Deployment of Sparse Sensor-Actuator Network in a Virtual Architecture

Sébastien Faye, Jean Frédéric Myoupo
Université de Picardie-Jules Verne, UFR Sciences, 33 rue Saint Leu, 80039 Amiens France
{sebastien.faye@etud.u-picardie.fr, jean-frederic.myoupo@u-picardie.fr}

Abstract: The use of a virtual architecture in wireless sensor networks provides a powerful and fast partitioning into a set of clusters, each being locatable in space by the sink or base station (BS for short). We first propose a protocol for the detection of empty clusters due to poor distribution of sensors or in low-density network. After this detection, we propose a strategy to allow mobile sensors (actuators) to move to position themselves on empty clusters in order to improve the routing of collected data.

Key Works: Wireless Sensor-Actuator Networks,
Virtual Architecture.

1. INTRODUCTION

The wireless sensor networks (WSN) are from the family of mobile ad-hoc (MANET), but have additional features and constraints: typically, they consist of a wide range of sensors with limited energy capacity. Each sensor is powered from a battery non-rechargeable and non-replaceable ([2]) and has a low capacity in terms of memory, calculation (CPU), and transmission range in wireless. Each sensor is able to harvest a set of data in a certain environment, and transmit it in multi-hop way to a base station (BS) or sink, which may act as instructor of the network. The use of such networks is widespread in many applications: for example we can cite the monitoring of forests, critical infrastructure, or the detection of biochemical agents and in the military industries. Some examples of work can be found through [1], [2], [5], [10].

Technology related to sensors advancing day by day, it is common to see WSN composed of several thousand units to tens of thousands ([6], [12]). In large networks, the sensors can be grouped into clusters based on their proximity to propose a better management and data transmissions to be made, in order to significantly increase the scalability, economy energy, routing, and consequently the lifetime of the network. The structure provided by the use of clusters allows the use of various techniques to improve the quality of a WSN, such as data aggregation which consumes as much as the calculation procedure ([7]).

In this paper, we focus on the concept of virtual architecture developed by Wada et al. in [11]. The latter is created and orchestrated by BS, who is able to split the network into a set of clusters, depending on the strength and direction of the broadcast it can perform. We focus on the case where the network is sparse leaving many empty clusters formed in the virtual architecture, contrary in [11] where the WSN is assumed to be dense. Once detection has occurred, we introduce the notion of actuators (sensors that have the ability to move), to consider positioning strategy of actuators onto empty clusters.

The rest of the paper is organized as follows: Section 2 presents the model of our network and specifically the virtual architecture. The protocol for the detection of empty clusters is described in section 3. The strategy of feeding the empty clusters by the actuators in presented in section 4. Section 5 describes how to move the actuators onto the empty clusters. The simulation results are presented in section 6. And a conclusion ends the paper.

2. VIRTUAL ARCHITECTURE

Consider a set of anonymous sensors distributed around a master node BS. Figure 1 illustrates an example of such a network. In this WSN, each sensor has the ability to collect data, perform calculations, store data temporarily, and to transmit and receive information within a certain range R that does not cover the entire network because of energy constraints and cost. The transmission of information from one sensor to BS is therefore usually done in multi-hop. The BS has the opportunity to transmit information to some powers (to the sensor farther away).

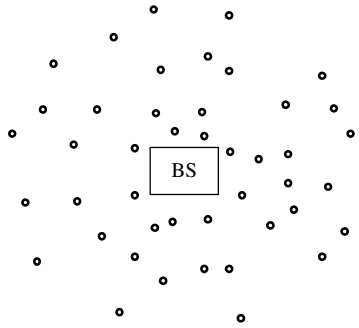

Figure 1. Example of a basic WSN

The virtual architecture proposed in [11] consists of a partition of the WSN into different zones by the BS: the BS has the opportunity to disseminate information from the lower to the higher range in order to create coronas. It has the opportunity to disseminate information in certain directions as [11] to create different angular sectors. The area (i, j) is the intersection of the corona i with the angular sector j. The sensors of the same area are therefore in the same geographical location and form a ***cluster***. This is illustrated in Figure 2. Other works on this type of architecture can be found in [3], [4] and [9].

