

Efficient Incremental Sensor Network Deployment Algorithm

Luiz Filipe M. Vieira¹, Marcos Augusto M. Vieira¹, Linnyer Beatrys Ruiz^{1,3},
Antonio A.F. Loureiro¹, Diógenes Cecílio Silva², Antônio Otávio Fernandes¹

¹Computer Science Department, Federal University of Minas Gerais
Avenida Antônio Carlos, 6627 - Pampulha - Belo Horizonte-MG Brazil

²Department of Electrical Engineer, Federal University of Minas Gerais
Avenida Antônio Carlos, 6627 - Pampulha - Belo Horizonte-MG Brazil

³Pontifícia Universidade Católica do Paraná
R. Imaculada Conceição, 1155 - CEP 80215-901 Curitiba-PR Brazil

{lfvieira, mmvieira, linnyer, loureiro, otavio}@dcc.ufmg.br

diogenes@cpdee.ufmg.br

Abstract. *We propose an efficient algorithm for incremental deployment of nodes in a wireless sensor network. By examining the distribution of node density, its energy level, and the sensing cover area, the algorithm indicates which position should have more nodes deployed and how many new nodes are necessary to cover the desired monitoring area. Our approach uses the largest empty circle problem, a well-known computational geometry problem which is $\Theta(n \log n)$, to incrementally deploy sensors in a stressed wireless sensor network. Experiments show that our algorithm is very close to the upper limit and performance 2.5 times better than the random insertion algorithm.*

Resumo. *Neste artigo é proposto um algoritmo eficiente para dispor nós incrementais em uma rede de sensores sem fio. Examinando a distribuição de densidade dos nós, o nível de energia deles, e a área sendo sensoriada, o algoritmo indica quais posições devem ter mais nós dispostos e quantos nós serão necessários para cobrir a área que se deseja monitorar. A abordagem apresentada utiliza o problema do maior círculo vazio, um problema de geometria computacional bem conhecido que possui complexidade $\Theta(n \log n)$, para incrementalmente dispor nós sensores em uma rede de sensores sem fio em uso. Experimentos mostram que o algoritmo é próximo do limite superior e que ele tem uma performance 2,5 vezes melhor que um algoritmo incremental aleatório.*

1. Introduction

We propose an efficient algorithm for incremental deployment of nodes in a wireless sensor network (WSN). By examining the distribution of node density, its energy level, and the sensing cover area, the algorithm indicates which position should have more nodes deployed and how many new nodes are necessary to cover the desired monitoring area. Most work has been done in determining the sensor deployment of the initial network [5, 7]. To our knowledge, this is the first to consider deterministic incremental deployment after the initial network has been used and considering the energy consumed. Another advantage of the proposed algorithm is its computation time. Combining well-known results in computational geometry, the worst case to compute the position to add a node is $\Theta(n \log n)$, where n is the total number of nodes in the network.

Recently, the idea of WSNs has attracted the researchers attention due to wide-ranged potential applications that will be enabled by WSNs, such as biological detection, home security, etc. [8, 14, 4, 13]. In addition to the new applications, wireless networks provide a viable alternative to several existing technologies. A key challenge in WSNs is to determine a sensor field architecture that minimizes cost, provides high sensor coverage, resilience to sensor failure, and appropriate computation and communication trade-off. Intelligent sensor deployment facilitates the unified design and operation of sensor systems, and decreases the need for excessive network communication.

The deployment of sensor networks varies with the application considered. In some environments, it can be predetermined and be strategically hand placed. The deployment can also be a priori undetermined: the sensors may be air-dropped or deployed by other means. The sensors can also be deterministically deployed by a robot, which can place them on the exact localization predetermined to optimize the use of network resources. A number of problems with WSN can be solved or diminished by including a mobile robot as part of the system [15]. Robot designed for WSN have already been built. Robomote [21] is a tiny mobile robot platform for ad-hoc Sensor Networks. Many others small robots teams are available today [11]. In this study, the sensors are assumed to be deterministically placed.

Determining the optimal placement - even in a greedy sense - is a fundamentally difficult problem [12]; the deployment algorithm described in this paper therefore relies on a number of heuristics to guide the selection of deployment locations.

To evaluate our design, we perform simulation comparison. We compared the incremental deployment algorithm with a random insertion algorithm. To strengthen our result, we considered four different energy models. As predicted, our design can save energy without losing sensing area.

