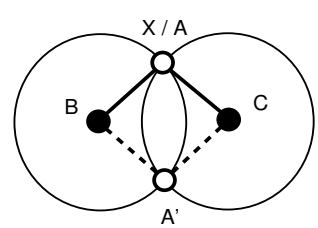

Fig. 5. X can be in A or in A'

A. Rule 1

This rule defines a bound for the estimated distance, from a node to an anchor, using Sum-Dist. The anchor is not a

neighbor of the node trying to compute its position.

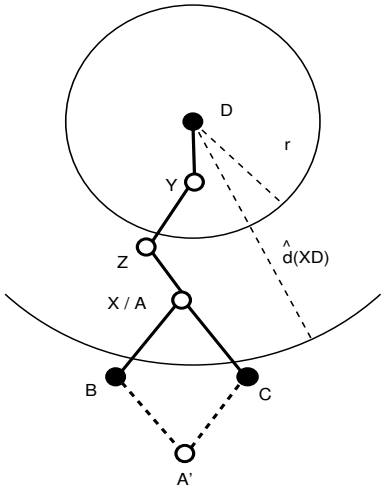

Fig. 6. Rule 1

Let X be the node looking for its position and D an anchor which is not a neighbor of X (Figure 6). X receives $D's$ position (ie. (x_D, y_D)) and learns its estimated distance to D (ie. $\hat{d}_{XD}$). Let us assume that X is in A; we must have $d_{AD} > r$ (First condition); otherwise, A and D will be neighbors. We also must have $d_{AD} \leq \hat{d}_{XD}$ (due to triangular inequality; $\hat{d}_{XD}$ is the sum of distances between nodes separating X and D); this represents the second condition. If the two conditions are satisfied then X may be (but not necessarily) in A.

Now, let us assume that X is in A'. If one of two conditions is not satisfied then X cannot be in A'; thus, it should be in A. However, if all conditions (for A and A') are satisfied then X cannot be certain about its position.

B. Rule 2

The authors in [15] propose a method using the anchors that are one and two hops neighbors of a node to resolve, whenever possible, the node's positions ambiguities. Rule 2 is based on the same principle but generalized to use all anchors.

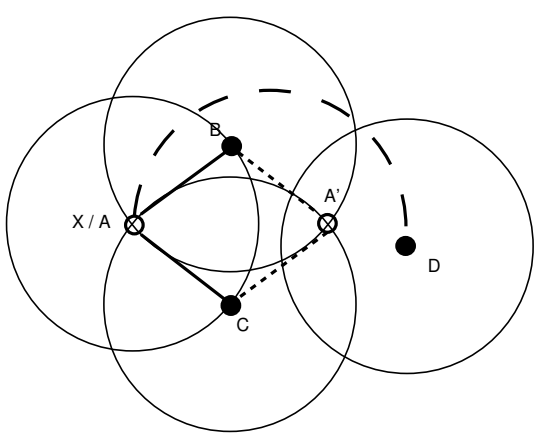

Fig. 7. Rule 2

Let X be the node looking for its position and D an anchor which is not a neighbor of X (Figure 7). When X receives $D's$ position (i.e. (x_D, y_D)), it checks whether $d_{A'D} \leq r$; if the response is yes, then A' and D are neighbors. Therefore, X concludes that it is not in A'; if $d_{AD} > r$ then X concludes that it is in A. Neverseless, if $d_{AD} > r$ and $d_{A'D} > r$ then X cannot determine its position.

C. Rule 3

This rule is applied when a node has at least three anchors in its neighborood.

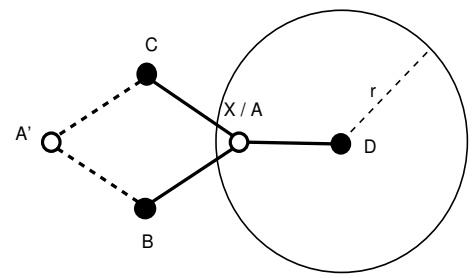

Fig. 8. Rule 3

Let X be the node looking for its position and D an anchor which is a neighbor of X (Figure 8). When X receives $D's$ position (i.e. (x_D, y_D), it checks whether $d_{A'D} > r$; if the response is yes, then A' and D are not neighbors. Therefore, X concludes that it is not in A'; if $d_{AD} \leq r$ then X concludes that it is in A. Neverseless, if $d_{AD} \leq r$ and $d_{A'D} \leq r$ then X cannot determine its position.

