Intensity-based Event Localization in Wireless Sensor Networks

Markus Waelchli, Matthias Scheidegger, Torsten Braun

Intensity-based Event Localization in Wireless Sensor Networks[1]

Markus Waelchli, Matthias Scheidegger and Torsten Braun
Institute of Computer Science and Applied Mathematics
University of Bern
{waelchli, mscheid, braun}@iam.unibe.ch

***Abstract*— Event detection and event localization are inherent tasks of many wireless sensor network applications. The inaccuracy of sensor measurements on the one hand and resource limitations on the other make efficient event localization a challenging problem. In this paper we propose a fully distributed localization scheme that consists of two algorithms. The distributed election-winner notification algorithm (DENA) performs the determination of the closest sensor node to an event and notifies all other nodes about that winner. The intensity-based localization algorithm (ILA) provides a signal independent position estimation of the event and is calculated at the winner node. The novelty of the ILA algorithm is its independence from the kind of signal emitted by an event. In contrast, it solely requires knowledge about the intensity of an event. The location of an event can thus be estimated without pre-knowledge about the nature of the event and with fewer constraints on the sensor hardware. These properties constitute the practicability of the algorithm in generic applications.**

I. Introduction

In sensor network applications event detection and localization are common features which imply two main challenges, namely how to observe a possible event location in a distributed manner and how to compute the location of an event efficiently and accurately. Due to battery and resource constraints on the sensor nodes some present approaches shift the computational burden of estimating the location of a node away from the sensor nodes to a sink node with more computational power and more memory. The main disadvantage of these approaches is the increased data traffic to provide the sink node with the necessary information to localize the event. The observation of an event on the other hand is commonly done by building clusters around predefined locations. Building up these clusters involves always communication overhead. This overhead is increased in wireless sensor networks, where battery constraint devices may follow sleep cycles to save battery power. Consequently, exchanging information may be expensive due to frequent topology changes or synchronization overhead. For these reasons, we intend to provide a fully distributed event detection framework that avoids the drawbacks of increased data traffic between the sensed area and the base station and does not need any cluster formation.

[1]The work presented in this paper was supported (in part) by the National Competence Center in Research on Mobile Information and Communication Systems (NCCR-MICS), a center supported by the Swiss National Science Foundation under grant number 5005-67322.

In our approach, the detection of an event (e.g fire burst) is observed as a set of derived values simultaneously sensed by a node (e.g. increased temperature, significant shockwave, etc.). Furthermore, the significance of the event can be determined by the sensor nodes, i.e. an event can be distinguished from background noise. This can be achieved by the use of fuzzy logics or probability theory. The task of filtering this background noise is not subject of this paper and will be considered in future work. Furthermore, the event is decreasingly observable the farther away a node is. We propose that considering these requirements all nodes in the relevant region can derive the intensity with which they sensed a certain event. Moreover, this intensity can be inferred as the barycenter of the set of deviated values sensed by the node. Thereby, each sensed value satisfies a certain membership function, e.g. 80° Celsius could have a membership degree of 0.8 in relation to the predicate 'hot'. A key idea of our approach is to use fuzzy logic mechanisms to classify the deviated values on the one hand and to infer the intensity of the event from these values on the other. The intensity of an event is consequently represented as a value in the interval [0, 1].

Using these derived values we propose to use a distributed election algorithm that determines the relevant subset of sensors, which are responsible for handling the event further, e.g. sending their information to a base station, or aggregate the information among each other. The determination of the relevant sensor nodes is performed fully distributed with a minimal overhead of information exchange.

II. Related Work

Event detection and localization are intrinsic features of wireless sensor networks. Much work in this context has already been done. The proposed schemes differ in the way they get range estimations and how they perform event observations. Some localization approaches ([1], [2]) depend on either a central instance such as a sink node or a cluster head, where the measurements from the sensors in the field are collected and the event location is computed. In [1] The distance of a sensor node to an event is approximated using the time of arrival (TOA) of the signal emitted by the event. The TOA values are routed together with the sensor node positions to a sink node, where the location of the event is computed as the maximum of a four-dimensional consistency function. [2] uses a cluster head approach to track the location of an event. Thus,

the overhead of sensor node to sink communication is avoided but additional communication to maintain the cluster structure is necessary. Another approach is Sextant proposed by [3]. Sextant uses Bézier regions to represent the locations of nodes as well as of events and does therefore without knowledge of the node positions. Additionally, Sextant is independent of a central instance. However, network properties which are needed by Sextant have to be disseminated in a restricted area, whenever one of them changes. Other approaches ([4], [5]) are mainly concerned in enabling and establishing group communication and data aggregation in a predefined area which has to be observed. Both algorithms require cluster formation what leads to extra communication overhead. A distributed algorithm for object tracking has been proposed by [6]. This approach supports event detection and tracking, but no event localization. A moving object is thereby tracked by a changeable cluster of nodes.