On this basis, the authors in [11] define a number of mechanisms, such data aggregation, routing, and generally training coordinates. We will not detail these operations here. Interested readers can find them in the original article [11]. Here we focus only on sparse WSN in which some clusters can be empty. In such networks the routing is optimal, and the BS is not able to detect to detect innately these empty clusters.

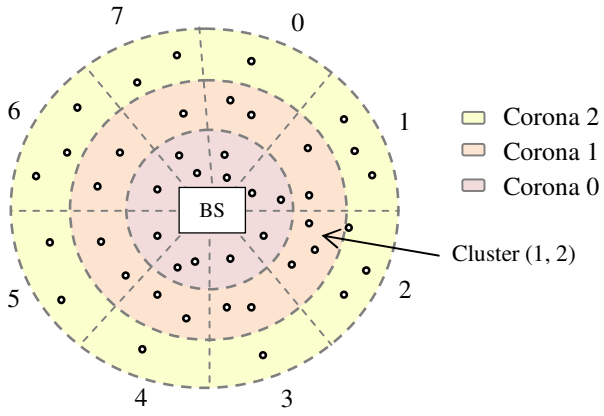

Figure 2. An virtual architecture representation.

Here we propose to equip him with a protocol to detect these empty clusters.

3. DETECTION OF EMPTY CLUSTERS

3.1 Notations

To clarify the result of our paper, we use the following notations below:

- (i,j) : cluster of corona i and angular sector j
- WSN* : the set of sensors-actuators
- ACT : the set of actuators
- s1, s2, … sn : slots time.

3.2 Assumptions

Now we assume that the WSN is clustered as in [11] as in Figure 3 below.

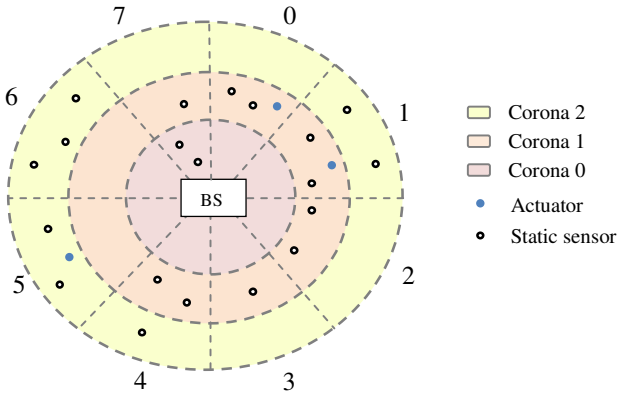

Figure 3. A sparse virtual architecture.

We consider the following assumptions:

1. We assume that the network of WSN* \ {ACT} is static, composed of anonymous immobile nodes once deployed. Optionally, adding and deleting nodes (for fault tolerance) is allowed, but considered rare. In contrast, ACT represents mobile nodes, which have an identity, and whose objective is to improve the connectivity of the network (for example).
2. The network topology is known to any sensor, but the density of nodes allows us to assume that WSN* \ {ACT} is connected, as well as all WSN*.
3. The BS characteristics: BS has the opportunity to disseminate information across the network more or less powerfully (i.e., at different ranges), and at different angular sectors (one-way broadcast).
4. Time is divided into a set of slots s1, s2, ... sn and the local clock of each sensor is synchronized with the one of the BS. In this way, each sensor is awake and listening to the network for each of these slots, but is put to sleep the remaining time to save energy overall network.
5. We consider that the BS and the actuators have a notion similar direction: it is necessary to allow BS to manage the movements of actuators.

