Comparing the Efficiency of Localization Algorithms with the Power-Error-Product (PEP)

Frank Reichenbach, Jan Blumenthal, Dirk Timmermann, *Member IEEE*
Institute of Applied Microelectronics and Computer Engineering,
University of Rostock, Warnemuende, Germany
Email: {frank.reichenbach;jan.blumenthal;dirk.timmermann}@uni-rostock.de

Abstract

The localization process in wireless sensor networks must produce precise positions at minimal requirements. It is hard to comply with these conditions, due to the strong resource limitation of each sensor node. Although numerous algorithms have already been presented, so far there exist neither an extensive simulative comparison of them nor a qualified parameter to indicate their efficiency. Thus, this paper introduces a selected number of the most important algorithms for sensor networks. These algorithms are compared with each other considering achieved localization error and consumed energy. Then, we propose an adequate parameter to quantify their efficiency - the "Power Error Product" (PEP).

1 Introduction

Hundreds of tiny electronic devices, able to sense the environment, compute simple tasks and communicate with each other, form a large wireless sensor network (WSN). Gathered information (e.g. temperature, humidity etc.) is transmitted in a multi hop fashion over direct neighbors to a data sink, where the data is interpreted [1]. With methods like self configuration and -organization the network reacts to node failures. WSNs enable new possibilities for e.g. timely detection of wood fire, monitoring of artificial dikes along a river, and precision farming. Due to the desired nodes size of only some millimeters, the dimensions of the communication module and the battery are critical. Consequently, the scarcest resource within a network is the available energy. Therefore, achieving a long lifetime of the sensor network requires low power hardware and power aware algorithms. After deploying the sensor network over an area of interest, initially the sensor nodes have no position information. However, the nodes position is important, because (i) measurements without a location where they were gathered are generally useless, (ii) fully localized sensor networks enable energy aware geographic routing, (iii) self configuration and -organization are key mechanisms for robustness and can easily be supported with position information, and (iv) in many applications the position itself is the information of interest like target tracking. Due to the high energy consumption and the increase of a nodes size, the Global Positioning System (GPS) can not be integrated on hundreds of energy constrained sensor nodes. For this reason a small number of existing localization methods assume that some sensor nodes already know their own position. Here, these nodes are called beacons. By measuring distances and/or angles to beacons a position can be estimated with a suitable localization algorithm.

With only few exceptions localization methods require observations to deduce distances from. These observations are gathered by different measuring techniques like time of flight, received signal strength, and differences of the phase. Signals can be either radio, infrared or ultra sound. Reading the received signal strength indicator (RSSI) is supported by almost every transceiver-hardware that makes this to a beneficial solution.

This paper is subdivided as follows. In Section 2 we classify important localization algorithms in WSNs. Then, in Section 3, we briefly describe these algorithms in detail. Next, in Section 4 the "Power Error Product" is introduced. After we described our used simulation tool and its modifications in Section 5, extensive simulations regarding the localization error and the energy consumption are shown in Section 6. Finally, this paper is concluded in Section 7.

2 Algorithm Classification

We classify localization algorithms into *exact* and *approximate* that mainly depends on their complexity and achieved precision. Most of the approximate localization algorithms need only few resources, but estimate a position with a relatively high localization error. Different approximate (also called coarse-grained) localization approaches exist in the literature. For example, Tian et al. completely avoid distances in their approach [2]. In another approach by Bulusu et al. every sensor node calculate the centroid as its own position [3]. This approach was extended with distances in form of weights, which further improved the precision [4]. Lastly, a very simple idea is to take the nearest beacon position as position estimate.

In contrast, exact (or fine-grained) localization of a sensor node features high precision and is based on solving either (i) linear systems with only the minimum required number of beacons or (ii) a high amount of beacons and distances to them, which leads to optimization problems. In more detail, with at least three beacons (in two-dimensions), sensor nodes estimate their positions via trilateration. More beacons than required result in an over determined system of equations that must be solved with e.g. a Least Squares Method (multilateration). Although a multilateration produces precise results, it is complex and resource-intensive and without optimizations not feasible on resource-limited sensor nodes.

