Minimizing Cost of Scalable Distributed Least Squares Localization

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. As a good solution of fine grained localization Reichenbach et al. introduced the Distributed Least Squares (DLS) algorithm, which splits the costly localization process in a complex *precalculation* and a simple *postcalculation* which is performed on constrained sensor nodes to finalize the localization by adding locale knowledge. This approach lacks for large WSNs, because cost of communication and computation theoretically increases with the 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 WSNs for the first time. In this work we present an adaptation of sDLS, that reduces the cost of update operations which are an integral part of sDLS.**

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

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 WSN, 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][2] or location based clustering as described in [3]. 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. Coarse-grained approaches like Centroid Localization (CL) [4] and Adaptive Weighted Centroid Localization (AWCL) [5] often abstain from exact distances, require less communications and computations and provide lower precision estimates. On the other hand, fine-grained approaches aspire to an exact localization with high precision, which is achieved by costly computations, using estimations of distances or angles. Achievable precision of such approaches, commonly based on a set of linear equations, hardly depends on distance or angle estimation, respectively. Distributed Least Squares (DLS) [6] firstly 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 in the network is able to communicate directly with the sink and is able to estimate its distance towards each beacon node. This makes the 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) [7], still saving the idea of DLS. Major changes enabled sDLS to be used in large WSNs, using individual precalculations instead of only one precalculation for the whole network, as used by DLS. 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 the precalculation demands for improvement. The actual work improves the cost of computation of sDLS by reducing the insert part of the update.

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 approach of sDLS, referred to as sDLS - no insert (sDLS$^{\text{ni}}$), 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 was developed to alleviate trade off between precision and cost of localization and provides localization with high precision and low cost [6]. The original approach 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).

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

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.

To linearize this system of equations a linearization tool [8] is used. Therefore an arbitrary beacon node is selected as linearizer, 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

$$\begin{gathered}\mathbf{A} = \begin{pmatrix} x_{k_1}-x_L & y_{k_1}-y_L \\ x_{k_2}-x_L & y_{k_2}-y_L \\ \vdots & \vdots \\ x_{k_n}-x_L & y_{k_n}-y_L \end{pmatrix}, \\ \mathbf{x} = \begin{pmatrix} x-x_L \\ y-y_L \end{pmatrix}, \mathbf{b} = \begin{pmatrix} b_{k_1L} \\ b_{k_2L} \\ \vdots \\ b_{k_nL} \end{pmatrix}\end{gathered} \tag{5}$$

In equation (5), the beacon nodes, used for localization, are denoted with indices $K = \{k_1, k_2, \ldots, k_n\}$ with $K \subseteq \{I \setminus L\}$. The choice of K shows one difference between DLS and sDLS. DLS uses only one system of equations which is used for localize all blind nodes. This system of equations contains all beacon nodes and uses the first one for linearization. Therefore it exists only one set K for the whole network which is given as $K = \{I \setminus L\}$ with $L = 1$. As mentioned before, in a large WSN no blind node will be able to access all beacon nodes directly for data exchange or distance estimation. The approach of sDLS to overcome this problem is to use individual sets of beacon nodes for localization, one for each beacon node. Each set contains only the beacon node itself and the 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 $i \in I$.

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 next difference between DLS and sDLS is how the linear system of equations are 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 $\mathbf{R}$ [9]. 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 is based on the precondition that each beacon node and each blind node is able to communicate with the sink. Furthermore each blind node have to be able to determine its distance to all beacon nodes, which requires direct communication to all beacon nodes.

The algorithm consists of four steps:

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.

sDLS abstains from a precondition as used in DLS. It uses individual precalculations for each beacon node, consisting of the beacon node itself and beacon nodes within its communication range. Therefore each beacon node discovers beacon nodes in its communication range and provides this information to the sink. 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. Except from the beacon node which is used for linearization, all beacon nodes can be deleted from the precalculation. To ensure that the linearizing beacon node is within the communication range of the blind node, the individual precalculation of a beacon node uses the beacon node itself as linearization tool.

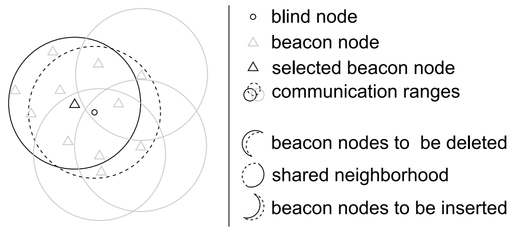

Figure 1. Blind node selecting precalculation of nearest beacon node to minimize update operations

Using sDLS, the initial algorithm is extended by 2 steps as follows:

Step 1 - *Discovery Phase:*
All beacons send a local broadcast to discover neighboring beacon nodes.
Step 2 - *Initialization Phase:*
All beacons send their position and a list of their 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 distance to accessible beacon nodes, receive precalculated data, update precalculation and estimate their own position by solving the postcalculation.