3.3 Description of the detection protocol

Phase1: *Initialization of the detection by the BS*

In the first slot s0 all sensors-actuators are awake. And the BS initiates the construction of the breadth search tree. For this, it broadcasts at range R_{bs} (corresponding to the first corona of the virtual architecture) a message of network discovery noted Build(zone) to all the sensors WSN*. WSN being connected, we are assured that at least one sensor receives the information. Here, Build(zone) = Build((-1, -1)), where (-1, -1) represents the area consisting solely of BS.

Phase 2 (recursive): *construction the breadth search tree*

This phase is repeated as s0 is not exceeded, and has 4 steps:

Step 1. All actuators that had no heard any message of construction during s0, when receiving the message Build(zone) they simply return a receipt to BS, stating their position for future movements.

Step 2. When receiving the message Build(zone), all static sensors that have never received a message of construction during s0, are able to temporarily (the time slot s0) store Build.zone, corresponding to the cluster where the sensor (sender of the message) is located (father's sensor).

Step 3. When receiving this information static sensors will then perform two actions:

a. If their cluster is different from the one of the message of construction received, then send an acknowledgment backwards to BS, directly or in multi-hop through the cluster Build.zone. This message looks like ACK(zone, parent), zone representing the cluster from which the acknowledgment is sent, and parent information Build.zone registered by the

sensor. In receiving this message the BS is able to complete his vision of the tree width. This step can be represented through the figure 4.

b. Continue to spread the message of tree construction by sending the message Build(zone) at a transmission range R (where R is the transmission range of a sensor, which we consider to be homogeneous).

When these first two phases are completed and slot s0 is expired, BS has an overview of the network and the tree width of the zones of virtual architecture installed on it, like the example shown in Figure 4 and figure 5.

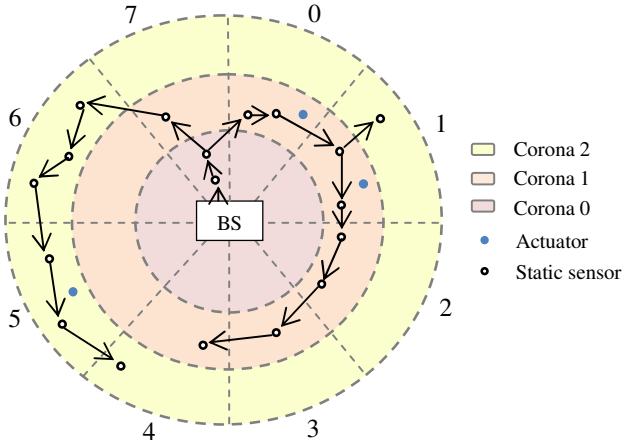

Figure 4. Construction of the breadth search tree.

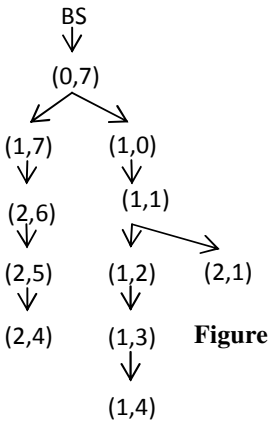

Figure 5. The breadth search tree.

Phase 3: *Detection of empty clusters*
BS being informed of all clusters of the virtual architecture of the network (he is the designer), he is able to detect empty clusters i simply browsing the generated tree.

4. STRATEGY OF FEEDING THE EMPTY CLUSTERS BY THE ACTUATORS

The empty clusters are now known to BS. We detail a strategy that could allow BS to place actuators on these empty clusters in order to gather and route eventual data collected on these clusters. Of course, this strategy is just a sample of application, and the user may choose to apply a priority policy, by choosing for example a part of the actuators for the collection of information in critical areas (empty clusters) that are not equipped with stationary sensors. Our approach consists of two main phases: the first (detailed below) improves the tree width obtained above, in order to reduce the number of hops required for certain nodes to communicate with BS. Second, we may use the remaining actuators to improve communication in some areas with small populations.