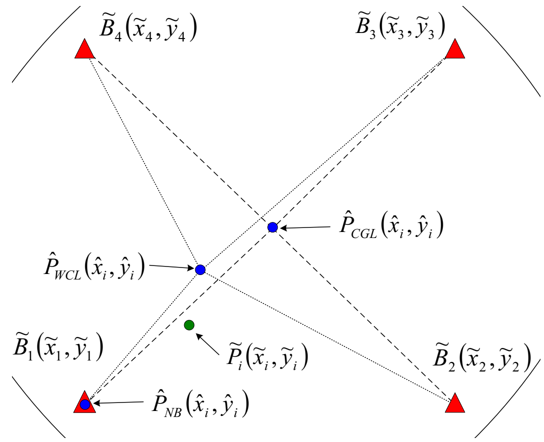

Figure 1. Exemplary localization with NB, CGL and WCL if four beacons are in range of an unknown point (here with exact position $\widetilde{P}_i(\tilde{x}_i, \tilde{y}_i)$); the circles at every corner are the sending range of the beacons

3 Algorithm Descriptions

This section describes the most important localization algorithms for sensor networks. Here, we assume b as the number of beacons and s the number of sensor nodes. Moreover, the localization error f_i is defined as distance between exact position $\widetilde{P}(\tilde{x}, \tilde{y})$ and approximated position $\widehat{P}(\hat{x}, \hat{y})$ of sensor node i:

$$f_i = \sqrt{(\tilde{x}_i - \hat{x}_i)^2 + (\tilde{y}_i - \hat{y}_i)^2} \qquad (1)$$

3.1 Nearest Beacon

Simply determining the nearest beacon (NB) in range by e.g. the highest received signal strength measurement is a robust approach to estimate an approximate position. It seems to be clear that the localization error can become very high depending on how far the next beacon is placed. However, this coarse estimate can be the start value for a refinement process, where the complexity strongly depends on a "good" initial position.

3.2 Coarse Grained Localization

The Coarse Grained Localization (CGL) was first proposed by Bulusu [3]. The beacons $\widetilde{B}_1 \ldots \widetilde{B}_b$ are deployed in a grid-aligned network (infrastructure case) with constant distances d to each other. Fig. 1 demonstrates an example of such a sensor network with four beacons. Prior to localization, all sensor nodes do not know their own position. During positioning, beacons periodically send broadcast messages containing their own position. All sensor nodes within transmission range of beacons receive these messages and store the beacon positions. Nodes that do not have beacons in range are not able to determine a position and therefore are called unknowns. Each sensor node i within beacons range estimates its position based on the well known centroid formula, where n is the number of received messages:

$$\widehat{P}_i(\hat{x}_i, \hat{y}_i) = \left(\frac{1}{n} \sum_{j=1}^{n} \widetilde{B}_j(\tilde{x}_j), \frac{1}{n} \sum_{j=1}^{n} \widetilde{B}_j(\tilde{y}_j) \right) \qquad (2)$$

3.3 Weighted Centroid Localization

Weighted Centroid Localization (WCL) also assumes already deployed beacons, but contrary to the before described CGL algorithm all beacons can be distributed randomly and not on a grid. In the first phase all beacons broadcast their position $\widetilde{B}_j(\tilde{x}_j, \tilde{y}_j)$ to all sensor nodes within their transmission range. While receiving packets, every sensor node measures the received signal strength and stores it as well as the beacon position. After all positions are gathered, the sensor node estimates its approximate position $\widehat{P}_i(\hat{x}_i, \hat{y}_i)$ by a *weighted* centroid determination with all n beacons in transmission range:

$$\widehat{P}_i(\hat{x}_i, \hat{y}_i) = \left(\frac{\sum\limits_{j=1}^{n} w_{ij}(d) \cdot \widetilde{B}_j(\tilde{x}_j)}{\sum\limits_{j=1}^{n} w_{ij}(d)}, \frac{\sum\limits_{j=1}^{n} w_{ij}(d) \cdot \widetilde{B}_j(\tilde{y}_j)}{\sum\limits_{j=1}^{n} w_{ij}(d)} \right) \tag{3}$$