This scheme could be used in management architecture for WSN [19] or projects such as SensorWeb [1].

In this paper we preset a new framework to determine the sensor deployment in a used WSN, minimizing energy consumption, maximizing the lifetime of the network and minimizing the cost involved. It is organized as follows, in the next section we give an overview of the WSN. In Section 3 we summarize the related work. Then, in Section 4, we present the algorithm based on the largest empty circle, a well-known problem in computational geometry. Section 5 contains a wide array of experimental results. Section 6 is a brief discussion of future works and the conclusion.

2. WSN Architecture

This section gives an overview of the WSN architecture used in this work. WSNs are networks composed of a great number of sensor nodes. The objective of these networks is to collect information.

WSNs usually do not have a previous infrastructure, like cellular or local wireless networks. WSN is an ad hoc network, since its topology is dynamic, due to the fact that sensor node can wake-up joining the network, or go to sleep, leaving the WSN.

Data sinks typically are sensor nodes that usually differ from others sensor nodes. Data sink purpose is to collect information from the network and send to an external observer. It connects the WSN to the outside world. This node has more energy, longer radio range and does not need sensors.

The major resource at WSN is energy. Each sensor node is composed of a small

battery, with a limit capacity. It is almost infeasible to recharge all battery since WSN can be composed of thousands of sensor nodes. Therefore, the WSN project focus, from hardware design to network protocols, is saving energy.

3. Related Work

Minimizing energy consumption, maximizing the lifetime of the network, minimizing the cost involved has been a major design goal for WSNs.

T. Clouquer et al. [6] realized a study in minimizing the cost of deployment sensors to achieve the desired detection performance for target detection. The minimum path exposure is used as a measure of the goodness of deployment. They did not work with deterministically placed sensors or considered the energy consumed by the network.

D. Tian et al. [22] propose an approach to prolong the system lifetime that preserves sensing covering. A node decides to turn it off when it discovers that its neighbors can help it to monitor its whole area. This works presents a scheduling scheme; nodes take turns in saving their energy without affect the service provided. This can also be used with our algorithm.

K. Chakrabarty et al. [5] present an integer linear programming (ILP) solution for minimizing the cost of sensor for complete coverage of the sensor field. Energy consumed by the network is not considered. The exact solution to the ILP takes an excessive amount of computation time. Our new approach reduces the computation time, taking advantage of known results in computational geometry [2].

In [7], it is proposed a sensor placement for grid coverage to construct the initial network. Sensor behavior is modeled probabilistically, due to the uncertainty associated with sensor detection. Our approach is different, we consider the energy of each node, and our algorithm can be applied not just to grid placement.

Computational Geometry results have been applied in the study of WSNs. Meguerdichian et al. [16] use the Delauney triangulation and the Voronoi diagram to identify the path of maximal breach of surveillance.

To our knowledge this is the first time that computational geometry is combined with algorithm to incrementally deploy sensors in a used WSN and to reduce computation time.

4. Incremental Deployment

Many nodes with communication, sensing and processing abilities compose a WSN. These nodes have restricted energy and the lifetime of the system is affect by the energy on the nodes. To increase the lifetime of the network, new nodes can be deployed. Since the local observations made by the sensors depend on their position, the performance of the network is a function of the deployment. The area where events, which the sensors are sensing, can occur will be denominated "desired monitor area".

Sensor node may not be covering the desired monitor area. Some reasons are the network density is low, nodes were not correctly deployed at the monitoring area, or sensor nodes lost their energy. Here we propose an algorithm to help incremental deployment of sensors. We define a scheme that examining the distribution of node density, its energy level, and sensing cover area, the algorithm indicates the quantity and which positions should have more node. The design utilizes ideas from the largest empty circle problem, which is explained next.

4.1. Largest Empty Circle

Following the definition in [18], the largest empty circle problem is: “Find the largest empty circle whose center is in the (closed) convex hull of a set of n sites S , empty in that it contains no sites in its interior, and largest in that there is no other such circle with strictly large radius”.

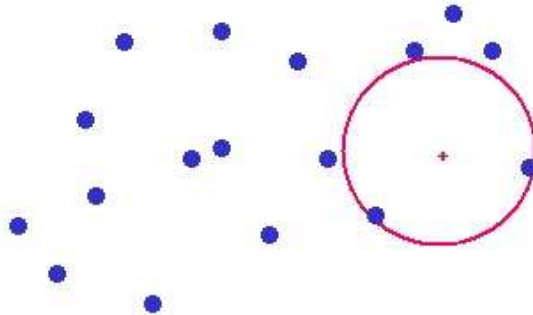


Figure 1: Largest Empty Circle Example.

