Improvements on Scalable Distributed Least Squares Localization for Large Wireless Sensor Networks

Ralf Behnke, Jakob Salzmann, Dirk Timmermann
Institute of Applied Microelectronics and Computer Engineering
University of Rostock
18057 Rostock, Germany
{*ralf.behnke, jakob.salzmann, dirk.timmermann*}*@uni-rostock.de*

***Abstract*—Wireless Sensor Networks (WSNs) have been of high interest during the past couple of years. One of the most important aspects of WSN research is location estimation. A good solution of fine grained localization is the Distributed Least Squares (DLS) algorithm, which splits the costly localization process in a complex *precalculation* and a simple *postcalculation*. The latter is performed on constrained sensor nodes, finalizing the localization by adding locale knowledge. This approach lacks for large WSNs, because cost of communication and computation theoretically increases with network size. In practice the approach is even unusable for large WSNs. An important assumption of DLS is that each blind node is able to communicate with each beacon node to receive the precalculation and to determine distances to beacon nodes. This restriction have been overcome by scalable DLS (sDLS), which enabled to use the idea of DLS in large WSN for the first time. Although, sDLS has lower cost of computation than DLS, for large networks, this cost, caused by matrix updates, is pretty high. In this work an adaptation of sDLS is presented, which dramatically reduces cost of computation by circumventing matrix updates as often as possible.**

***Keywords*-wireless sensor networks; localization; scalability; optimization**

I. Introduction

Recent technological advances have led to the development of tiny wireless devices, which are able to sense their environment, compute simple tasks and exchange data among each other. Interconnected assemblies of such devices, called Wireless Sensor Networks (WSNs), are commonly used to observe large inaccessible areas. In many applications of WSNs, knowledge of nodes' locations is mandatory for a meaningful interpretation of sensed data. Location-awareness is not only necessary to assign a location to measured values but also to perform geographic routing [1] or location based clustering [2]. Due to existing limitations in terms of size and energy consumption, local positioning within the network is preferred over utilizing common positioning systems like GPS. Therefore, the presence of location-aware sensor nodes is typically assumed which are referred to as beacon nodes. These nodes know their own position a priori or via common positioning systems. The remaining nodes, which we refer to as blind nodes, are assumed to use communication and any kind of distance estimation or neighborhood information to estimate their position with the help of beacon nodes.

Existing localization techniques can be divided into coarse-grained and fine-grained localization. Commonly this classification reflects the trade off between precision and resource consumption of the corresponding techniques. While coarse-grained approaches like Adaptive Weighted Centroid Localization (AWCL) [3] require less communications and computations and provide lower precision estimates, fine-grained approaches aspire to an exact localization, which is achieved by use of costly computations. Distributed Least Squares (DLS) [4] combined high precision with relatively low complexity. It splits the costly localization calculation into *precalculation* and *postcalculation*. Independent from a specific blind node, the complex precalculation is performed on a high-performance sink. The remaining postcalculation is less complex and performed on resource-constrained blind nodes.

As a major drawback of DLS, it presumes that each blind node is able to receive precalculation data from the sink and is able to estimate its distance towards each beacon node. This makes DLS infeasible for use in large multi-hop networks which represents one of the most interesting scenarios for WSNs. Furthermore, communication and computational effort on each blind node increases with the number of beacon nodes and, therefore, with the applied network size.

The described drawbacks have been overcome by scalable DLS (sDLS) [5], still saving the idea of DLS. The use of individual precalculations instead of only one precalculation for the whole network, as used by DLS, enabled sDLS to be used in large WSNs. By the use of sDLS the computational cost becomes independent from network size, also communication effort scales better. Although sDLS outperforms DLS in all aspects, given that the network is large enough, costly updating of precalculations demands for improvement. The actual work improves cost of computation of sDLS by abstaining from updating precalculations as much as possible.

