A LOCALIZATION ALGORITHM EXTENSION FOR THE EVOLVABLE SENSOR NETWORK

DuyBach Bui, Daeyoung Kim
Real-time and Embedded Systems Laboratory
Information and Communications University
103-6, Munji-dong, Yuseong-gu, Daejon, 305-714
South Korea
{bduybach, kimd}@icu.ac.kr

ABSTRACT
A localization algorithm is an important component in a wireless sensor network. The requirements for such an algorithm include high estimated node position accuracy and low communication overhead. In an evolvable sensor network, where some running sensor nodes exhaust after a working period and some new nodes are added to maintain the proper operation of the network, a localization algorithm used to find position of new added nodes should also satisfy the requirements about accuracy and overhead. Moreover, the algorithm should be able to exploit information of nodes which are in the old network. This paper proposes an extension of a localization algorithm so that it can work in an evolvable sensor network. The simulation results show that the extension obviously meets the requirements. Especially, under certain conditions, the estimated positions of the new added nodes are more accurate than that of the old nodes since the extension can exploit the information of the old network.

KEY WORDS
Sensor Network, Localization, Location Finding.

1. Introduction

Recently, a network of a mass of low-power wireless smart sensors (WSN) has been emerging as an appealing research topic. This system consists of many small form factor battery-powered nodes, each equipped with sensors, one or more microcontrollers and a radio transceiver. After deployed in an area, the sensor nodes self-organize to build a wireless network. Environmental information of the area is collected by the sensor nodes, then sent to base stations. WSN holds much promise in many civilian applications, such as smart environment, natural habitat monitoring, disaster relief, etc.

Feasibility of many WSN applications and various location-aided network protocols depends on the availability of sensor node positions. A sensed data with position information is more meaningful for various applications, such as smart environment [2], natural habitat monitoring [3], and disaster relief. In addition, in many WSN protocols, sensor node positions are considered necessary information. Examples are energy-efficient location-aided routing protocols [1, 4, 5], energy-efficient topology control protocols [11, 12] and target tracking protocols.

This paper analyzes and proposes an extension of our Parametric Learning-based Distance Localization (Para-LDL) algorithm [14], with which the algorithm can work in an evolvable sensor network. The proposed algorithm is a part of our sensor network project, ANTS (An evolvable Network of Tiny Sensors). ANTS is a term used to refer to a sensor network which is capable of developing, self-organizing over many generations of software architectures, hardware, network structures and network topologies after being deployed.

The following is a simple example of ANTS. Considering a sensor network which includes *N* nodes and has been operated for a period of time, at time *t* a number of nodes *n*, in that running network, are energy-exhausted. *n* is smaller than *N*. We call the nodes which are still working at the time *t live nodes*. To guarantee the proper operation of the sensor network, some *new nodes* should be added. To start working, the added nodes must be able to self-organize into the network. We assume that the boot-up procedure includes a self-localization process. Our aim in this paper is to improve Para-LDL so that it is able to support the *new nodes* estimating their positions.

The paper is structured as follows. Section 2 gives a brief review of existing localization algorithms, including Para-LDL. In Section 3, the requirements for a localization algorithm working on an evolvable sensor network and the improvement of Para-LDL to meet those requirements are discussed. The simulation setup and evaluation is described in Section 4, with lastly, the conclusion in Section 5.

2. Existing Localization Algorithms

Figure 1 shows the model of wireless sensor networks of concerns.

This network includes a small number of *anchor* nodes, knowing their own positions a priori by either using GPS or being manually configured. The other majority of nodes are unknown position nodes, called *unknowns*. Both *anchors* and *unknowns* are equipped with a low power measuring device, which is used for measuring distances between the nodes and its neighbors. A measured distance is subject to error because of both intrinsic technological constraints and extrinsic environmental conditions. Some *unknowns*, neighbors of *anchors*, can measure distances to *anchors* directly, but most of them are multi-hops away from *anchors* and must estimate distances to *anchors* relying on others' distances. We assume that every node has the same radio range and can measure distance within the radio range. This is a practical assumption if a sensor node uses RSSI for ranging.