A common approach to estimate node locations is triangulation against known positions derived from reference points. APS [7] and GPS-Free [8] use angle of arrival (AOA) and time of arrival (TOA) respectively to calculate the position of a node. Both schemes are not practical for event localization as they depend on specific hardware. In contrast, we will propose a multilateration scheme that does only depend on the feasibility to sense an event on a sensor node.

III. Distributed Election-winner Notification

A key problem of event detection is the difficulty to identify and organize the sensor nodes, which are relevant for the event in a distributed manner with as little communication overhead as possible. To fulfill this task we propose the fully distributed election-winner notification algorithm (DENA). The DENA algorithm basically consists of two parts. In a first step the node closest to the event determines itself as winner node. In a second step the winner node notifies all other nodes about its election. The principle of the algorithm is depicted in Fig. 1.

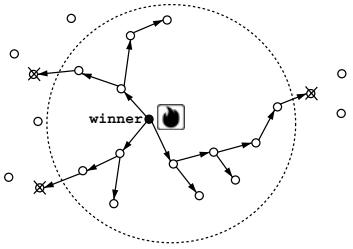

Fig. 1: Notification of election losers by winner node

The winner node broadcasts a notification message to inform all other nodes about its election. The notification message is thereby only retransmitted by sensor nodes that have overheard the event, i.e. the nodes bordered by the dotted line in Fig. 1. All other nodes having received the notification message (canceled in Fig. 1) simply ignore it. Additionally to its basic functionality of electing the winner node and distributing the notification message, the DENA algorithm offers functionality to provide the localization algorithm (see Section IV) with the information it needs. The DENA algorithm operates as follows:

1) Initially, each sensor node overhearing an event immediately calculates the intensity of the event as described in Section I. Furthermore, it schedules a notification message to inform all its neighbors about its election. The release time of the message is delayed according to the value of the derived intensity of the event, i.e. the higher the intensity the shorter the delay. We use the dynamic forwarding delay (DFD) mechanism proposed in [9]. A detailed description of the DFD concept we use is given below.
2) The sensor node closest to the event calculates the shortest delay. Consequently, it broadcasts the notification/IREQ message first. As it starts the notification message distribution it is implicitly signalized as election winner.
3) To gather the necessary information to perform the location estimation on the winner node, the notification message is combined with an information request message (IREQ) that has to be distributed within the two-hop neighborhood of the winner node. The reason for querying the two-hop neighborhood is given in Section V-B.
4) Each sensor node receiving the notification/IREQ message knows the existence of the winner node and immediately cancels its own election.
5) Each node that has received the notification message rebroadcasts it. Additionally, all nodes within two-hop neighborhood of the event generate an information respond message (IREP) to provide the winner with the position information it needs to calculate the event location. The IREP message may be piggy-backed on the notification messages of each two-hop neighbor to avoid additional transmissions.
6) The algorithm terminates when all election-losers have rebroadcast the notification message. All one-hop neighbor nodes of the winner perform their own location estimation as soon as they have overheard the piggy-backed IREP messages of their neighbors. Thereafter, they forward their results to the winner node which is responsible to calculate the final position estimation.