WCL uses distance information as a weight w_{ij}. Small distances to neighboring beacons lead to a higher weight than to remote beacons. Further, every coordinate of a beacons position obtains a weight depending on the distance $w_{ij}(d_{ij})$. The calculation of a weight by an RSSI-value and their optimization is described detailed in [4].

3.4 Approximate Point in Triangulation

This algorithm is well known as the "Range Free Localization" by He et al. [2]. It is based upon generating all possible triangles $\frac{b!}{3! \cdot (b-3!)}$ with the available beacon positions. Every sensor node checks on which triangle surface it is placed by a "Approximate Point in Triangulation"-Test (APIT-Test). For details of this complex process it must be referenced to [2]. With all successfully detected triangles on which the sensor node is placed, the resulting intersection is calculated. This allows narrowing down the field surface to a smaller surface. Finally, a centroid determination with all remaining beacons is completed that estimates an approximate position.

3.5 Atomic Multilateration

One of the most cited localization methods is the "Fine Grained Localization" by Savvides et al. [5]. In this paper we will explain a similar version of this algorithm that we call the "Atomic Multilateration" (AM). This algorithm requires at minimum three beacons in the Euclidean room. The calculation is based on the Least Squares Method with an overdetermined system of equations.

The procedure is the following: all sensor nodes measure the distances to all available beacons in transmission range. With the resulting distances and the Euclidean formula, where $\widehat{P}_i(\hat{x}_i, \hat{y}_i)$ is the estimated position of the i-th sensor node and $\widetilde{B}_j(\tilde{x}_j, \tilde{y}_j)$ is the exact position of the j-th beacon, we can calculate the localization error:

$$f_i(\hat{x}_i, \hat{y}_i, c) = ct_{ij} - \sqrt{(\tilde{x}_j - \hat{x}_i)^2 + (\tilde{y}_j - \hat{y}_i)^2} \tag{4}$$

Here, c is the speed of an ultrasonic signal in air and t_{ij} is the time that the signal needs from sensor node i to beacon j. With minimum three beacons we get a non linear system of equations that we linearize, square and reconvert to:

$$-\tilde{x}_j^2 - \tilde{y}_j^2 = \hat{x}_i^2 + \hat{y}_i^2 + \hat{x}_i - 2\tilde{x}_j + \hat{y}_i - 2\tilde{y}_j - c^2 t_{ij}^2 \tag{5}$$

By subtracting the last equation $i = h$ from all others this leads to:

$$-\tilde{x}_j^2 - \tilde{y}_j^2 + \hat{x}_h^2 + \hat{y}_h^2 = 2\tilde{x}_i(\hat{x}_h - \tilde{x}_j) + 2\hat{y}_i(\hat{y}_h - \tilde{y}_j) + c^2(t_{jh}^2 - t_{ij}^2), \tag{6}$$

This can be written in the form $A\mathbf{x} = \mathbf{b}$ and solved by the Least Squares Method that leads to $\mathbf{x} = (A^T A)^{-1} A^T \mathbf{b}$. Finally, the position vector can be solved on every sensor node completely distributed.