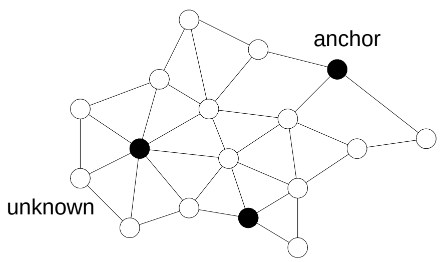

Figure 1. Localization Model of Wireless Sensor Networks

A two-dimensional node position can be calculated by triangulation if one knows the distances between the node, at least three *anchors*, and the positions of *anchors*. While positions of *anchors* can be easily obtained through communication, measuring and/or estimating distances to *anchors* poses a much greater challenge. The challenge comes from the constraints of existing distance measurement technology. In which technologies often produce unreliable results. Many algorithms try to solve this problem. We discuss these algorithms divided based on the usage of distance measurement.

2.1 Range-free Localization

Avoiding using measured distance in estimating positions, range-free approaches are robust in terms of ranging error. In DV-Hop [6], an *unknown* estimates distance to an *anchor*, by multiplying the number of hops between them with the average distance of each hop, which is computed by *anchors*. Another algorithm, introduced in [13], also uses the same mechanism as DV-Hop, i.e. multiplying hop count with the average hop distance to estimate distance. This algorithm, however, uses an offline average hop distance, which is estimated using the formula, introduced in [9]. Given the node density of a wireless ad-hoc network, one can estimate the expected hop distance a priori by using this formula. Other proposed schemes such as APIT [10] also work as a range-free scheme with some different assumptions from the above algorithms. First, although it can work with static *anchors*, this algorithm is designated for systems with mobile *anchors*. In addition this algorithm considers an *anchor* with radio range greater than other nodes. APIT shares the same inherent disadvantage as previous range-free approaches in that, without using measured distance, they are incapable of providing fine-grain node positions.

2.2 Range-based Localization

In range-based approaches, measured distances are used for estimating distances to *anchors*. Authors in [7] simply estimate distance from a node to an *anchor* by adding up measured distances of each hop on the way from the node to the *anchor*. Although this algorithm reports a low position error in an environment of low ranging error, its performance dramatically reduces once the distance measurement error exceeds 10%. The other range-based scheme, Euclidean [6] also works well only in an environment with small distance measurement error and high node density.

2.3 Multi modal Localization

Different from above approaches, Para-LDL in [14] uses multi modal information when estimating distances. For example, it may use both range-base and range-free information. To do so authors in [14] proposed a distance model in which a node-to-node distance is represented as a function of different distance-related variables. The proposed distance-related variables between node i and *k* are:

1. *Smallest sum distances (ssd)*: the shortest sum of measured distances of each hop, on the way from node *i* to node *k*. This is range-based information.
2. *Smallest number of hops (snh)*: the smallest number of hops from node *i* to node *k*. This is range-free information.

The distance function which is used for representing a node-to-node distance is:

$$d_w\left(ssd, snh\right) = w_0 \times ssd + w_1 \times snh + w_2$$

Coefficient vector (w_0 w_1 w_2) which is associated with each node is computed by each *anchor* using machine learning technique. *Unknowns* use the average of the closest anchors' coefficients as their own coefficients. As long as the coefficients are found and the distance function is defined, a node is able to estimate its distance to anchors. For example, if w_0=1, w_1=2, w_2=3 and a node with *ssd* and *snh* to an anchor are 10 and 3, respectively, then its estimated distance to the anchor is 1*10+2*3+3=19.

Distinguished from other approaches, which estimate distances from a node to *anchors* explicitly using one kind of available information only, Para-LDL calculates distances using a machine learning method, exploiting different kind of available information, giving more robust and better results. Para-LDL is also proved to be competitive in communication overhead and computation. For these reasons, Para-LDL was chosen to extend in our proposal.

3. Para-LDL Extension

In this section we discuss an extension of the Para-LDL algorithm, with which Para-LDL can work in An evolvable Network of Tiny Sensors (ANTS).

Firstly, the requirements of this extension in priority of importance are as follows:

1. The estimated position of a *new node* must have the same degree of accuracy as the old nodes with reasonable communication overhead.
2. A *new node* should find its position without the need of running the localization algorithm in every *live node* again.
3. *New nodes* should exploit information about the old network which is maintained by *live nodes* to better estimate their position.

To meet these requirements, we design our algorithm with following principles:

1. Upon request from the new nodes, the *live nodes* transmit their kept information, i.e. *anchors*' coordinates, distance-related variables and coefficients of the distance function, to the *new nodes*.
2. Upon request from the new nodes, the *anchors* can estimate and broadcast coefficients of the distance function again.

The algorithm following these principles can avoid flooding since localization-related data is transferred when it is required by the *new nodes*. In addition, the information of old network which is maintained by the *old nodes* can be used by the *new nodes*.