The intensity ω derived at each node in the reception area decreases with the distance d to the event. The general equation for this relation is $d \sim \sqrt[\alpha]{\frac{1}{\omega_{\min}}}$, where α is larger than one and $\omega_{\min}$ is the minimum intensity necessary to identify an event (see Section IV). The release time of the winner strongly depends on the amplitude of the event. Consequently, the weaker an event is, the slower it is detected by the DENA algorithm. Furthermore, it is crucial that each election-loser has to be notified before it determines itself as winner. This has to be taken into account for the design of the DFD function. The protocol proposed so far does not consider the case of collisions between transmissions. For this case we propose the usage of a backoff mechanism with an exponential time window after which the notification message is

rebroadcast if no notification message from another node was overheard in the meantime. We argue that collisions will not occur frequently, as the DFD is designed to avoid them. The simultaneous election of multiple winners is possible albeit not very probable. In this case, each winner node calculates its own position estimation and handles the result further, e.g. sends it to the base station. The algorithm does not yet consider any object tracking or the occurrence of simultaneous events. These are difficult topics and will be investigated in future work. Finally, an efficient broadcast protocol is used to minimize the number of retransmitting nodes. This protocol is again based on the DFD mechanism. The algorithm is discussed in the next section.

It must be mentioned that in our framework each sensor node knows its own location. This can be achieved by GPS, or by other location algorithms ([10], [11], [12]).

Dynamic Forwarding Delay

We use the dynamic forwarding delay (DFD) concept in two respects: Once to determine the release time of the notification/IREQ message, and once to perform an efficient broadcast in the reception area of an event. The DFD basically depends on the node position x and looks as follows:

$$DFD = MAX\_Delay \cdot f(x), \qquad f(x) \in [0,1]$$

The function f calculates a delay in dependence of the position of the message receiver. By the concept of the DFD, the decision to forward a packet is shifted from the sender to the receiver avoiding communication overhead to supply the sender with the information about its vicinity. This is in particular important in sensor networks, where battery constraint devices may follow sleep cycles to save battery power. In these networks gathering information about the neighborhood is expensive according to frequent topology changes or synchronization overhead. For this reason we think that a receiver based retransmission scheme is more appropriate. A key feature of the DFD mechanism used for the

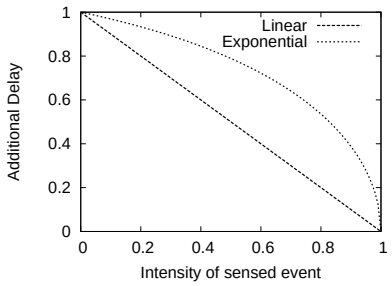

Fig. 2: DFD Functions

release time determination is the support of significant release time differences for nearby nodes. This is in particular true for the nodes close to the event. Thus, simultaneous winner election as well as collision probability are decreased. We propose an exponential DFD function as depicted in Fig. 2. For nodes with a higher significance, i.e. nodes that have derived a higher event intensity, the DFD timers are distributed over a larger interval. Thus, the probabilities of simultaneous winner election among these nodes as well as of collision occurrences are decreased. The exact specification of the DFD function for the notification/IREQ release has not yet been done. Its existence is however warranted and needs just the effort to find an adequate function and the according parameters. Thereby, the trade-off between efficiency and overhead caused by additional transmissions has to be minimized.

1) Dynamic Delayed Broadcast Protocol (DDB): Apart of using the DFD for the winner election, we use the DFD concept to perform an efficient broadcast. The algorithm operates in principle as follows: the DFD determines the time a node is delaying a broadcast message before rebroadcast it. While the expiration of this time, the node overhears the transmissions of other messages and cancels its rebroadcast if the retransmit threshold RT is under run, e.g. the distance to a sensor node that has already released the message falls below RT. The retransmission threshold RT may also be zero.

In [13] we have investigated different metrics for the rebroadcast decision. Thereby, the most efficient metric for the calculation of the DFD as well as for the decision whether to rebroadcast depends on the additional area a node may cover with its rebroadcast. Thereby, each sensor node calculates the additional area it covers as well as the DFD, whenever it overhears the transmission of the broadcast message. The node with the shortest DFD releases the packet first. The DDB protocol has been evaluated extensively and compared to well-known protocols. It was shown that the DDB protocol performs even better than a neighbor-based protocol in terms of energy consumption in most simulations. This is in particular true under frequent topology changes, what makes it useful for sensor networks where nodes often follow sleep cycles and the topology consequently changes frequently. Detailed information can be found in [13]. The main drawback of the DDB approach is however its computational complexity. To minimize this complexity, but nevertheless benefit from its advantages we redesigned the DFD metric. In the rest of this section we first shortly introduce the DDB concept with additional area coverage and then introduce our new metric which approximates the additional area approach, but uses much fewer energy.