The trilateration is not recommended, even if a node has at least three anchors in its neighborhood, because it is very sensible. When we use trilateration, the position of a node corresponds to a point in the intersection of the three circles centered in the positions of the three anchors. If one of the distances with anchors is wrong or if one of neighbors is not accurately located, then the node may not be able compute its position. The second case is shown in Figure 9; the localization of node D is slightly inaccurate: D is located in D'. The circle centered in D' does not intersect in the same point with the two others circles. It is easy to see that with our rule this problem is resolved. Thus, when range errors are equal to 0% then the trilateration can be used; otherwise, new techniques are required.

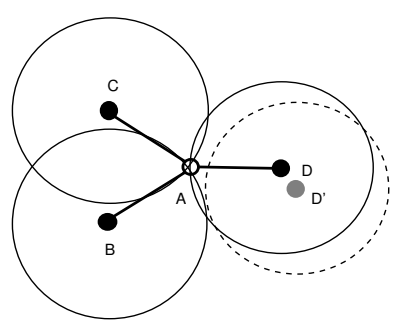

Fig. 9. Trilateration cannot be applied

In the following, we present a technique, called *voting process*, that allows minimizing the impact of range errors in the localization process and thus increasing the efficient of MuR.

D. voting process

Ideally (e.g. without range errors), when a node receives an anchor position and when one of rules can be applied then the node resolves potential ambiguity and determines its position. However, if the anchor related information (e.g., position and estimated distance) is not accurate due to an error of measurement or an attacker who has the control of a node, the localization process may be in jeopardy. For better understanding, let us consider the example shown in Figure 10.

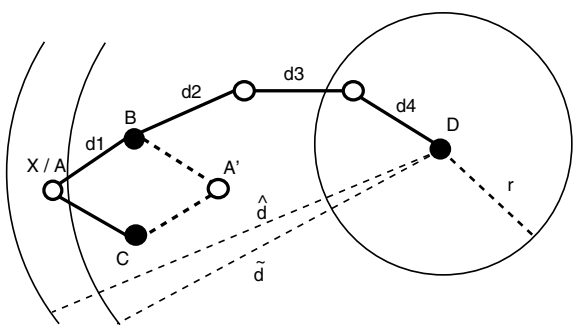

Fig. 10. Error of Sum-Dist due to range errors

Let $\hat{d} = d_1 + d_2 + d_3 + d_4$ be the result of Sum-Dist between nodes D and X without range errors and $\tilde{d} = \tilde{d}_1 + \tilde{d}_2 + \tilde{d}_3 + \tilde{d}_4$ the result of Sum-Dist with range errors ($\tilde{d}_1, \tilde{d}_2, \tilde{d}_3, \tilde{d}_4$ being distances with range errors). We assume that $\tilde{d} < \hat{d}$. We can see in Figure 10 that without range errors, X cannot resolve ambiguity with anchor D since no rule can be applied. If we consider range errors, the estimated distance is $\tilde{d}$ and then X determines that it is located in A' using Rule 1; this finding is correct. Therefore, X cannot rely on one anchor to determine its position when considering range errors.

We propose to use a technique, called *voting process*, that makes use of more than one anchor to determine the positions of non anchor nodes. When a node can be located at two positions p_1 and p_2, it checks the applicability of the MuR's rules with all anchors it knows. When a rule can be applied with an anchor allowing to determine the position of node in p_1 (resp. p_2), then the node increments a counter cp_1 (resp. cp_2). Now, if $cp_1 - cp_2 \geq threshold$ (resp. $cp_2 - cp_1 \geq threshold$), then the node is located in p_1 (resp. p_2). Note that without range errors the $threshold$ is equal to 1.