LDL Evolution Improvement

Para-LDL Evolution Improvement (Para-LDL-EI) is an extension of Para-LDL. The main procedure of the extension for estimating the position of a *new node* is as follows:

1. After being deployed, the *new node* starts the localization process by broadcasting a request packet (RQP). After broadcasting an RQP, the *new node* waits for reply packets. Upon receiving this RQP, each neighbor of this node, a *live node*, answers by sending a reply packet (REP), which includes information associated with its collected *anchors*, i.e. the *anchors*' coordinates, distance-related variables to the *anchors* and distance function coefficients of those *anchors*.
2. Upon receiving the REPs, the *new node* records *anchors*' associated information. The *new node* then selects the most appropriate value of each distance-related variable among the reply packets for each anchor. For example, if *ssd* is the used distance-related variable, then if the reply packet of neighbor *i* contains the smallest *ssd* value of *anchor k* compared to those of other neighbors, that value is the most appropriate value of *ssd* associated with the distance to *anchor k*. The *new node* then adjusts the selected distance-related variables, for example increasing *snh* by one, and finally broadcasts an REP for its collected anchors one more time. This new reply packet has the same structure as the previous ones. This second reply broadcasting aims to provide information to other *new nodes*, which do not have *live nodes* as neighbors. Note that only *new nodes* perform this procedure. After finishing this procedure, the *new node* has enough information to estimate its position. The *new node* can stop the localization process here or continue with the following steps. We refer to these two alternatives as Para-LDL-EI(1) and Para-LDL-EI(2), respectively. The main difference between the alternatives is that the latter may stimulate *anchors* to compute and broadcast the distance function coefficients again while the former does not. The effects of this difference will be shown in the next section.
3. The *new node* then broadcasts a fake beacon packet for each *anchor* it collected. This packet has the same structure as the original beacon packet, which is used during the first localization [14]. After broadcasting, the *new node* waits to receive the re-computed coefficients from anchors for a certain amount of time.
4. Each node treats the fake beacon packet in the same way as an original beacon packet. If an *anchor* receives the fake beacon packet, and the distance-related variables in that packet are more appropriate than the collected ones, the *anchor* learns the distance function and broadcasts a re-computed coefficients packet, again.
5. After waiting for a certain amount of time, the *new node* computes its own coefficients function using the new coefficients of anchors which are re-computed and the old coefficients collected from the REPs about the anchors which are not re-computed. Finally, the *new node's* position is estimated.

4. Simulation

In this subsection, we demonstrate and analyse the simulation results of Para-LDL-EI(1) and (2).

We use NS2 with 802.11 protocol at the MAC layer to simulate the new algorithm. In our simulation, each node has the same radio range and can measure distance in the same range as radio range, which is practical if sensor nodes use radio signal strength for measuring distance.

The distance measurement error has a Gaussian distribution with zero mean and varied standard deviation. The standard deviation is interpreted in terms of the radio range. For example, the 0.1 standard deviation means that the standard deviation is 10% of the radio range. The simulation scenario includes 100 nodes, which are randomly deployed, with anchor fraction of 10% and connectivity degree of 14. Measured distances are subject to zero mean Gaussian distributed error with a standard deviation of 10% of radio range. We assume that *anchors* have much more energy than ordinary nodes, not being exhausted over several generations. The number of *new nodes* equals the number of *exhausted nodes*.