Phase 1: Improvement of the breadth search tree
Let H be the height of the graph obtained in section 3. Consider C+1 where C is the number of coronas used by the virtual architecture (example in Figure 4: we have H = 6, and C = 4). The purpose of the first phase is to improve the value of H, reducing it until it reaches - if possible H = C - (requires a sufficient number of actuators). So here we apply a strategy of feeding the empty cluster in terms of the number of actuators available and known to BS.
At this point, two cases arise:

Step 1. If at the start we have H = C, then the application of this phase is not necessary because the routing is optimal we can directly proceed to Phase 2.

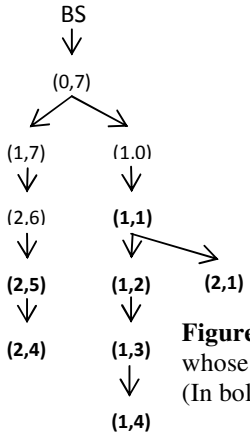

Figure 6. Detection of clusters whose access can be improved (In bold on the tree)

Step 2. Otherwise, we have to determine the priority empty clusters among all the empty clusters of our virtual architecture. The priority empty clusters are those that we must first feed (by an actuator) in order to improve the height H of the tree. To do this, BS calculates from the tree the C - 1 lists: each list number i denoted L[i] contains the list of corona j ε L[i] corresponding to clusters (i, j) whose access can be improved. A cluster (i, j) to which access can be improved (by positioning correctly an actuator) is such that i+1 is strictly below the level of (i, j) in the tree. This is illustrated through Figure 6.

From this list, we can easily deduce the priority empty clusters: consider the cluster (i, j) such that j ε L[i], *the*

priority empty clusters are the empty clusters (m, j) with m<i. This is illustrated through Figure 7.

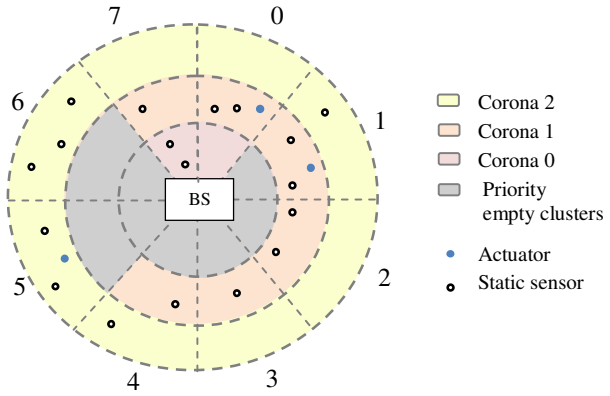

Figure 7. Detection priority empty clusters.

5. MOVING THE ACTUATORS TO THE EMPTY CLUSTERS

It is necessary to provide to an actuator in cluster (i, j), all information that will help it to move correctly to the empty cluster (i2, j2). BS has an overview of the network and knows the parameters used by the virtual architecture. One of its roles is then to calculate and transmit the movements to be performed by each actuator, using simple formulas from trigonometry. Below, we detail the various calculations that BS must do to move an actuator from (i, j) to (i2, j2).

We assume that BS knows the angle α of an angular sector and the scope β of a corona. Let A be a virtual point that we consider to be the center of the cluster (i, j). Let B be a second virtual point considered the center of (i2, j2).

5.1 Computation of the distance Df between A and B:

- Distance from BS to A : $|AD| = \beta \times i + \frac{\beta}{2}$

- Distance from BS to B : $|BD| = \beta \times i2 + \frac{\beta}{2}$

- Let [BS,0) be the half straight line starting from BS and following the left end of the angular 0. The angle formed by [BS,0) and [BS,A] is : $A.R = \alpha \times j + \frac{\alpha}{2}$

In the similar way [BS,B] : $B.R = \alpha \times j2 + \frac{\alpha}{2}$

- Therefore the angle between [BS,A] and [BS,B] is :
R1 = MAX(A.R, B.R) – MIN(A.R, B.R).