The remainder of the paper is organized as follows. Section II covers the original DLS algorithm as well as the newer sDLS algorithm. In Section III, the improved ap-

978-1-4244-6857-7/10/$26.00 ©2010 IEEE

proach of sDLS, referred to as sDLS - no update ($sDLS^{nu}$), is described. Section IV describes the simulation environment which was used to evaluate the algorithm. Simulation results are presented in Section V. Finally, Section VI summarizes the presented work and covers future work.

II. Related Work

The DLS algorithm as well as the sDLS approach can be divided into two parts. Firstly the arithmetical part is to be considered, followed by the algorithmic part.

A. Arithmetic Background

The system of equations which have to be solved for localization of a blind node is originally build by distance equations as given in equation (1). Here x and y is the unknown position of a blind node. The known position of a beacon node is denoted as x_i and y_i, while the distance between both nodes is denoted as r_i. The number of beacon nodes, utilizable for localization is given as m.

$$(x-x_i)^2+(y-y_i)^2 = r_i^2 \quad (i \in I; I = \{1, 2, \ldots, m\}) \tag{1}$$

To linearize this system of equations an arbitrary beacon node is used as linearization tool [6], denoted with index L and utilized as given in equation (2). This reduces the number of equations by 1.

$$\begin{gathered}(x - x_L + x_L - x_i)^2 + (y - y_L + y_L - y_i)^2 = r_i^2 \\ (L \in \{1, 2, \ldots, m\}, i \in \{1, 2, \ldots, m\} \setminus L)\end{gathered} \tag{2}$$

Restructuring the equations leads to equation (3), where r_L denotes the distance between blind node and linearizer, r_i is the distance between blind node and beacon node and d_{iL} denotes the distance between linearizer and beacon node.

$$\begin{aligned}(x - x_L)(x_i - x_L)+& \\ (y - y_L)(y_i - y_L) &= \frac{1}{2}\left[r_L^2 - r_i^2 + d_{iL}^2\right] \\ &= b_{iL}\end{aligned} \tag{3}$$

The restructured system of equations can be written in matrix form as

$$\mathbf{Ax} = \mathbf{b} \tag{4}$$

with

$$\begin{gathered}\mathbf{A} = \begin{pmatrix} x_{k_1} - x_L & y_{k_1} - y_L \\ x_{k_2} - x_L & y_{k_2} - y_L \\ \vdots & \vdots \\ x_{k_n} - x_L & y_{k_n} - y_L \end{pmatrix}, \\ \mathbf{x} = \begin{pmatrix} x - x_L \\ y - y_L \end{pmatrix}, \mathbf{b} = \begin{pmatrix} b_{k_1 L} \\ b_{k_2 L} \\ \vdots \\ b_{k_n L} \end{pmatrix}\end{gathered} \tag{5}$$

In equation (5), matrix $\mathbf{A}$ only consists of beacon position data, while $\mathbf{b}$ contains distances between beacon nodes and blind nodes. Therefore calculations on $\mathbf{A}$ can be performed at a powerful sink outside the WSN. The localization will be finalized on each blind node by performing the remaining part of the calculation. The beacon nodes, used for localization, are denoted with indices $K = \{k_1, k_2, \ldots, k_n\}$ with $K \subseteq \{I \setminus L\}$. DLS and sDLS differ in the choice of K. DLS uses only one system of equations, containing all beacon nodes, which is used for localize all blind nodes, using the first beacon node for linearization. Therefore it exists only one set K for the whole network, given as $K = \{I \setminus L\}$ with $L = 1$. In a large WSN no blind node will be able to access all beacon nodes directly for data exchange or distance estimation. To overcome this problem, sDLS uses individual sets of beacon nodes for localization, one for each beacon node. Each set contains only the beacon node itself and beacon nodes within its communication range. Therefore sDLS uses m systems of equations with $K_i \subseteq \{I \setminus L_i\}$, $L_i = i$ and $i \in I$.