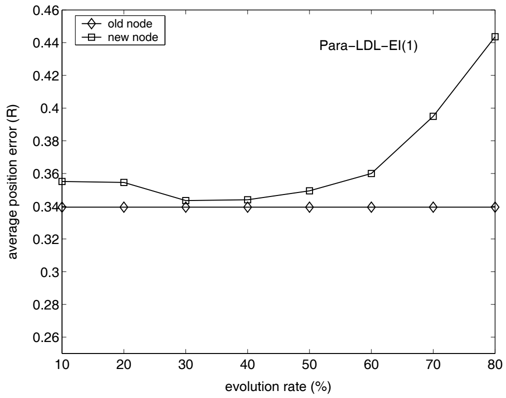

Figure 2. Average position error of Para-LDL-EI(1) with different evolution rate

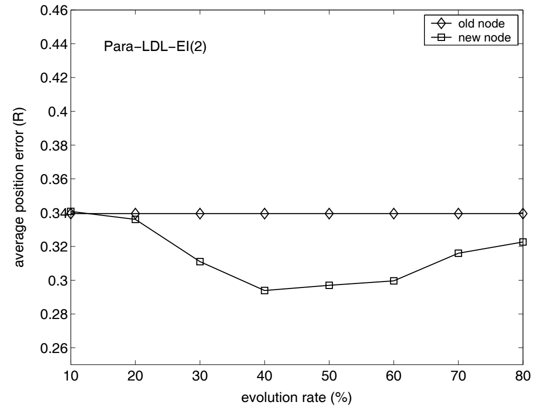

Figure 3. Average position error of Para-LDL-EI(2) with different evolution rate

The evolution rate is a term which refers to the percentage of *exhausted nodes.* For example, an evolution rate of 50% means that in 100 nodes, 50 nodes are exhausted and that many nodes are added for replacing the *exhausted nodes*. The simulations run as follows:

Stage 1: 100 nodes are randomly deployed. The nodes use Para-LDL to find their position.

Stage 2: After a period of time, a number of random nodes among the deployed nodes are turned off. Also, many new nodes are randomly deployed and work. The nodes use Para-LDL-EI to find their position.

The nodes' position error and communication overhead in each stage are recorded and analysed. Figures 2 and 3 demonstrate the average position errors of *new nodes*, which are produced by Para-LDL-EI(1) and (2) with different evolution rates. The average position error of old nodes, which is computed from stage 1, is also shown in figures for comparison. Figure 2 shows that Para-LDL-EI(1) can work efficiently when the evolution rate is smaller than 60%. Within this evolution rate, the average position error of *new nodes* is approximately 2% higher than that of old nodes. The average position error is at a minimum when the evolution rate is between 30% and 50%.

A striking observation from Figure 3 (Para-LDL-EI(2)) is that the average position error decreases in the next generation and the degree of reduction depends on the evolution rate, i.e. it is highest when the evolution rate is within 40-50% and lower with other rates. Note that although *exhausted nodes* are not working when *new nodes* are deployed, their information is still maintained by *live nodes*. This information is implicitly represented in terms of distance-related variables. For example, a *live node* stores *snh*=4 to anchor *k*, which was achieved from a path which passed through an *exhausted node*. Through LDL-EI, this information is used by a *new node* when it receives a reply packet from the *live node*. With this information, the *new node* has more opportunity to choose the appropriate value of distance-related variables. This leads to higher estimated position accuracy. This result is the same as the effect when the node density is increased as discussed in [14]. This effect is called the pseudo-density effect. Para-LDL-EI(1) also takes the advantage of the pseudo-density effect, which results in its higher accuracy when the evolution rate is within 30-50%.

Para-LDL-EI(2) exploits this effect much more efficiently by using fake beacon packets, which allows anchors to re-estimate the distance function coefficients.

Note that a lower evolution rate is equivalent to the smaller density increase, which results in a lower accuracy than that of a higher evolution rate. On the other hand, if the evolution rate is too high, the number of *live nodes*, who maintain the information of the old network and disseminate it in the next generation network, is also small; consequently, the amount of information of the old network remaining in the next generation is small. As a result, the pseudo-density effect is smaller and the node accuracy is lower than for those of a lower evolution rate.

The pseudo-density effect is saturated after several generations. Figure 4 shows this fact. This result is drawn from the simulation in which a sensor network evolves through several generations with the same evolution rate, i.e. 50%. The figure demonstrates that the average position accuracy steadily reduces until the third generation. However, it does not significantly change after that.

Figure 5 shows the communication overhead of Para-LDL-EI(1) and Para-LDL-EI(2). An important observation is that Para-LDL-EI(1) requires very low overhead, i.e. 1-2 packets per node with every evolution rate, whereas that of Para-LDL-EI(2) is higher and increases when the evolution rate increases.

With these results, we recommend that in cases where a low communication overhead is strictly required and the evolution rate is small, LDL-EI(1) is the appropriate choice, otherwise LDL-EI(2) is the better one. In addition, the evolution rate of 40-50% is the most effective, in terms of position accuracy.