E. Algorithm

Algorithm 1: Process when u receives an anchor's position from v (a neighbor of u)

Data : Reception of message $< v, a, (x_a, y_a), \hat{d}_{va} >$, where v is a neighbor of u, a is the anchor, (x_a, y_a) the coordinate of a and $\hat{d}_{va}$ is the estimated distance between v and a

Result: rules test

/* $sumdist(\hat{d}_{va})$ returns the estimated distance between u and a taking into account the estimated distance between v and a */
/* $ambiguity()$ returns the two possible positions (p_1, p_2) for u, when it can be at some positions. */
/* $rule1()$ returns position p for u if rule 1 can be applied */
/* $rule2()$ returns position p for u if rule 2 can be applied */
/* $rule3()$ returns position p for u if rule 3 can be applied */
/* $assignPosition(u, p)$ assigns position p to node u */
/* $N(u)$ is the set of u's neighbors */
/* $Anchors$ is the set of anchors known by u */
/* cp_1, cp_2 are counters related to p_1 and p_2 */

/* Update of distance between u and anchor a. It is important for rules */
$\hat{d}_{ua} \leftarrow sumdist(\hat{d}_{va})$;
$(p_1, p_2) \leftarrow ambiguity()$;
if $p_1 \neq null$ *and* $p_2 \neq null$ *and* $u \notin Anchors$ **then**
  if $rule1() = p_1$ **then**
    $cp_1 \leftarrow cp_1 + 1$
  if $rule1() = p_2$ **then**
    $cp_2 \leftarrow cp_2 + 1$
  if $rule2() = p_1$ **then**
    $cp_1 \leftarrow cp_1 + 1$
  if $rule2() = p_2$ **then**
    $cp_2 \leftarrow cp_2 + 1$
  if $rule3() = p_1$ **then**
    $cp_1 \leftarrow cp_1 + 1$
  if $rule3() = p_2$ **then**
    $cp_2 \leftarrow cp_2 + 1$
  if $|cp_1 - cp_2| \geq threshold$ **then**
    if $cp_1 - cp_2 \geq threshold$ **then**
      $assignPosition(u, p_1)$;
    if $cp_2 - cp_1 \geq threshold$ **then**
      $assignPosition(u, p_2)$;
    $Anchors \leftarrow Anchors \cup u$;
    for $v \in N(u)$ **do**
      $send < u, u, (x_u, y_u), 0 >$;

Each anchor broadcasts its position. When a node receives an anchor position, it executes the algorithm 1. It computes the estimated distance from the anchor to itself. Then it applies the 3 localization rules with each known anchor and it executes the voting process in order to obtain its position, whenever possible. When a node has more than two anchors in its neighborhood, it randomly chooses two and then determines its position.

When a node is located it becomes an anchor and broadcasts its position. A node stops when it has positions of all anchors and no rule can be applied.

VI. Simulations

A. Simulation environment

We used the simulator created by Langendoen and Reijers [14] based on OMNET++; it is a discrete event simulator [16]. This simulator allows using the three distance estimation techniques; in our simulation, we configured the nodes to use only Sum-Dist. Concurrent transmissions are allowed if the transmission areas (circles) do not overlap; when a node wants to broadcast a message while another message in its area is being transmitted, it must wait until this transmission is completed. The simulator uses CSMA policy. In our simulations, all messages are delivred.

At the beginning of simulation, we generate a random network topology according to the number of nodes and the number of anchors. The nodes are randomly placed, using uniform distribution, within a square area; the anchors are selected randomly. The range between connected nodes is blurred by drawing a random value from a normal distribution having a parameterized standard deviation and having the real range as the mean. The simulator makes use of this error model; it is based on the work of Whitehouse and Culler [17]. The connectivity (average number of neighbors) is controlled by specifying the radio range. For easier comparison between different scenarios, range errors as well as errors on position estimates are normalized to the radio range. For example, a 50% position error means a distance of half the range of the radio between the real and estimated positions.

We use different scenarios while changing the number of nodes. For our analysis, each scenario is performed 100 times. The number of experiments is sufficient since the gap between experimentations is relatively small.