- The distance from A to B is given by the following formula:

$$Df = \sqrt{A.D^2 + B.D^2 - 2A.D \times B.D \times \cos(R1)}$$

5.2 Computation of the angle to send A in the direction of B:

- Let [A, 0) the half-line parallel to [BS, 0) but starting from A (assumption (5)). Here we look for the angle between [A, 0) and [A, B], denoted by Rf.
- Here consider only the horizontal axis: Let [0, A] be the segment of length DA from A to the line [BS, 0), perpendicular to [BS, 0). Let [0, B] be a similar segment of length DB, in the case of point B.

$$DA = A.D \times \sin(\Delta A)$$

$$DB = B.D \times \sin(\Delta B)$$

Dx is the angle at x of the rectangular triangle thus formed (formed by BS, x, and the point to the segment [BS, 0]), taking the following values:

If x.R < 90 Then Δx = x.R
Else If x.R < 180 Then Δx = 180 - x.R
Else If x.R < 270 Then Δx = x.R - 180
Else Δx = 360 - x.R

- Let us consider now the rectangular triangle of hypotenuse [A, B]. Set Dtmp the length of its adjacent side having the end point A as:

 # If an point is on the left side of BS and another on its right side
If (A.R > 180 and B.R < 180) or (A.R < 180 and B.R > 180) then Dtmp = DA + DB
 # If both A and B are on the same side de BS
Else Dtmp = MAX(DA,DB) – MIN(DA,DB)

- We can then easily calculate the angle between the adjacent side of [A, B], say R2.

$$R2 = \cos(\frac{Dtmp}{Df})$$

- Rf is then determined by adjusting R2 depending on the position of A relative to B. Several cases are possible. These cases are summarized algorithmically as follows:

If (A.R < 180 and B.R > 180) Then *# A on the left BS and B on the right*
    If (A.R > 360 – B.R) Then Rf = 270 + R2 *# A below and on the right of the line of B*
    Else Rf = 180 + R2
Else (B.R < 180 and A.R > 180) Then *# A on the right of the line BS, B on its left*
    If (B.R > 360 – A.R) Then Rf = 90 + R2 *# A above and on the left of B*
    Else Rf = 90 - R2

Else (B.R < 180 and A.R < 180) Then *# A and B on the left side of BS*

If (A.D > B.D) Then *# A on the left of B*

If (A.R > B.R) Then Rf = 90 R2 *# A is above B*

Else Rf = 90 – R2

Else *# A on the right of B*

If (B.R > A.R) Then Rf = 270 + R2 *# A is below B*

Else Rf = 180 + R2

Else *# A and B are on the right side of BS*

If (A.D < B.D) Then *# A on the left of B*

If (A.R < B.R) Then Rf = 90 + R2 *# A is above B*

Else Rf = 90 – R2

Else *# A is on the right of B*

If (B.R < A.R) Then Rf = 270 + R2 *# Ais below B*

Else Rf = 180 + R2

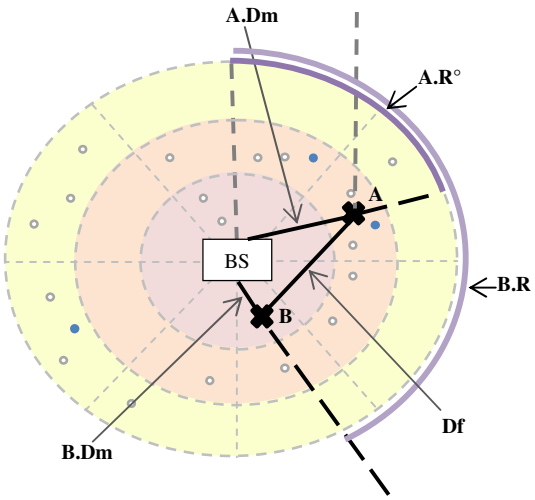

Figure 8.a. Illustration of variables and calculations used for the placement of actuators: calculation the distance between A and B.