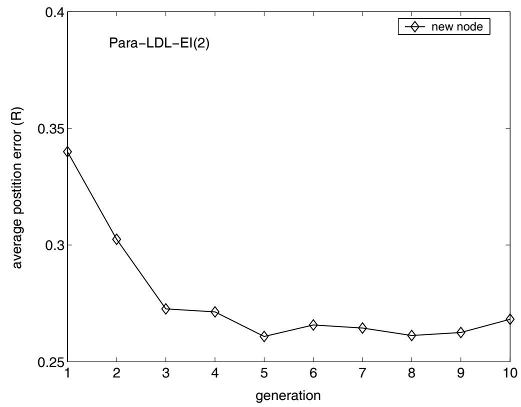

Figure 4. Average different error with different generations

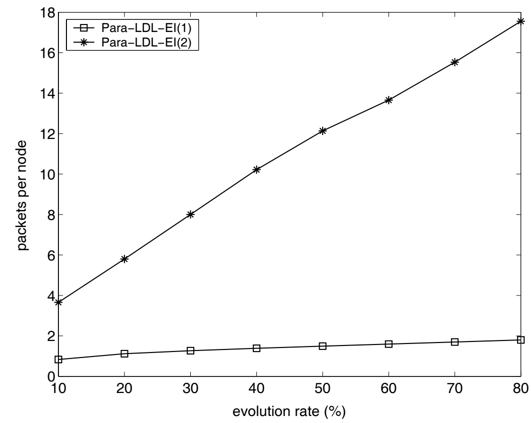

Figure 5. Communication overhead

5. Conclusion

This paper is the first in discussing the localization algorithm in an evolvable sensor network. The proposed extension obviously satisfies the requirements in that the estimated position accuracy of new nodes is at the same degree with that of old nodes and the communication overhead is reasonable small. Moreover, by using available information in an old network, the performance of the extension exceeds that of the original, i.e. with evolution rate from 20 to 80%, Para-LDL-EI(2) produces new nodes' position accuracy higher than that of old nodes.

References:

[1] V. Rodoplu and T. H. Meng, Minimum Energy Mobile Wireless Networks", *IEEE Journal Selected Areas in Communications, vol. 17, no. 8,* Aug. 1999, pp. 133344.

[2] M.B Srivastava, R.Muntz and M.Potkonjak, Smart Kindergarten: Sensor-based Wireless Networks for Smart Developmental Problem-solving Environments, *Proceedings of the seventh annual international conference on Mobile computing and networking, Mobicom* 2001, 132-139.

[3] http://www.greatduckisland.net/

[4] B. Karp and H.T. Kung. "GPSR: greedy perimeter stateless routing for wireless networks". *ACM MOBICOM* 2000.

[5] S. Ratnasamy, B. Karp, L. Yin, F. Yu, D. Estrin, R. Govindan, S. Shenker. "GHT: A Geographic Hash Table for Data-Centric Storage". *ACM WSNA* 2002.

[6] D. Niculescu and B. Nath. Ad hoc positioning system. In *IEEE Globecom* 2001, volume 5, pages 2926 – 2931, April 2001.

[7] 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*, Atlanta, GA, September 2002.

[8] K. Langendoen, N. Reijers, Distributed Localization in Wireless Sensor Networks: A Quantitative Comparison, In *Computer Networks: The International Journal of Computer and Telecommunications Networking,* Volume 43 , Issue 4 (November 2003), page 499 - 518.

[9] L. Kleinrock and J. Silvester, Optimum Transmission Radii For Packet Radio Networks Or Why Six Is A Magic Number. *Proc. Natnl. Telecomm. Conf.*, pages 4.3.1-4.3.5, 1978

[10] T. He, C. Huang, B.M. Blum, J. A. Stankovic, T. Abdelzaher, Range-Free Localization Schemes for Large Scale Sensor Networks, In *MobiCom* 2003.

[11] Y. Xu, J. Heidemann, D. Estrin, Geography-informed Energy Conservation for Ad-hoc Routing, *In the Proceedings of the Seventh Annual ACM/IEEE International Conference on Mobile Computing and Networking (ACM Mobicom), July 2001.*

[12] Y. Xu, S. Bien, Y. Mori, J. Heidemenn, D. Estrin, Topology Control Protocols to Conserve Energy in Wireless Ad-hoc Networks, *Technical Report*, Jan 2003.

[13] R. Nagpal, H. Shrobe, J. Bachrach, Organizing a Global Coordinate System from Local Information on an Ad Hoc Sensor Network, In *the 2nd International Workshop on Information Processing in Sensor Networks* (IPSN '03), Palo Alto, April, 2003.

[14] DuyBach Bui, Daeyoung Kim, A Localization Algorithm With Learning-Based Distance, *Technical Report*, April 2004. http://resl.icu.ac.kr/~bduybach/publications/bach-ldl.pdf