Figure 1 shows an example of Largest Empty Circle. Given a set of points, called sites, the answer is at the cross symbol, which represents the center of the wanted circled. Cross symbol represents the location whose distance to the nearest node is as large as possible.

4.2. Assumptions

We work with homogenous [19] WSN distributed on a 2-D plane field. Our scheme does not loss generality because it can easily be extend to n -dimension. And it also can be extend to heterogeneous WSN. One way to do this is by adding fictitious sites. Each node is immobile, although the network topology could be dynamic, since nodes can become unavailable permanently or temporarily.

We assume that we have information about each node position. The position does not need to be global; it could be relative to the base station or to a known point. As point out in [23], obtaining reliable node location has been studied in different contexts. Using Global Positioning System (GPS), we can determine geographical location with fair accuracy [9]. Others localization systems exist. One example is to use sensor nodes called beacons, which know their coordinates in advance. Other nodes can approximate distances to beacons and decide their locations by trilateration. For more information see [3, 10].

Another assumption is that we have the approximate energy level in each node. The energy map of the network can been obtained as shown in [23, 17]. Furthermore, there are components such as DS2438 that allows monitoring battery energy [20].

Applications of the algorithm assume that the sensors can be strategically placed, by any form. For example, the sensors could be deterministically deployed by a robot, which can place them on the exact localization predetermined to optimize the use of network resources. SensorWeb [1] is a NASA project that could be using such an application.

Sensor nodes are deployed through a desired monitor area, form a randomly placed network, wake-up, self-test, establish dynamic communication among them, composing an ad hoc network. The algorithm is used in those situations, which the network is deployed, and not optimally configured, such as positioning each node to form a grid. In those cases, the new sensors can just replace old ones.

An advantage of the algorithm approach is the fact that it can be computed outside the WSN or at the data sink, saving the network energy and not requiring the algorithm to be distributed.

4.3. The problem

Given the localization and the approximate energy level of each node in a WSN, what is the minimum number of new nodes that should be added to the network so that it does not lose any covering area? Also, where should the new nodes be positioned?

The solution should be computed in reasonable time, considering that it may be applied to networks with more than 50 nodes.

4.4. The algorithm

The input of the algorithm consists of the position and the energy map of the network.

First, we classify the operational state of each node using the parameters defined by the MANNA architecture [19], as shown in Table 1 classification.

Operational State	Energy Level
Active	$\geq 30\%$
Critical	$< 30\%$ and $> 15\%$
Inactive	$\leq 15\%$

Table 1: Operational State of sensor node.

Only nodes with active state will become sites. Nodes on other operational state will be ignored. Others policies and others table values could be used instead. Using the set of sites, the algorithm calculates the largest empty circle. Regions with larger area are the ones that need new nodes because the nodes are not healthy (active) or there are few nodes at the region. The center of the circle indicates where is the best position to add a node. Repeating the process while the largest empty circle is larger than a threshold will indicate how many new nodes are necessary to guarantee that every circle area larger than the threshold is being covered and also where these new nodes should be placed.

To calculate the largest empty circle is necessary to have a Convex Hull, otherwise any point outside it could be the locale of the center of the circle. We choose four fictional nodes to define the Convex Hull of the rectangle surveillance area. The number of fictional nodes depends on the shape of the desired monitoring area. It does not affect the algorithm since it keeps looking for the largest empty circle until the area of the circle is larger than a threshold. If the region near the border were uncovered, the algorithm would, anyway, indicate that region.

Figure 2 shows the structure of the network from the design vision. At the border, fictional nodes were created to delimit the Convex Hull. Nodes not active were discarded. The algorithm calculates the largest empty circle considering only active nodes. The center of the circle indicates the best position to add a node. The worst-case complexity of Largest Empty Circle is $\Theta(n \log n)$. The algorithm can be extended to work with restricted areas (areas where we don't want to place sensors). Let's consider a real application of WSN, an application of monitoring and sensing a volcano. When the sensors are deployed, it is desirable that the sensors are not placed inside it or where the lava can reach them, otherwise, the sensors may be destroyed. Fictitious sites can be considered on the restricted area, so that the algorithm does not consider adding a sensor on the restricted area.