These calculations can be illustrated by the example in Figure 8a and 8b, where an actuator placed in (1,1) must go to (0.3): BS sends an instruction to the actuator located at (i, j) to move on a distance Df with an angle Rf° to the north common with BS (assumption (5)).

Example: For α = 45° and β = 30m : A.R = 67.5, A.D = 45, B.R = 157.5, B.D = 15, DA ~= 41.5, DB ~= 5.74, Dtmp ~= 35.8, R2 ~=40°. This yields Df ~=47m and Rf ~=230°.

The actuator then moves on 47m following a path of 230° to the direction north that it shares with BS (assume that the start is the angular sector 0).

Once the actuator placed it sends an acknowledgment to BS containing the area where it is located. BS checks that everything coincides and regularly asks for the actuator's status and position, in order to guide it or replace it if something goes wrong. Conversely, the actuator can send alert to BS messages in case of problems.

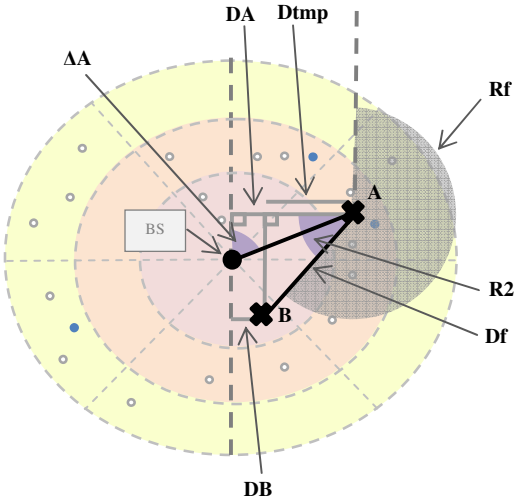

Figure 8.b. Illustration of variables and calculations used for the placement of actuators: calculation of the angle.

6. SIMULATION RESULTS

Here we focus on percentages of empty clusters that may be present in the virtual architecture described above. Our simulations were performed using the software WSNet [13] and is based on circular area of 1km, 4km and 9km in diameter, on which we randomly generated connected network of 500, 1000, 1500, 2000 and 2500 sensors. We take here an ideal case, where BS is located at the center of this circular area. Figure 9 illustrates the simulations obtained from a virtual architecture with 6 coronas and an angular sector of 30 °. Figure 10 illustrates the simulations results obtained from a virtual architecture with 4 coronas and an angular sector of 45 °. Each point of the various curves is the average result of 10 simulations

As we can see on the curves below there are empty some clusters on virtual architecture considered especially in low-density network. If we take the example in Figure 9 where the angular sectors are 30 ° and with 6 coronas, we see almost 80% of empty clusters in the case where 500 sensors are spread over a circular area of 9km in diameter. However, this percentage slows down to 60% in figure 10. So there is a need to manage the existence of empty clusters: the detection and the implementation of strategies with potential sensors and actuators. Furthermore, note that the simulations were

performed under ideal circumstances: the area of sensors is circular, the BS is at the center, and we do not take into account environmental constraints. It appears evident that these points are rarely met (the BS can be at the network edge, rectangular area of sensors, ...).

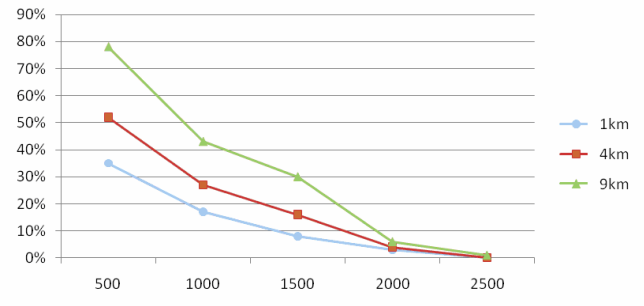

Figure 9. Average percentage of the number of empty clusters based on the number of sensors, for 3 zones of different sizes. Here, we have 6 coronas and angular sectors of 30° each.