3.6 Distributed Least Squares

In the following, we explain our new algorithm. Distributed Least Squares localization (DLS) starts with the creation of an over-determined system of non-linear Euclidean distances of the form $(\hat{x}_i - \tilde{x}_j)^2 + (\hat{y}_i - \tilde{y}_j)^2 = r_{ij}^2$ with $j = 1, 2, \ldots, b$ (where b is again the number of beacons, $\widehat{P}_i(\hat{x}_i, \hat{y}_i)$ is the estimated position, $\widetilde{B}_j(\tilde{x}_j, \tilde{y}_j)$ is the position of beacon j and r_{ij} is the distance between them). This system of equations is linearized with a term extension (see [6]). This leads to the form $A\mathbf{x} = \mathbf{b}$, where A is the coefficient matrix, $\mathbf{b}$ is the right side vector and $\mathbf{x}$ is the solution vector. By applying the Least Squares Method we obtain the known Normal Equation $\mathbf{x} = \left(A^T A\right)^{-1} A^T \mathbf{b}$.

The matrices in this equation have two important benefits. First, all elements in the coefficient matrix A are generated by beacon positions $\widetilde{B}_1(\tilde{x}_1, \tilde{y}_1) \ldots \widetilde{B}_b(\tilde{x}_b, \tilde{y}_b)$ only. By assuming that we can establish communication links between all sensor nodes and all beacons (e.g. by multi-hop-techniques), then matrix A is the same on every sensor node. Second, the vector $\mathbf{b}$ contains the distances between sensor node i and beacons $r_{i1} \ldots r_{ib}$ that must be estimated on every sensor node independently. Given these facts, the Normal Equation can be split into two parts - a more complex part, the *precalculation*: $A_p = \left(A^T \cdot A\right)^{-1} \cdot A^T$ and a simple part: $A_p \cdot \mathbf{b}$, which is further called the *postcalculation*. Here, the *precalculation* is executed on one powerful base station, which additionally avoids high redundancy, because this *precalculation* would normally be executed on all sensor nodes separately. But it is very important to emphasize that the *precalculation* is identical on every sensor node. Thus, it is calculated only once, conserving expensive energy resources. The simple *postcalculation* is then executed on every sensor node with its individual distance measurements to all beacons. This approach complies with two important design strategies for algorithms in large sensor networks - a resource-aware and distributed localization procedure. Finally, this can be achieved with less communication overhead than it is required for other exact algorithms.

4 How to compare algorithms efficiency?

Both, the localization error and the energy that was consumed to get the position are essential to quantify the efficiency of a localization algorithm. In mobile networks, the time in which the position was estimated is an additional parameter. However, in this paper we focus on static networks, where error and energy are the dominating parts. For that, we define the "Power Error Product" (PEP), which was derived from the well known power delay product used by hardware developers. Like the name already presumes the PEP can be calculated with $PEP = P \cdot E$, where P is the power and therefore the energy that was consumed by the network. Moreover, E stands for the mean localization error that can be calculated by averaging all partial errors f_i of all nodes: $1/s \sum_{i=1}^{s} f_i$. If the localization algorithms under test are simulated in one simulator tool with the same configuration and thus under the same assumptions, the PEP allows a fair comparison.

5 Simulating Localization Algorithms

Numerous simulation tools for wireless sensor networks are available. Well known simulation platforms are NS2, OMNeT++, J-Sim, SENSE, and Shawn. We decided to use J-Sim as simulation framework, which was developed by the University of Ohio. For our purposes, we needed to modify J-Sim by replacing some components to include our own energy model. This new model supports power consumption when processing data on a node, which was not possible yet. Further, we integrated hardware dependent power consumptions on bases of the MICA-2 system. In the following, we describe the energy model in more detail.

5.1 New Energy Model

We added a more complex and more realistic energy model respectively. Our energy model considers (i) power-mode dependent energy consumption with *sleep* and *active* mode, (ii) energy for switching from *sleep* to *active* mode, (iii) distance dependent transmission of packets, (iv) computation complexity, (v) distance estimations with RSSI measurements, and (vi) position estimation via GPS on beacons. The specific energy parameters are based on the MICA2-mote, which is currently the most popular sensor node platform. At this point, we describe the dominating consumer on sensor nodes and how we modeled them.