Figure 3 shows the flow graph of the incremental algorithm.

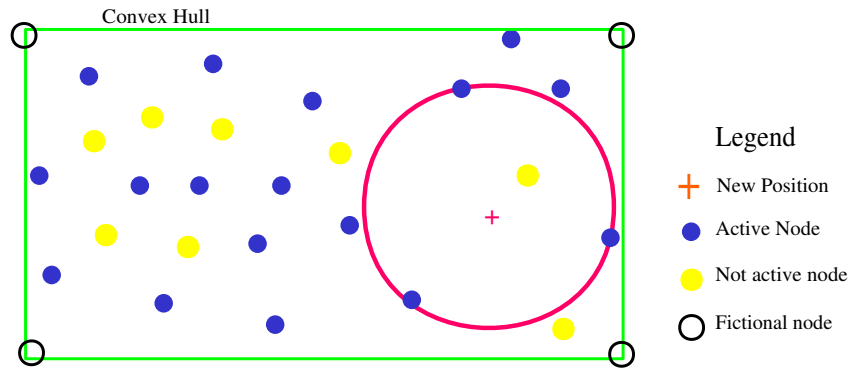


Figure 2: Algorithm example.

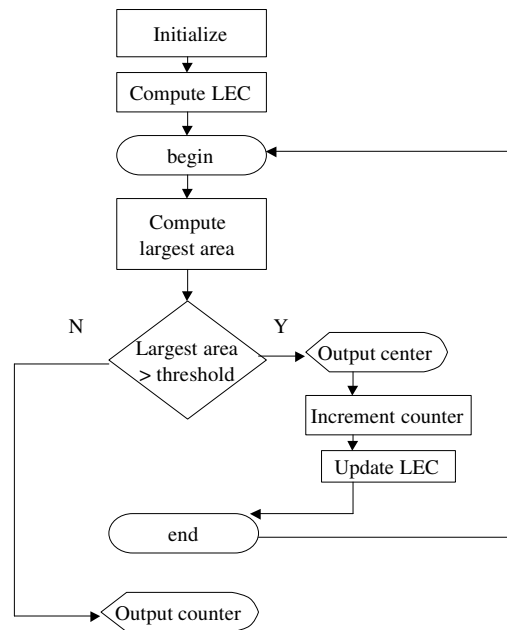


Figure 3: Flow Graph

We will refer, in this paper, to this algorithm, shown in pseudocode in Algorithm 1, as the incremental algorithm.

It should be clear that the aim of the algorithm is to determine the positions to add new nodes, and not to trace the route that passes through all these points and that minimizes energy consumption. This could be done using the information returned by the incremental algorithm.

4.5. Guarantee connectivity in the graph

The network can be seen as a graph, where the sensor nodes are nodes in this graph. An edge connecting two nodes means that the two sensors are able to communicate. It is desired that the graph be connected, meaning that all sensor nodes are able to communicate with the data sink. The connectivity in the graph can be guaranteed by setting the value of threshold to $radius / (2 * \cos 30^\circ)$, where radius is the radio range. This can be calculated from the case where there are three nodes that can not communicate among themselves, which is shown in Figure 4.

Considering the sensing radius instead of the radio radius, it is possible to guarantee that the whole area among the sensor nodes is being sensed.

Algorithm 1 Incremental Algorithm

Input: position and the energy level at each sensor node in the network.

Output: the quantity and the position of the sensor nodes.

for each active node **begin**

 set it as a site;

end for

count_new_nodes = 0;

Calculate Largest Empty Circle ();

do begin

 calculate largest_empty_circle_area();

if(largest_empty_circle_area > threshold) **then begin**

 output the center of the Largest Empty Circle and set it as a site;

 count_new_nodes = count_new_nodes + 1;

 update Largest Empty Circle ();

 keep_searching = true;

end if

else begin

 keep_searching = false;

end else

while (keep_searching)

output count_new_nodes;

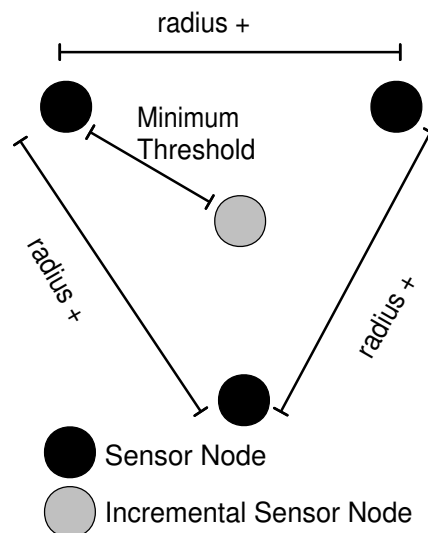


Figure 4: Minimum threshold to guarantee connectivity in the network.