III. New Approach to Improve Cost of Computation

As described in [7], sDLS improved cost of communication and computation in comparison to DLS for large WSNs. In contrast to DLS the postcalculation phase was extended by an update process, inserting and deleting beacon nodes to and from a precalculation, respectively. Therefore cost of computation consists of the following parts:

1) delete inaccessible beacon nodes from precalculation
2) insert accessible beacon nodes into precalculation
3) estimate blind nodes' position by solving the remaining system of equations

It was already shown in [7] that solving the system of equations takes up only a small part of the computation, while deletion and insertion takes the most. To reduce the cost of computation, this approach aims to left out parts of the update process. While deletion of inaccessible beacon nodes is inevitable to make the system of equations solvable, insertion of additional nodes is an optional part of the update process. Therefore the sDLS^{ni} approach abstains form complete the precalculation with additional beacon nodes, as much as possible. That means, additional beacon nodes within the communication range of a blind node, are only used for localization, if the number of used beacon nodes would otherwise be less than two. This ensures that localization does not fail because of the sDLS^{ni} approach.

IV. Simulations

To verify performance of sDLS^{ni} and sDLS without modification, the MatLab based network simulator Rmase is used [10]. 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. The Rmase layer structure provides layers, implementing addressing, queuing, aggregation and routing. The original Rmase also includes an application simulation as a core component, which generates messages on each node.

For our simulations, this application simulation has been extracted from the Rmase core. It was replaced by an application layer which can hold various, realistic applications. As previously done in [7], a static bidirectional spanning-tree routing as well as an additional layer which turns collided packets into received packets to avoid overhead was used.

A random deployment of n^2 nodes within a field of $n * n$ arbitrary distance units (adus) was utilized. The first node was always used as sink, while 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^{ni} algorithm have been performed concurrently.

V. RESULTS

The new sDLS^{ni} approach is compared to the original sDLS in terms of localization accuracy and cost of computation on blind nodes. Since the new approach only includes changes on blind nodes, it does not affect data transmission, which is therefore not investigated in this work. Additionally the number of beacon nodes, used by a blind node as well as the update-rate is analyzed.

A. Update Performance

The sDLS algorithm uses matrix updates to adapt precalculated data sets to the beacon nodes in range of a blind node. The aim of the sDLS^{ni} approach was to keep the number of updates down by abstaining from insert operations as much as possible. The graph in figure 2 shows how many beacon nodes have been used by a blind node on average. Furthermore it is depicted how many beacon nodes have been deleted from the precalculation and how many beacon nodes have been added to the precalculation. The results show, that for large networks, sDLS^{ni} uses about 8 beacon nodes, localizing a blind node, while sDLS uses about 12 beacon nodes. As expected, the number of delete processes is the same for both approaches. The number of insert operations, which is about 4, using sDLS has been dramatically reduced by sDLS^{ni}. It is about 0.06 for small networks and decreases with network size. It is assumed that this is reasoned by blind nodes near the network border, having less beacon nodes in its communication range than other blind nodes.

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 of operations 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 3 the overall computations, i.e. including update operations and final position estimation, of both algorithms are illustrated. As the number of multiplications is strongly affected by update operations, it was significantly

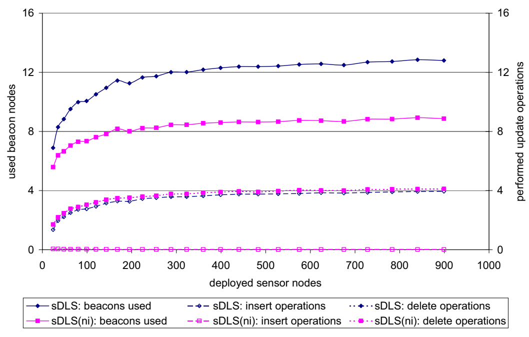

Figure 2. Mean number of update operations performed by a blind node

decreased using sDLS^{ni} as depicted. Also the number of additions was reduced. The number of powers and square roots, which was initially small was also slightly reduced. Compared to sDLS, cost of computation was reduced by about 12%.The saving in computation increases with the number of beacon nodes, used for localization.