Battery Aspect Mica2 nodes use conventional AAA lithium-ionic batteries with an energy amount of ca. 3325 J. Beacons are more powerful, why we assumed an energy level of 21600 J, which is generally used in standard GPS-Modules.

Communication Aspect The basic state of the communication module (CM) is *sleep* that consumes according to Ye et al. in [7] ca. 90 µJ. If the CM switches from *sleep* to *active*, this consumes energy depending on the following formula:

$$E_{start} = E_{init} + E_{on} + E_{switch} + E_{tts} + E_{eval} \quad (7)$$

With the specific parameters of the MICA2 communication module this leads to an energy quantum of 35.64 µJ.

Receiving Mode: When receiving a packet, the communication module switches from *sleep* to *receive*. After receiving all data it can be switched back to the low power state *sleep*. We assume any media access protocol that controls the exact receiving time. The energy for receiving a packet is given by the start energy E_{start} and the receiving energy per bit $E_{receive} = 2.3375$ µJ/bit (see [7]).

Sending Mode: The amount of energy for sending data is also subject to the maximum sending range, which can be adjusted by a specific sending power. The relationship of sending power and maximum sending range is not trivial. In [8] extensive analyzes were presented that define different sending power to specific sending ranges. Out of these results we interpolated a corresponding function $E_{send}(r) = 1,3235 \cdot 10^{-10}\text{J} \cdot r^2 - 9.42 \cdot 10^{-10}\text{J} \cdot r + 6.2181 \cdot 10^{-7}$. This equation additionally considers the sending energy per single bit. Together with the start energy this yields in the sending energy: $E_{start} + E_{send}(r)$. Finally, we have to include the energy for measuring the distance between two nodes by RSSI. For that, a beacon sends a 10 Byte long packet that is sufficient to transmit its network address to sensor nodes. On the other side, when sensor nodes receive this packet they simultaneously measure the receiving power of the signal.

Computation Aspect The old energy model was unable to assign an energy quantum to a specific algorithm part. For example: A function *estimatePosition()* includes the formula for a trilateration. The amount of energy that is needed to execute this function depends on the time interval the microcontroller works in *active* mode to process it from start to end. Because it is almost impossible to model a realistic time in a simulator tool, we decided to count the number of floating point operations (flops). In most data sheets the energy per operation is specified. The ATMega128L consumes at 4 MHz approximately 16.5 mW (with 242 MIPS/W)[1]. According to [9] this leads to 4 nJ per operation. Now, we can multiply flops with the energy per operation and get a specific energy value. However, this assumes that the microcontroller is in *sleep* state (according to [10]: 51.3 µJ), if nothing must be calculated. The