Another difference between DLS and sDLS is how the linear system of equations is to be solved. DLS uses normal equations, which leads to a restructuring of equation (4) as given in equation (6). In this case $\mathbf{A}_p = \left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T$ and $\mathbf{d}_p = \mathbf{d}^2$ present the precalculation, performed on the sink.

$$\mathbf{x} = \left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T\frac{1}{2}\left[r_L^2 - \mathbf{r}^2 + \mathbf{d}^2\right] \tag{6}$$

sDLS takes into account that there will be beacon nodes included in a precalculation that are not in the communication range of a blind node and vice versa. Therefore the precalculated data has to be updated on the blind node. For that reason sDLS uses qr-decomposition to solve the linear system of equations. This allows updating and downdating of $\mathbf{Q}$ and $\mathbf{R}$ [7]. Doing so, with $\mathbf{A} = \mathbf{QR}$ and $\mathbf{R}$ upper triangular, the system of equations given in equation (4) becomes restructured as given in equation (7). The precalculation is presented by $\mathbf{Q}^T$, $\mathbf{R}$ and $\mathbf{d}_p = \mathbf{d}^2$, individually determined for each beacon node.

$$\mathbf{Rx} = \mathbf{Q}^T\frac{1}{2}\left[r_L^2 - \mathbf{r}^2 + \mathbf{d}^2\right] \tag{7}$$

B. The Algorithms

The DLS algorithm consists of four steps as stated below. Direct communication between blind nodes and all beacon nodes is required for distance estimation.

Step 1 - *Initialization Phase:*
All beacons send their position to the sink.
Step 2 - *Precalculation Phase:*
Sink computes $\mathbf{A}_p$ and $\mathbf{d}_p$.
Step 3 - *Communication Phase:*
Sink sends precalculated data to all blind nodes.
Step 4 - *Postcalculation Phase:*
Blind nodes determine distance to every beacon node,

receive precalculated data and estimate their location by solving the postcalculation.

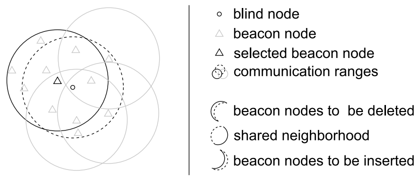

Figure 1. Blind node selects precalculation of closest beacon node

The sDLS algorithm uses individual precalculations for each beacon node, consisting of the beacon node itself, used as linearization tool, and beacon nodes within its communication range. This ensures that the linearizing node, which can not be deleted without a complete recalculation, is within the blind nodes' communication range. A blind node is expected to use the precalculation of the closest beacon node. As illustrated in figure 1 the number of beacon nodes that have to be added by the blind node as well as those that have to be deleted from precalculation is relatively small. To enable individual precalculations, each beacon node discovers beacon nodes in its communication range and provides this information to the sink. Containing two additional steps, the sDLS algorithm can be stated as follows:

Step 1 - *Discovery Phase:*
Each beacon node sends a local broadcast to discover neighboring beacon nodes.

Step 2 - *Initialization Phase:*
Each beacon node sends its position and a list of its neighbors to the sink.

Step 3 - *Precalculation Phase:*
Sink computes $\mathbf{Q}^T$, $\mathbf{R}$ *and* $\mathbf{d}_p$ *individually for each beacon node.*

Step 4 - *Distribution Phase:*
Sink sends precalculated data to beacon nodes.

Step 5 - *Communication Phase:*
Beacon nodes send precalculated data to blind nodes.

Step 6 - *Postcalculation Phase:*
Blind nodes determine distances to accessible beacon nodes, receive precalculated data, update precalculation and estimate their position, solving the postcalculation.

C. Computational Cost of sDLS

Above mentioned updates of individual precalculations, take a high amount of computation, caused by matrix updates of Q and R. Figure 2 shows the mean number of operations, which have to be performed by each blind node to estimate its position using sDLS. Besides overall cost, including updates and final position estimation, cost of final estimation of a blind nodes' position is depicted separately. It is shown, that updating the precalculation takes up to 95% of the overall cost. Therefore the following approach aspires to save this cost by abstaining from update operations.