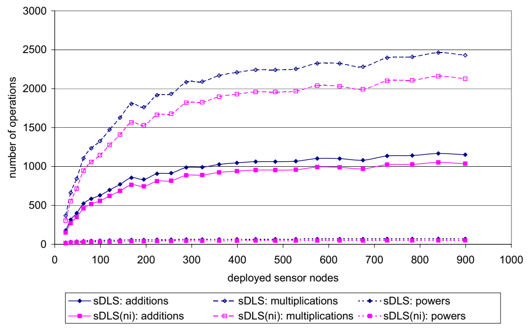

Figure 3. Mean number of operations performed on a blind node

In contrast to figure 3, only the final calculation part i.e. position estimation after the update process, is shown in figure 4. Similar to the before mentioned results, cost of computation has been significantly improved. Here, most saving was taken by additions, followed by multiplications and powers.

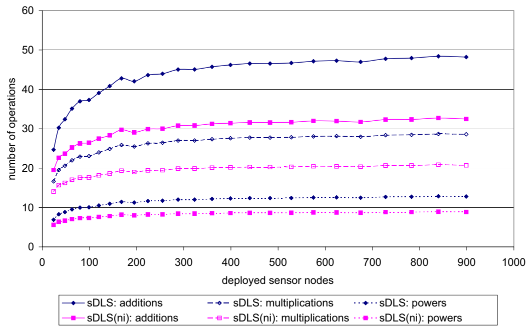

Figure 4. Mean number of operations performed for final determination on a blind node

C. *Localization*

To compare performance of localization, each blind node estimates its position with both approaches. After successful localization, the distance between real position and estimated position is regarded as localization error. The results in figure 5 show that the averaged localization error of sDLS^{ni} slightly exceeds that of sDLS. Due to the fact, that the existing outliers mainly appear on sDLS^{ni}, it is assumed that they are reasoned by few blind nodes with few beacon nodes in range, with disadvantageous arrangement. This is affirmed by the fact, that the difference between both approaches decreases for large WSNs, which indicates the decreasing influence of blind nodes near to the network border.

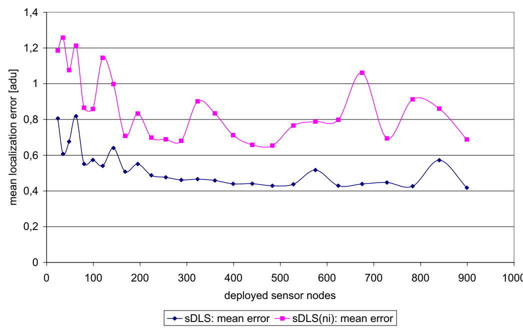

Figure 5. Mean error of localization over total number of deployed nodes

VI. Conclusion

The presented sDLS^{ni} approach significantly improves sDLS concerning cost of computation. Saving of computation is about 12%. The very efficient improvement comes only with a small impairment of localization accuracy.

As it is known from literature, downdating a matrix takes more computations than updating a matrix. Therefore, the results illustrated in figure 3 and figure 4 showed that abstaining from insert beacon nodes saves computation, but the larger part of the update process obviously is taken by deletion of beacon nodes. Therefore it is our aim to reduce number of deletions in future work.

Furthermore, cost of computation can be improved by a modification of the precalculated data, sets provided to the blind nodes. Also a well-directed selection of precalculated data by the blind node may lead to an improvement in cases when blind nodes fail in selecting the closest beacon node. Also data transmission, which we did not mention in this work, can be significantly reduced if beacon nodes would share precalculated data or parts of it. In general a modified precalculation can reduce data transmission. Using a cluster based structure like 4-MASCLE [3] may be a suitable solution to share precalculations among beacon nodes.

Acknowledgment

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

This paper has been accepted for publication in the proceedings of WSN Workshop that had been held in conjunction with NTMS 2009.

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. Al-Karaki and A. Kamal, “Routing techniques in wireless sensor networks: a survey,” *Wireless Communications, IEEE*, vol. 11, no. 6, pp. 6–28, Dec. 2004.

[3] 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.

[4] N. Bulusu, J. Heidemann, and D. Estrin, “GPS-less low cost outdoor localization for very small devices,” *IEEE Personal Communications Magazine*, vol. 7, no. 5, pp. 28–34, Oct. 2000.

[5] 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.

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

[7] 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.

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

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

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