5. Experiments

Ideally, we would like to compare the coverage produced by this algorithm with that produced by an optimal solution. Unfortunately, finding the optimal solution for any non-trivial example is extremely difficult [12], even when we have good a priori maps of the environment. Consequently, in this paper, we make no attempt to find such solutions.

In order to evaluate our design for adding new nodes to the network, we compare its performance to adding nodes randomly, considering different types of energy consumption. Although placing sensor nodes randomly is a naive approach, it serves as a base for comparison. In this section, we present our results and discuss their implications.

5.1. Metrics

The key performance criterion in adding new nodes to WSN is to maintain the network active and sensing the desired area. Every point of the coverage area should be inside an active sensor. Our metric takes as relevant the area being coverage and the number of nodes added.

5.2. Energy Dissipation Model

We used the models proposed by [23] to stress the network. The UNIFORM dissipation model, capture uniformly distributed sensing activity. More precisely, during a sensing event, each node n in the network has a probability p of initiating a local sensing activity, and every node within a circle of r centered at n consumes fixed amount of energy e . This latter feature of the model is inspired by collaborative sensing algorithm.

In the uniform dissipation model, the energy at all nodes decreases at approximately the same rate. However, in a realistic environment, different regions in a sensor field may have different energy dissipation rates. To model this, there is the HOTSPOT model. In this model, there are h hotspots uniformly distributed in random on the sensor field. Each node n has a probability of $p = f(x)$ to initiate a local sensing activity, and every node within a circle of radius r centered at n consumes fixed amount of energy e ; where f is a density function and $x = \forall i \{ \min |h - h_i| \}$ is the distance from n to nearest hotspot. We used three density functions: $f(x) = ae^{-ax}$ is the density function for EXPONENTIAL distribution, where the hotspot effect drops quickly with increasing distance; the PARETO density $f(x) = a/(x+1)^{a+1}$, where the impact of the hotspot falls off more gradually than an exponential distribution with the same value a , and the NORMAL distribution.

5.3. Settings

Our experiments were conducted on a square desired sensing area, with area equal to 80,000 dm². The radius of the sensing area of each sensor is 20 dm. To simplify calculation, we define the area being covered as the inscribed square instead of a circle. So each node would have an area of 800 dm². With this number, a perfect placement of 100 nodes would cover 100% of the area. First, we compare the incremental algorithm against a random adding node algorithm using a random network with no energy model. Second, we repeated the comparison using all the energy dissipation models described in subsection "Energy Dissipation Model".

The first experiment generated an initial network by positioning 50 nodes at random. Then we used the incremental algorithm to add nodes until we reached 100% of covering area. Using the same initial network, we add nodes randomly until we reached 100% of covering area.

The second experiment, we place 100 nodes on the desired area covering 100% of the area, and simulate using each of the energy models to find out the nodes that would be considered inactive. Then we add nodes until we cover 100% of the area, using the incremental algorithm and the random insertion algorithm.

5.4. Results

Figure 5 shows the results of the first experiment. We plot the ratio of the area being covered by the initial network, which was the same for both algorithms, and the ratio of the area being covered by the incremental and the random insertion algorithm.

As shown Figure 5, adding nodes with the incremental algorithm covered more area than adding randomly. With 109 nodes, we covered 100% of the sensing area. This is