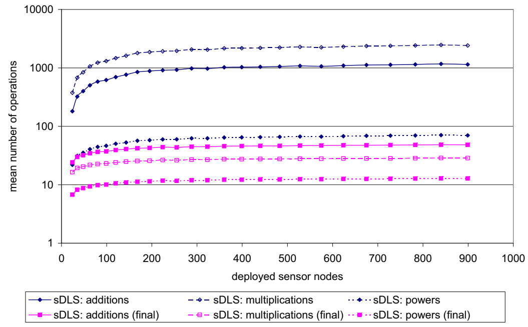

Figure 2. Mean cost of computation of sDLS

III. Reducing cost of computation

In contrast to DLS the postcalculation phase of sDLS, performed on blind nodes, was extended by an update process, inserting and deleting beacon nodes to and from precalculations, respectively. Therefore cost of computation consists of the following parts:

1) deletion of inaccessible beacon nodes
2) insertion of accessible beacon nodes
3) estimate of position, using postcalculation

As previously shown, solving the system of equations takes up only a small part of the computation, while deletion and insertion takes the most. In the following Subsections it is described, how sDLS^{nu} reduces the number of insert operations and abstains from delete operations.

A. Saving Insert Operations

In most cases insert operations can be easily saved by leaving out additional beacon nodes. Therefore the sDLS^{nu} approach abstains from complete the precalculation with additional beacon nodes as much as possible. An impairment of localization accuracy is expected due to the reduced number of beacons. Since the system of equations can be only solved if at least the linearizer node and two additional beacon nodes are used for calculation, sDLS^{nu} still uses matrix updates to insert additional beacon nodes, if necessary to meet this condition. This ensures that localization does not fail in consequence of the sDLS^{nu} approach.

B. Saving Delete Operations

As known from literature [7], deletion is even more expensive than insertion. Therefore, saving delete operations is most important. In contrast to insert operations, the system of equations becomes unsolvable if distance information of beacon nodes, included in precalculation, is absent. There are two possibilities to solve the system of equations with inaccessible beacon nodes. The first one is to update precalculations as done by sDLS. The second possibility, which is used by sDLS^{nu}, is to estimate missing distance information.

As shown in figure 1, blind node and inaccessible beacon nodes are on opposite sites. As the blind node selected the closest beacon node for precalculation, the distance between blind node and linearizing node, estimated by the blind node, tends to be small. Squared distances between linearizer and inaccessible beacon nodes are already included in precalculated data, as given in equations (3) and (7). Therefore, sDLS^{nu} uses the sum of both distances, to estimate distances towards inaccessible beacon nodes. Using this approach, sDLS^{nu} completely abstains from delete operations. The additional cost for distance estimation is to solve equation (8).

$$r_i = r_L + \sqrt{d_{iL}^2} \quad (8)$$

IV. Simulations

To verify performance of sDLS^{nu} and sDLS without modification, the MatLab based network simulator Rmase is used [8]. The simulator provides a realistic radio communication model, including spatial and temporal normal distributed fading, random transmission errors, collisions and a CSMA-CA MAC layer. As previously done in [5], a static bidirectional spanning-tree routing was used for communication.

A random deployment of n^2 nodes within a field of $n * n$ arbitrary distance units (adus) was utilized. While the first node was always used as sink, the remaining nodes have been randomly chosen as blind nodes (50%) or beacon nodes (50%). The field size parameter n was varied from 5 to 30. The average communication range, given by the radio model was 3 adus. For each field size the average over 100 simulations has been determined. In each simulated network sDLS as well as the modified sDLS^{nu} algorithm have been performed concurrently.

V. Results

The new sDLS^{nu} approach is compared to sDLS in terms of localization accuracy and cost of computation on blind nodes. Additionally the number of beacon nodes used by a blind node as well as the update-rate is analyzed.