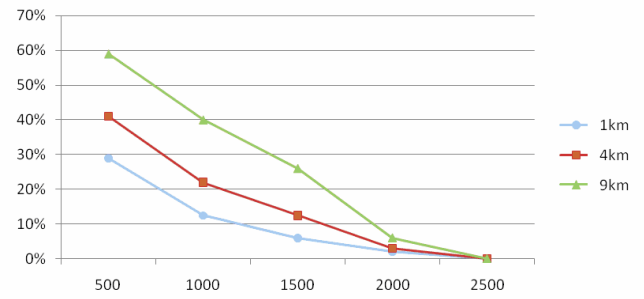

Figure 10. Average percentage of the number of empty clusters based on the number of sensors, for 3 zones of different sizes. Here, we have 4 coronas and angular sectors of 45° each.

7. CONCLUSION

In this paper we have presented a protocol to identify the empty clusters in a virtual architecture of a WSN. Next we have studied few strategies and methods to use actuators, and move on these empty clusters. We have shown how actuators can be guided by the BS to position themselves on desired empty clusters. However we could have also assumed that each actuator is equipped with GPS that should facilitate the management of the movements. Similarly we could have assumed that each actuation is equipped with APS [8] protocol.

In future work, it would be interesting to study such problems with more strong conditions: for example by removing the assumption of connectedness and reducing the density advantage of the network, we could explore the possibility of using the actuator nodes to allow some connected components to reach BS.

REFERENCES

[1] J. Agre and L. Clare, An integrated architecture for cooperative sensing networks, *IEEE Computer* 33(5) (2000) 106-108.

[2] I. F. Akyildiz, W. Su and Y. Sankarasubramaniam. Wireless sensor networks: a survey, *Computer Networks* (38), pp. 393-422, 2002.

[3] F. Barsi, A.A. Bertossi, F. Betti Sorbelli, R. Ciotti, S. Olariu and M. C. Pinotti Asynchronous Training in Wireless Sensor Networks. *Proceedings of the 3rd international conference on Algorithmic aspects of wireless sensor networks,* pp. 46-57, 2007

[4] A.A. Bertossi, S. Olariu, and M.C. Pinotti. Efficient Training of Sensor Networks. *ALGOSENSORS*, pp. 1-12, 2006

[5] C. Intanagonwiwat, R. Govindan and D. Estrin, Directed diffusion: A scalable and robust communication paradigm for sensor networks, in: *Proc. MOBICOM'00*, Boston, MA (August 2000).

[6] J.M Kahn, R.H Katz and K.S.J. Pister, Mobile networking for Smart Dust, in: *Proc. MOBICOM'99*, Seattle, WA, August 17-19 (1999).

[7] C. Karlof, N. Sastry and D. Wagner. TinySec: A Link Layer Security Architecture for Wireless Sensor Networks. *SenSys04*, November 35, 2004.

[8] D. Niculescu and B. Nath, "Ad hoc positioning system (APS),_ *Proceedings of IEEE GLOBECOM*, pp. 2926-2931, San Antonio,TX, November 2001.

[9] S. Olariu, A. Wada, L. Wilson, and M. Eltoweissy. Wireless Sensor Networks: Leveraging the Virtual Infrastructure. IEEE Network, vol. 18,pp. 51-56, 2004

[10] C.-C. Shen, C. Srisathapornphat and C. Jaikaeo, Sensor information networking architecture and applications, *IEEE Personal Communications* (October 2000) 16-27.

[11] A. Wadaa , S. Olariu , L. Wilson , M. Eltoweissy , K. Jones, Training a wireless sensor network, *Mobile Networks and Applications*, v.10 n.1-2, p.151-168, 2005.

[12] B. Warneke, M. Last, B. Leibowitz and K. Pister, SmartDust: communicating with a cubic-millimeter computer, *IEEE Computer* 34(1) (2001) 44-51.

[13] http://wsnet.gforge.inria.fr