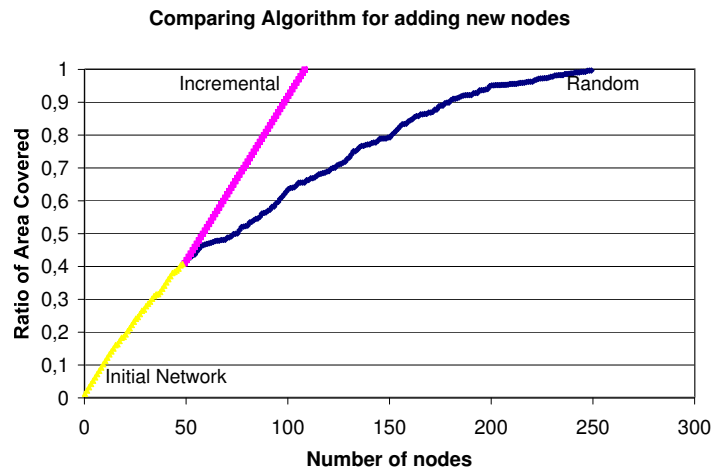


Figure 5: Comparing algorithm for adding new nodes until we reach 100% of covering area.

very close the upper limit, which in our experiment was 100 (total sensing area / sensing area of a node). The random insertion algorithm needed 249 nodes to covered 100%. In other words, in terms of number of nodes necessary to cover the desired monitoring area, the incremental algorithm performance 2.5 times better than the random insertion algorithm.

The second experiment considers networks with different energy models.

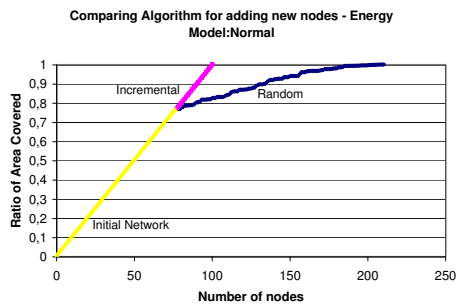
Figure 6 shows that the incremental algorithm performs close to the upper limit and independent of the energy model. It tries to cover the area that is not being covered and it also takes into account the energy information, covering the areas with low energy and consequently, being independent of the energy model.

6. Conclusion

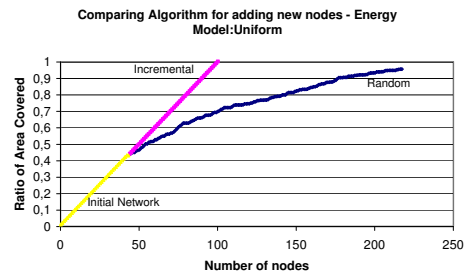
Resource of wireless sensor network is critical. A good sensor deployment technique, in applications that it is possible to deterministically place sensors, is necessary to make the most of the network resource. Our design defines a scheme that exams the distribution of node density, its energy level, and sensing cover area, and indicates the quantity and positions where new node should be deployed.

This scheme is an efficient algorithm for incremental deployment of nodes in a wireless sensor network that combines results in computational geometry. It increases the lifetime of the network (as new nodes are added as needed) and minimizes the cost involved (as the minimum number of sensor that maintains the covering area is found). Furthermore, it only takes

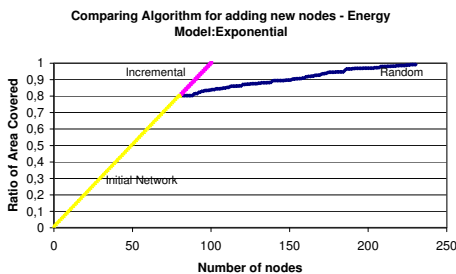
Experiments show that our algorithm is very close to the upper limit and performance 2.5 times better than the random insertion algorithm. It also works independently of the energy model.



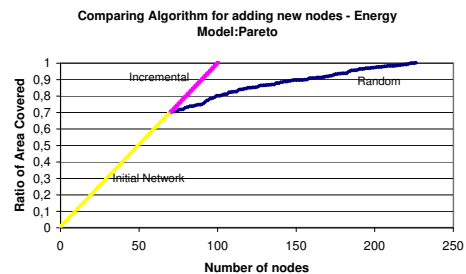
(a) Adding new nodes to a network that lost energy as described by the normal model.



(b) Adding new nodes to a network that lost energy as described by the uniform model.



(c) Adding new nodes to a network that lost energy as described by the exponential model.



(d) Adding new nodes to a network that lost energy as described by the Pareto model.

Figure 6: Comparison of the algorithm considering different energy models.

Acknowledgments

This work was partially supported by CNPq, an entity of Brazilian Government aimed at scientific and technology development. Process 55.2111/02-3, 13.3556/2002-6, 133555/2002-0 and 830107/2002-9.