A. Update Performance

The aim of the sDLS^{nu} approach was to reduce the number of matrix updates as much as possible by abstaining from insert operations as well as delete operations. Figure 3 shows the mean number of beacon nodes used for localization by blind nodes. While sDLS uses about 12 beacon nodes, the number of beacon nodes, used by sDLS^{nu} is slightly higher. Regarding the number of insertions and deletions, also depicted in figure 3, more beacon nodes become deleted than inserted, using sDLS. The difference between both operations is similar to the difference of used beacon nodes. While the number of deletions used by sDLS^{nu} is identical to zero, the number of inserts only tends to zero and decreases with network size.

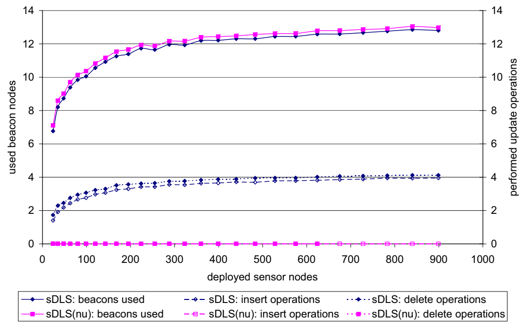

Figure 3. Mean number of update operations performed by a blind node

B. Cost of Computation

To quantify cost of computation, the number of operations has been counted on each blind node. Due to the different complexity, three kinds of operations have been analyzed. Additions and subtractions have been summed up as additions, multiplications and divisions are combined in multiplications, and powers include squares and square roots. In figure 4 the overall computations, i.e. including update operations and final position estimation, of both algorithms are illustrated. The logarithmic representation shows that the number of multiplications, which is strongly affected by update operations, was most significantly decreased using sDLS^{nu}. About 98% of multiplacations and 95% of additions have been saved in large WSNs. Even the number of powers and square roots have been reduced by about 75%.

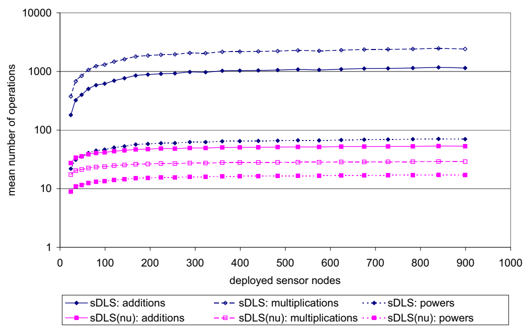

Figure 4. Mean number of operations performed on a blind node

In contrast to figure 4, only the final calculation part i.e. position estimation after the update process, is shown in figure 5. Due to the slightly increased number of used beacon nodes, the number of computations within this part have been also slightly rised. But as illustrated in figure 4, this increase is marginal compared the achieved savings.

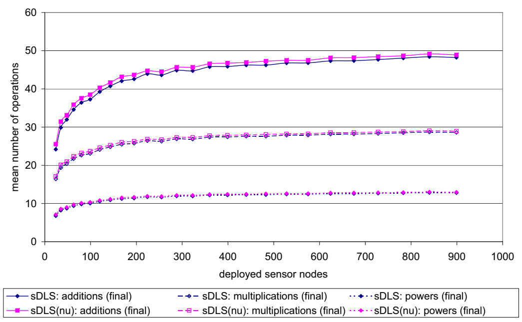

Figure 5. Mean number of operations performed for final determination

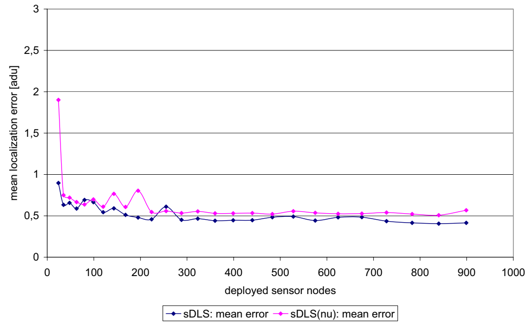

Figure 7. Mean error of localization over total number of deployed nodes

Additionally, figure 6 compares overall cost of the new sDLS^{nu} approach and cost, spend for the final calculation of sDLS. It is shown that the cost is nearly the same. The additional cost is caused by the performed distance estimation and the number of used beacon nodes, which is slightly higher.