[1] MIPS: Million Instructions per Second

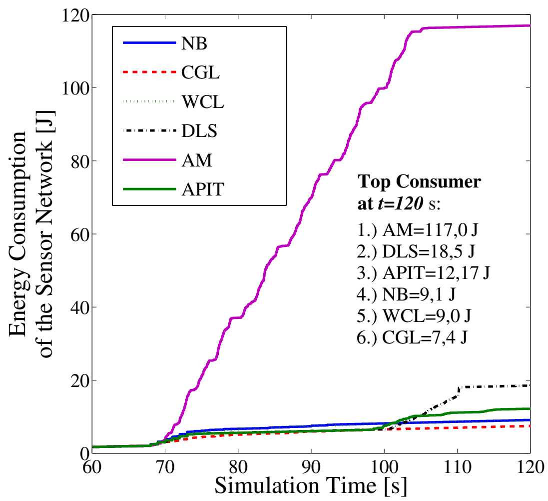

Figure 2. Accumulated energy consumptions of all nodes over simulation time

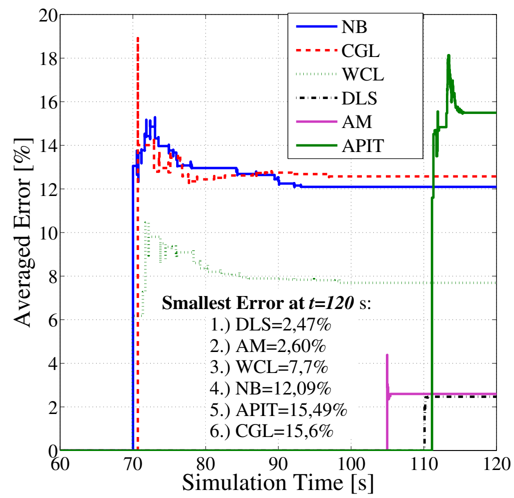

Figure 3. Mean localization error over simulation time

switching time between both states is according to [11] exactly 2 ms, which results in 33 µJ when running the microcontroller at 4 MHz.

GPS Aspect Every beacon estimates its position with GPS autonomously. The u-blox 5 GPS chipset is utilizable on beacons. This chipset consumes ca. 50 mW and needs ca. 1 s to estimate a position[2].

5.2 Configuration

To achieve a fair comparison between the algorithms under test, we run all simulations with the same configuration as following: sensor field size=100 m× 100 m, s=300, b=30, transmission range (sensor nodes)=15 m, transmission range (beacons)=30 m, and simulation time=150 s. Nodes had been distributed uniformly over the sensor field. To simulate noisy distances, all exact distances were falsified. That means, distances were stochastically generated on basis of a Gaussian distribution with the exact distance as mean value and a variance of 10.

6 Results and Discussion

Fig. 2 shows the accumulated energy consumption of all nodes in the network. Both algorithms DLS and AM consume more energy then the approximate algorithms, which is clear, because they are more complex. DLS needs only twice of the energy than NB, CGL and WCL to estimate the nodes positions. In contrast, AM consumes more than a factor of 14. This immense waste of energy is due to the high communication overhead that AM demands, because many packets with only a small payload circulate through the network and are forwarded by every sensor node. DLS demands the receiving of only one packet on every resource limited sensor nodes with a fixed amount of data. NB consumes 1.7 J more energy than CGL, because NB requires distances that are estimated via RSSI measurements. WCL can be classified somewhere between, because it also needs distance measurements, but uses them as a coarse weight in the calculation process. This reduces the error and requires only half of the energy that DLS needs.

In Fig. 3 the localization error is depicted. DLS and AM achieve the same error, because both algorithms bases on a Least Squares Method. However, with $E = 2.6\%$ they show their better precision against the approximate algorithms, which achieve errors of only $\Delta E = 7.7 - 15.6\%$.

NB, CGL, and WCL consider only beacons in one hop range, while the others use all beacons for localization. However, the one hop method can lead to unconnected sensor nodes, which are unable to localize themselves, due to missing beacon positions. Thus, NB, CGL, and WCL have 2.0%, 2.67%, and 1.67% unconnected sensor nodes after simulation, which can only be avoided with additional energy overhead. It must be emphasized that we repeated all test cases several times. However, all simulations showed same tendencies.

We finally calculated the PEP for each algorithm, which was now possible with the simulation results. Please remember, a small PEP indicates a very efficient algorithm. Fig. 4 contains all PEP's. Here, the DLS algorithm shows

[2] This value is valid for a warm start of the module.

the smallest PEP, while all approximate algorithms are following. The highest PEP was produced by AM, which has a high precision, but consumes to much energy for generating the result.