We analyzed different scenarios using three topologies with 100, 150 and 225 nodes respectively. The nodes are distributed in a square 100×100. Thus, we can evaluate the behavior of MuR in an environment with moderate and high density. The range radio is set to 18, 16 and 14 respectively. We consider range errors of 0%, 5% and 10% respectively.

B. Results

We show graphics related to only the second scenario (i.e., network of 150 nodes). But we comment also the two others scenarios with 100 and 225 nodes.

Figure 11 shows the behavior of MuR with a network containing 150 nodes. Each graphic contains four curves. Each curve represents the percentage of nodes located (without take into account the anchors) (axis Y) with a percentage of errors respectively smaller than 1%, 5%, 10% or 15% according to the percentage of anchors in the network.

Figure 11, with 0% of range errors, shows that only 8% of the network nodes as anchors is necessary to locate all nodes with an error smaller than 1%. For a network of 100 nodes and 225 nodes, only 6% and 5% respectively of the nodes as anchors is necessary to achieve the same results. To the best of our knowledge, there is no other method, in the open literature, that achieves similar results.

The others graphics show that the introduction of range errors degrades the performance of MuR; however, the number of nodes being able to compute theirs positions with some errors remain considerable. For example, in the network with 150 nodes and a percentage of range errors equal to 5% (second graphic in Figure 11), with only 10% of the nodes as anchors, MuR locates half of nodes with a maximum of range errors of 15%. Figure 11 (third graphic) shows that with a percentage of range errors equals to 10% only 10% of the nodes as anchors is necessary to locate the third of nodes with at a maximum of errors of 15%.

The simulation results shows that the more the connectivity is high, the more MuR is sensible to range errors. This is due to Sum-Dist; in an environment with high connectivity, the number of nodes separating two nodes increases; thus, the increase of range errors degrades the performance of Sum-Dist. We believe that it would be interesting to compute errors and thus reduce the impact on Sum-Dist. For example, the authors in [18] propose a technique to compute errors of measures in ToA. Moreover, MuR can locate nodes with error precision bigger than 15% when the range error rate is high. Protocols, using MuR in preliminary step, must be able to detect these ”imprecise” anchors while observing the inconsistency of their information, contrary to the ”precise” anchors.

Finally, we checked the efficiency of *voting process* and, without this process in our scenario with 150 nodes and range errors equal to 10%, the number of anchors reduced at least of 30%.

We evaluated the performance of each rule by executing the simulations using only rule 1, rule 2, and rule 3 respectively. The results show that the three rules are important especially rule1; using Sum-Dist, rule 1, sometimes, allows increasing the number of anchors to 100% (i.e., locating all the nodes).