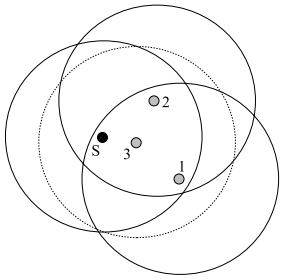

Fig. 3: Broadcasting with additional coverage

The basic functionality of broadcasting with additional area coverage is depicted in Fig. 3. The source node S starts

the communication by broadcasting its message. All three receivers calculate their DFD according to the additional area they cover with their retransmission. Node 1 is farthest away from S and accordingly calculates the largest additional area it would cover with its rebroadcast what leads to the shortest DFD. Nodes 2 and 3 overhear that retransmission and recalculate the additional area they newly cover and the respective DFD. Node 2 calculates the shorter DFD and rebroadcasts its message next. Node 3 overhears this message again and cancels its broadcast as it is totally covered by the transmissions of the other nodes. The DFD of the additional area coverage approach is calculated as follows:

$$DFD = MAX\_Delay \cdot \sqrt{\left(\frac{e - e^{\frac{AC}{AC_{MAX}}}}{e-1}\right)}$$

The additional area is denoted by AC and is always between zero and the maximal additional area AC_{MAX} a retransmitting node may cover. AC_{MAX} is achieved when a node is exactly placed on the border of the previous sender. In this case it covers an additional area of $\sim 61\%$. As mentioned above, the main drawbacks of this approach are its computational complexity and its memory demand. To reduce this overhead we redesigned the DDB by using a new metric. In Fig. 4 an approximation of calculating the additional area by the usage of the triangle connecting any three neighbor nodes is depicted.

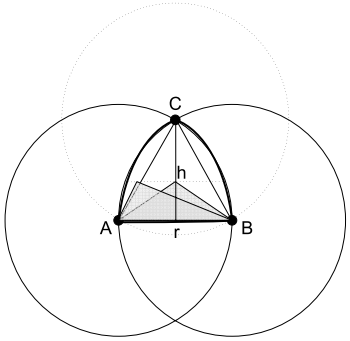

Fig. 4: Broadcasting with the triangle metric

Assuming that nodes A and B have already released their message, any node C that has overheard both transmissions lies in the intersection area of both transmissions. Moreover, the additional area a node covers is larger the farther away C from the connecting line $\overline{AB}$ is, largest at the position of node C in Fig. 4, i.e. the area A_{MAX} in this case is $\frac{\sqrt{3}}{4} \cdot r^2$. Consequently, we use the area of the triangle built by A, B and any node C in the intersection area as an indicator for the additional area a node C covers. Thereby, all nodes C with the same distance h to $\overline{AB}$ calculate the same triangular area, i.e. all nodes on the parallel line to $\overline{AB}$ with distance h. This adds a certain error, as the additional area a node covers depends on its location on the parallel line. The error is maximized at the center of line $\overline{AB}$. The area of the according triangle is zero, whereas the additional area a node covers is:

$$4\left[\int_0^{\frac{r}{4}} \sqrt{(r^2 - x^2)}\mathrm{d}x - \int_{-\frac{r}{2}}^{-\frac{r}{4}} \sqrt{(r^2 - x^2)}\mathrm{d}x\right] \sim 0.021 r^2\pi$$

We argue that a maximal deviation of about 2% is tolerable and should not affect the algorithm in a destructive way.

Once determined, the area $A_{Triangle}$ of the triangle is used to calculate the DFD:

$$DFD = MAX\_Delay \cdot \sqrt{\left(\frac{e - e^{\frac{A_{Triangle}}{A_{MAX}}}}{e-1}\right)}$$

The reason to use an exponential function is to favor nodes with a bigger triangle and to minimize the probability of collisions among these nodes. Obviously, the node with the shortest DFD broadcasts first. With this broadcast a triangle as mentioned above is virtually created. All nodes located within this triangle cancel their retransmission of the message (a similar approach to cancel the retransmission was used by [14]). This test is very simple and can be easily calculated by the use of barycentric coordinates. It adds however some errors. The problem is depicted in Fig. 5:

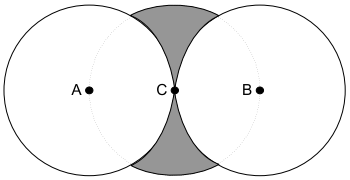

Fig. 5: The maximal loss of additional coverage

If the node overhearing two sending nodes is positioned exactly on the border of each node it must lie on the center of the connecting line between both nodes. In this case it cancels its retransmission, as it is within the triangle with height h = 0. The additional area it could cover is however $\sim 22\%$:

$$4\left[\int_0^{\frac{r}{2}} \sqrt{(r^2 - x^2)}\mathrm{d}x - \int_{\frac{r}{2}}^{r} \sqrt{(r^2 - x^2)}\mathrm{d}x\right] \sim 0.22 r^2\pi$$

The determination of the DFD as explained in this section is only applied if a node has already overheard at least two messages. In the case of the broadcast initiator the instant release of the message is obvious. If a node has overheard exactly one retransmission it calculates its DFD according to the mathematically exact additional area it covers. In this special case (intersection of two circles) the computation is simple.

IV. Intensity-based Localization Algorithm

Existing localization algorithms ([1], [8]) depend on the possibility to distinguish two kinds of signals transmitted by an event. Thereby, the distance of the event is derived from the time difference of arrival (TDOA) of two different signals. For example, [1] uses the time difference of arrival between the shock wave and the muzzle blast generated by a gun.

From the time difference of arrival of these two signals the distance between the sniper and the measuring sensor node is calculated. In many cases the dependence on two different kinds of signals is restrictive and not easy to fulfill. In contrast to these algorithms, the algorithm discussed in the next section depends only on the intensity derived by a sensor node. This algorithm is generically computable and does not depend on predefined hardware what supplies a good degree of freedom.

A condition to determine the position (e_x, e_y) of an event E is that on any sensor node in the significance area the intensity determining the amplitude of the occurred event can be derived. We assume that the intensity ω_X derived at a sensor node X is related to the distance d_X the sensor node is away from an event, e.g the farther away a sensor node is, the lower its derived intensity is. This relationship is formalized in the following relation:

$$\omega_X \sim \frac{1}{d_X^\alpha} \qquad , \alpha > 1 \tag{1}$$

The exponent α in (1) affects the degree of attenuation of the measured intensity in dependence of the distance to the source of the event. (1) is a generalized formula of the acoustic, radio, etc. path loss models. The attenuation of an acoustic signal is for example similar to $\frac{1}{d^2}$.

It is crucial that the intensity cannot be used as a direct substitute of the distance in order to estimate the position of an event, but the square root of the ratio of the intensities of two sensor nodes is equal to the ratio of the distances of the two sensor nodes to an event. This will be shown in this section. Furthermore, we will show that if a sensor node A knows its own intensity ω_A and position (a_x, a_y) as well as the positions and intensities of at least three not collinear neighbor nodes B, C, D it can calculate the position of the event. The distance d_A of a sensor node A from the location of an event e can be calculated with the theorem of Pythagoras:

$$d_A^2 = (a_x - e_x)^2 + (a_y - e_y)^2 \tag{2}$$

From (2) and (1) we can derive the general equation to get the ratio of the intensities of two sensor nodes A and B:

$$\frac{(a_x - e_x)^2 + (a_y - e_y)^2}{(b_x - e_x)^2 + (b_y - e_y)^2} = \left(\frac{\omega_B}{\omega_A}\right)^{\frac{2}{\alpha}} \tag{3}$$

(3) means that the ratio of the distances from two sensor nodes A and B to the event location is equal to the ratio of the intensities derived on both nodes. It forms a circle, unless the ratio is 1. This case will be discussed later. As the position of the event E is contained on all three circles and the intersection point of the three circles is uniquely determined, the location of the event E is equivalent to the intersection point. This is true at least as long as the intensities derived at the sensor nodes are correct (an example generated with Maple is depicted in Fig. 6).