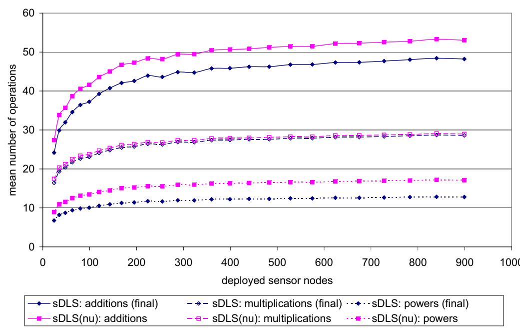

Figure 6. Mean number of operations performed for final determination (sDLS) and complete determination (sDLS^{nu}), respectively

C. Localization

To compare performance of localization, each blind node determines its localization error as distance between real position and estimated position. The results in figure 7 show that the averaged localization error is slightly higher, using sDLS^{nu}. The assumed reason is that the beacons used by sDLS^{nu} tend to be arranged less polydirectional. Compared to sDLS, sDLS^{nu} left out beacons at one side and uses additional beacons on the opposite side.

VI. Conclusion

The presented sDLS^{nu} approach provides significant improvements, concerning cost of computation, saving about 95%. Accuracy of localization is only slightly impaired. Furthermore, it causes no additional communication.

While cost of computations have been significantly reduced in this work, cost of data transmission was not addressed until now. One possibility to reduce data transmission may be to share precalculated data or parts of it amoung nearby beacon nodes. A suitable solution to share precalculation among beacon nodes may be a cluster based structure like 4-MASCLE [2].

It is also a challenge to improve localization to achieve at least accuracy of the original sDLS.

Acknowledgment

This work was supported by the German Research Foundation under grant number BI467/17-2 (keyword: Geosens2)

References

[1] K. Akkaya and M. F. Younis, “A survey on routing protocols for wireless sensor networks,” *Ad Hoc Networks*, vol. 3, no. 3, pp. 325–349, 2005.

[2] J. Salzmann, R. Behnke, M. Gag, and D. Timmermann, “4-MASCLE - Improved Coverage Aware Clustering with Self Healing Abilities,” *International Symposium on Multidisciplinary Autonomous Networks and Systems (MANS 2009)*, Jul. 2009.

[3] R. Behnke and D. Timmermann, “AWCL: Adaptive Weighted Centroid Localization as an efficient Improvement of Coarse Grained Localization,” *Positioning, Navigation and Communication, 2008. WPNC 2008. 5th Workshop on*, pp. 243–250, Mar. 2008.

[4] 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,” *Distributed Computing in Sensor Systems*, pp. 514–528, 2006.

[5] R. Behnke, J. Salzmann, D. Lieckfeldt, and D. Timmermann, “sDLS - Distributed Least Squares Localization for Large Wireless Sensor Networks,” *International Workshop on Sensing and Acting in Ubiquitous Environments*, Oct. 2009.

[6] W. S. Murphy and W. Hereman, “Determination of a position in three dimensions using trilateration and approximate distances,” Tech. Rep., 1999.

[7] D. S. Watkins, *Fundamentals of matrix computations*, 2nd ed., ser. Pure and Applied Mathematics. New York: Wiley-Interscience [John Wiley & Sons], 2002.

[8] Y. Zhang, M. Fromherz, and L. Kuhn, “Rmase: Routing modeling application simulation environment,” 2009, http://www2.parc.com/isl/groups/era/nest/Rmase/.