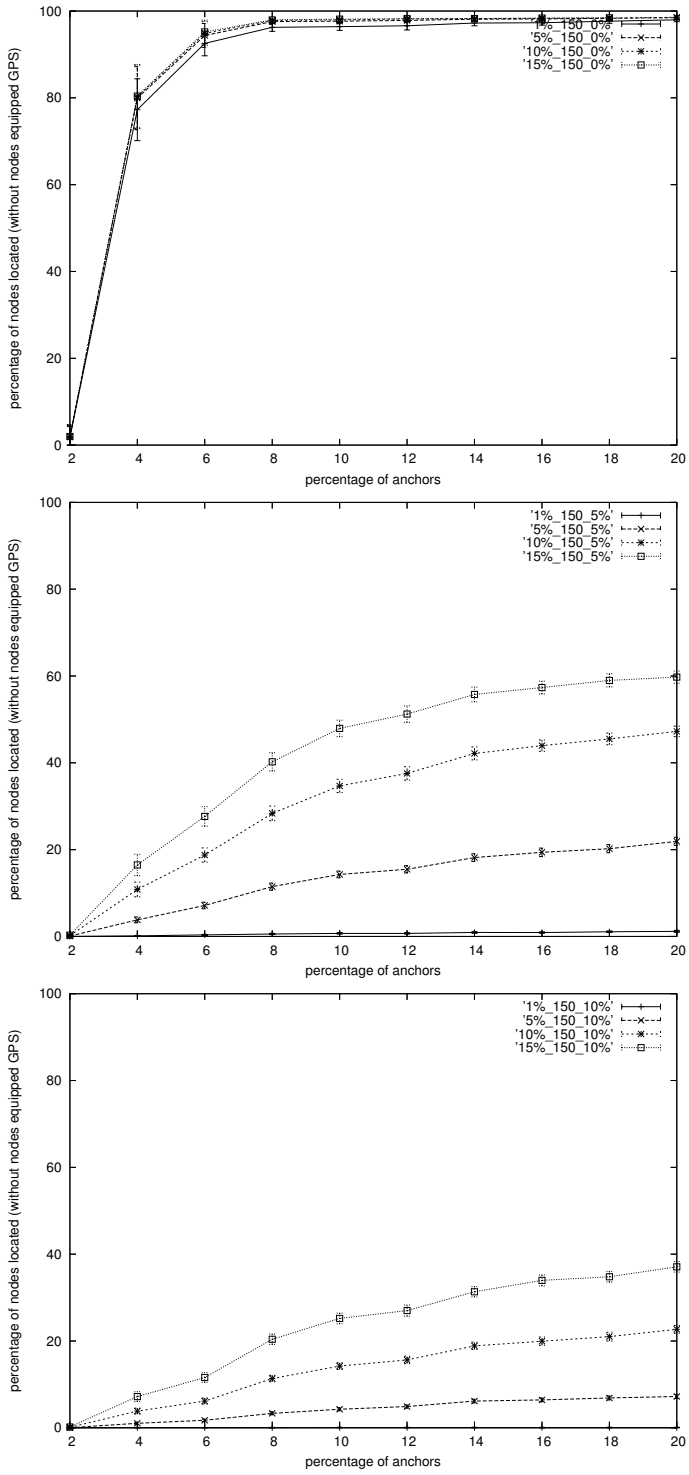

Fig. 11. A network with 150 nodes and range errors equal to 0%, 5% and 10%

VII. Conclusions

In this paper, we proposed a new rule-based technique, called MuR, which increases the number of anchors in a sensor network using estimation of distances. We defined three rules using the localization of anchors in order to resolve ambiguity when a node can be located at more than one position. We proposed a *voting process* that minimizes the impact of range errors or human attacks.

MuR can be used as a preliminary step by existing localization techniques; indeed, the performance of these techniques is closely related to the number of anchors in the network.

Simulation results show the superiority of MuR. In some cases, MuR can locate all nodes in the network. The simulator, we used, does not take into account all conditions in a real sensor network; it will be interesting to check the efficiency of MuR in a sensor network testbed; we are in the process of setting such a testbed using crossbow sensors.Interesting parameters to investigate include to localize nodes, etc.

MuR does not always determine positions for all nodes: either a node is located or it does not know its position. We are investigating techniques to extend MuR to determine or estimate positions for all the nodes.

Acknowledgment

We graciously acknowledge K. Langendoen, T. Parker and O. Visser for their simulator and for answering to our questions related to their simulator. Also, we thank A. Hafid for his careful reading and comments.

References

[1] J. Hightower and G. Borriello, "Location systems for ubiquitous computing," *IEEE Computer 34(8) pp. 57-66*, August 2001.

[2] P. Bahl and V. N. Padmanabhan, "Radar: An in-building rf-based user location and tracking system," *in: INFOCOM pp. 775-784*, 2000.

[3] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan, "The cricket location-support system," *International Conference on Mobile Computing and Networking, Boston*, 2000.

[4] N. Bulusu, J. Heidemann, and D. Estrin, "Adaptative beacon placement," *The Twenty First International Conference on Distributed Computing Systems, Phoenix*, 2001.

[5] S. Čapkun, M. Hamdi, and J.-P. Hubaux, "Gps-free positioning in mobile ad-hoc networks," *in Proceedings of the 34th Hawaii International Conference on System Sciences*, 2001.

[6] T. He, C. Huang, B. M. Blum, J. A. Stankovic, and T. Abdelzaher, "Range-free localization and its impact on large scale sensor networks," *IEEE Personal Communications Magazine*, 2005.