References

- [1] National Aeronautics and Space Administration (NASA). Sensor webs, jet propulsion lab. <http://sensorwebs.jpl.nasa.gov>.
- [2] Alok Aggarwal and Subbach Suri. Fast algorithm for computing the largest empty rectangle. In *Proc. of the third annual symposium on Computational geometry*, pages 278–290, 1987.
- [3] N. Bulusu, J. Heidemann, and Estrin. Gps-less low cost outdoor localization for very small devices, 2000.
- [4] Alberto Cerpa, Jeremy Elson, Michael Hamilton, Jerry Zhao, Deborah Estrin, and Lewis Girod. Habitat monitoring: application driver for wireless communications technology. In *Workshop on Data communication in Latin America and the Caribbean*, pages 20–41, Costa Rica, April 2001. ACM Press.

- [5] K. Chakrabarty, S. S. Iyengar, H. Qi, and E. Cho. Grid coverage for surveillance and target location in distributed sensor networks. *IEEE Transactions on Computers*, 51:1448–1453, December 2002.
- [6] Thomas Clouquer, Veradej Phipatanasuphorn, Parmesh Ramanathan, and Kewal Saluja. Sensor deployment strategy for target detection. In *Wireless Sensor Networks and Applications (WSNA)*, Atlanta, GA, 2002.
- [7] S. S. Dhillon, K. Chakrabarty, and S. S. Iyengar. Sensor placement for grid coverage under imprecise detections. In *Proc. International Conference on Information Fusion*, pages 1581–1587, 2002.
- [8] Deborah Estrin, Ramesh Govindan, John Heidemann, and Satish Kumar. Next century challenges: scalable coordination in sensor networks. In *Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, pages 263–270. ACM Press, 1999.
- [9] B. R. Badrinath et al. Special issue on smart spaces and environments. *IEEE Pers. Commun.*, October 2000.
- [10] L. Girod. Development and characterization of an acoustic rangefinder, 2000.
- [11] Bob Grabowski. Small robot survey. http://www.contrib.andrew.cmu.edu/~rjg/webrobots/small_robot_survey.htm%1.
- [12] Andrew Howard, Maja J Mataric, and Gaurav S. Sukhatme. An incremental deployment algorithm for mobile robot teams. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2002.
- [13] I.F.Akyildiz, W.Su, Y.Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: A survey. *IEEE Communications Magazine*, 40(8):102–14, August 2002.
- [14] J. M. Kahn, R. H. Katz, and K. S. J. Pister. Next century challenges: mobile networking for smart dust. In *Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, pages 271–278. ACM Press, 1999.
- [15] Anthony LaMarca, Waylon Brunette, David Koizumi, Matthew Lease, Stefan B. Sigurdsson, Kevin Sikorski, Dieter Fox, and Gaetano Boriello. Making sensor networks practical with robots. *International Conference on Pervasive Computing*, 2002.
- [16] Seapahn Meguerdichian, Farinaz Koushanfar, Miodrag Potkonjak, and Mani B. Srivastava. Coverage problems in wireless ad-hoc sensor networks. *IEEE Infocom*, pages 1380–1387, 2001.
- [17] Raquel A. F. Mini, Badri Nath, and Antonio A. F. Loureiro. Prediction-based approaches to construct the energy map for wireless sensor networks. XXI SBRC, May 2003.
- [18] Joseph O’Rourke. *Computational Geometry in C*. Cambridge University Press, 1993.
- [19] Linnyer Beatrys Ruiz, José Marcos Nogueira, and Antonio A. F. Loureiro. Manna: A management architecture for wireless sensor networks. In *IEEE Communication Magazine*, volume 41, February 2003.
- [20] Dallas Semiconductor. Ds2438 datasheet. <http://pdfserv.maxim-ic.com/arpdf/DS2438.pdf>.
- [21] Gabriel T. Sibley, Mohammad H. Rahimi, and Gaurav S. Sukhatme. Robomote: A tiny mobile robot platform for large-scale sensor networks. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA2002)*, volume 2, pages 1143–1148, Washington, DC, May 2002.

- [22] Di Tian and Nicolas D. Georganas. A coverage-preserving node scheduling scheme for large wireless sensor networks. In *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*, pages 32–41. ACM Press, 2002.
- [23] Jerry Zhao, Ramesh Govindan, and Deborah Estrin. Residual energy scans for monitoring wireless sensor networks. In *IEEE Wireless Communications and Networking Conference (WCNC 02)*, Orlando, FL, March 2002.