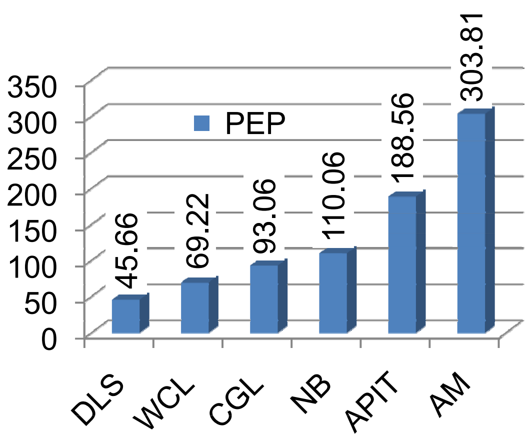

Figure 4. Power Error Product of selected localization algorithms

7 Conclusion

This paper presented a comparison of important localization algorithms in wireless sensor networks. For that, we extended the well known packet simulator J-Sim with a more realistic energy model and implemented selected localization algorithms.

The main question of this paper was how the efficiency of localization algorithms can be verified and clearly presented. For this purpose, we proposed the ”Power Error Product” (PEP), combining the dominating parts in one parameter. These parts are the localization error and the consumed energy. With the PEP we were able to show that our new localization approach, the ”Distributed Least Squares”-algorithm (DLS) is very efficient. DLS achieved the smallest PEP (≈ 46) at reasonable resource overhead on each sensor node. Its direct competitor, the ”Atomic Multilateration” (AM), leads to a PEP of ≈ 304 and selected approximate algorithms ranged between ≈ 69 and ≈ 189.

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, ”A Survey on Sensor Networks”, *IEEE Communications Magazine*, vol.2, pp. 102-114, 2002

[2] T. He, C. Huang, B. M. Blum, J. A. Stankovic, and T. Abdelzaher, ”Range-free localization schemes for large scale sensor networks”, *In Proceedings of the 9th annual international conference on Mobile computing and networking*, San Diego, CA, USA, pp. 81-95, 2003

[3] N. Bulusu, ”GPS-less low cost outdoor localization for very small devices”, *IEEE Personal Communications Magazine*, vol. 7, pp. 28-34, 2000

[4] J. Blumenthal, F. Reichenbach, and D. Timmermann, ”Precise Positioning with a Low Complexity Algorithm in Ad hoc Wireless Sensor Networks”, *PIK - Praxis der Informationsverarbeitung und Kommunikation*, vol. 28, pp. 80-85, 2005

[5] A. Savvides, C. C. Han, and M. B. Srivastava, ”Dynamic Fine Grained Localization in Ad-Hoc Networks of Sensors”, *In Proceedings of the 7th ACM MobiCom*, pp. 166-179, 2001

[6] F. Reichenbach, A. Born, D. Timmermann, and R. Bill, ”A Distributed Linear Least Squares Method for Precise Localization with Low Complexity in Wireless Sensor Networks”, *In Proceedings of the IEEE International Conference on Distributed Computing in Sensor Systems*, pp. 514-528, 2006

[7] W. Ye and J. Heidemann, ”Ultra-Low Duty Cycle MAC with Scheduled Channel Polling”, *Technical Report, ISI-TR-2005-604b*, 2005

[8] A. Falchi, ”Sensor Networks: Performance Measurements with Motes Technology”, *Master thesis*, University of Pisa, 2004

[9] M. Srivastava, ”Wireless Sensor and Actuator Networks: A Driver for Holistic Education in Microsystems”, *In Proceedings of the International Conference on Micromisc Systems Education*, 2005

[10] J. Polastre, R. Szewczyk, and D. Culler, ”Telos: Enabling Ultra-Low Power Wireless Research”, *In Proceedings of the 4th International Symposium on Information Processing in Sensor Networks*, pp. 364-369, 2005

[11] L. Necchi and L. Lavagno and D. Pandini and L. Vanzago, ”An Ultra-low Energy Asynchronous Processor for Wireless Sensor Networks”, *In Proceedings of the 12th IEEE International Symposium on Asynchronous Circuits and Systems,* pp. 78, 2005