[7] D. Niculescu and B. Nath, "Dv based positioning in ad koc networks," *Journal of Telecommunication Systems*, 2003.

[8] N. Bulusu, J. Heidemann, and D. Estrin, "Gps-less low cost outdoor localization for very small devices," *IEEE Personal Communications Magazine*, October 2000.

[9] B. P. et al., "Global positioning system: Theory and application," *Progress in Astronomics and Aeronotics*, vol. volume I, 1996.

[10] D. Niculescu and B. Nath, "Ad hoc positioning system (aps)," *in IEEE GLOBECOM 2001, San Antonio*, November 2001.

[11] A. Savvides, H. Park, and M. Srivastava, "The bits and flops of the n-hop multilateration primitive for node localization problems," *in: First ACM International Workshop on Wireless Sensor Networks and Application (WSNA), Atlanta, GA, pp. 112-121*, 2002.

[12] C. Savarese, J. Rabaey, and K. Langendoen, "Robust positioning algorithms for distributed ad-hoc wireless sensors networks," *in: USENIX technical annual conference, (Monterey, CA), pp. 317-328*, June 2002.

[13] A. Benslimane and A. Bachir, "Inter-vehicle geocastprotocol supporting non equipped gps vehicles," *in: LNCS 2865 ADHOC-NOW 03 Int. Conference on Ad-Hoc, Mobile and Wireless Networks, Montréal, Canada, Springer Publisher, pp. 281-286.*, October 2003.

[14] K. Langendoen and N. Reijers, "Distributed localization in wireless networks : A quantitative comparison," *Computer Networks, no 43, pp. 500-518*, 2003.

[15] M. Barbeau, E. Kranakis, D. Krizanc, and P. Morin, "Improving distance based geographic location techniques in sensor networking," *3rd International Conference on AD-HOC Networks & Wireless*, July 2004.

[16] A. Varga, "The omnet++ discrete event simulation system," *European Simulation Multiconference (ESM'2001), Prague, Czech Republic*, 2001.

[17] K. Whitehouse and D. Culler, "Calibration as aparameter estimation in sensor networks," *First ACM International Workshop on Wirless Sensor Networks and Application (WSNA), Atlanta, GA pp. 59-67*, 2002.

[18] S. Venkatraman, J. J. Caffery, and H. R. You, "Location using los range estimation in nlos environments," *in: IEEE VTS Spring VTC 2002, 2, pp. 856-860*, 2002.

Clément Saad received the M.Sc. degree in Computer Science from Montpellier 2 University, France, in 2005. He is currently working towards a doctoral degree at Avignon University in France, in the area of wireless sensor networks. He focus especially on positionning and routing problems.

Abderrahim Benslimane Abderrahim Benslimane is Professor of Computer Science and Engineering at the University of Avignon, France. He has been as Associate Professor at the University of Technology of Belfort-Montbeliard. He obtained the title to supervise research (HDR 2000) from the University of Cergy-Pontoise, France. He received the PhD degree (1993), DEA (MS 1989) from the Franche-Comte University of Besanon, and BS (1987) from the University of Nancy, all in Computer Science. His research and teaching interests are in wireless ad-hoc and sensor networks. Particularly, he works on multicast routing, inter-vehicular communications, quality of service, energy conservation, localization, intrusion detection and MAC layer performance evaluation. He was also interested in specification and verification of communication protocols, group communication algorithms and multimedia synchronization. He has several refereed international publications (book, journals and conferences) in all those domains. He is the header of Computer Networks and Multimedia Applications team of the Computer Laboratory of Avignon. He is involved in many national and international projects. He is IEEE member and member of the CA of the IEEE French section. He participates to the steering and the program committee of many international conferences. Currently, he is in sabbatical year for research at the Department of Computer Science, Ecole Polytecnique de Montreal.

Jean-Claude König is 46 and he is professor since 1992 and at the Montpellier University for eight years. He manages a search department of 450 members (Computer Sciences, Electronic, Microelectronic, robotic ...). His works concern the algorithms for network communication and the scheduling theory.