In order to prove the applicability of (3), we have to show that the denominator cannot be zero. This is however trivial as from (3) we can conclude that the denominator can only become zero if $b_x = e_x$ and $b_y = e_y$. This means the

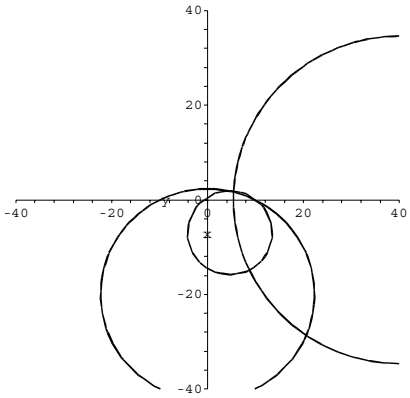

Fig. 6: Location of an event by intersection of three circles.

denominator can only be zero if the position of event E is exactly at the position of sensor node B. This case can be excluded, as the calculation of the position of an event is trivial if it occurs exactly at the location of a sensor node. Consequently, the position of an event is only calculated if it occurred not exactly at the location of one of the participating sensor nodes. In all these cases, the denominator cannot be zero.

Next, we calculate the intersection points of the circles that are derived from the ratios of the intensities of n non-collinear sensor nodes $S_1, .., S_n$, with $n > 1$. In order to facilitate the calculation, we set the point of origin of the coordinate system at the position of S_1 and the position of S_2 on the x-axis. This can be done without loss of generality. We will show that the calculation of the intersection point of the circles is equal to multilateration ([10], [12]). For better readability we replace ω_X^2 with ϕ_X. Using the ratios, we get the following equations:

$$\begin{aligned}
\frac{e_x^2 + e_y^2}{(s_{2_x} - e_x)^2 + e_y^2} &= \frac{\phi_{S_2}}{\phi_{S_1}} \\
\frac{e_x^2 + e_y^2}{(s_{3_x} - e_x)^2 + (s_{3_y} - e_y)^2} &= \frac{\phi_{S_3}}{\phi_{S_1}} \\
&\vdots \\
\frac{e_x^2 + e_y^2}{(s_{n_x} - e_x)^2 + (s_{n_y} - e_y)^2} &= \frac{\phi_{S_n}}{\phi_{S_1}}
\end{aligned}$$

If we dissolve the equations to zero and leave out the denominator from which we know that it cannot be zero , we get the following equations:

$$\begin{aligned}
\phi_{S_1}(e_x^2 + e_y^2) - \phi_{S_2}((S_{2x} - e_x)^2 + e_y^2) &= 0 \\
\phi_{S_1}(e_x^2 + e_y^2) - \phi_{S_3}((S_{3x} - e_x)^2 + (S_{3y} - e_y)^2) &= 0 \\
&\vdots \\
\phi_{S_1}(e_x^2 + e_y^2) - \phi_{S_n}((S_{nx} - e_x)^2 + (S_{ny} - e_y)^2) &= 0
\end{aligned}$$

The equations can be transformed to the following equations:

$$(\phi_{S_1} - \phi_{S_2})e_x^2 + (\phi_{S_1} - \phi_{S_2})e_y^2 + 2\phi_{S_2}S_{2x}e_x - \phi_{S_2}S_{2x}^2 = 0$$
$$(\phi_{S_1} - \phi_{S_3})e_x^2 + (\phi_{S_1} - \phi_{S_3})e_y^2 +$$
$$2\phi_{S_3}(S_{3x}e_x + S_{3y}e_y) - \phi_{S_3}(S_{3x}^2 + S_{3y}^2) = 0$$
$$\vdots$$
$$(\phi_{S_1} - \phi_{S_n})e_x^2 + (\phi_{S_1} - \phi_{S_n})e_y^2 +$$
$$2\phi_{S_n}(S_{nx}e_x + S_{ny}e_y) - \phi_{S_n}(S_{nx}^2 + S_{ny}^2) = 0$$

The system can be linearized by subtracting the first equation from the last $n-1$ equations. Therefore, the first equation has individually to be multiplied with $\frac{\phi_{S_1}-\phi_{S_3}}{\phi_{S_1}-\phi_{S_2}}$,..., $\frac{\phi_{S_1}-\phi_{S_n}}{\phi_{S_1}-\phi_{S_2}}$. The resulting equations are subtracted from equations $2, ..., n$. In all $n-1$ resulting equations the unknown variables e_x, e_y are on the left side of the equations:

$$2\phi_{S_3}(S_{3x}e_x + S_{3y}e_y) - \frac{2\phi_{S_2}S_{2x}e_x(\phi_{S_1} - \phi_{S_3})}{\phi_{S_1} - \phi_{S_2}}$$
$$= \phi_{S_3}(S_{3x}^2 + S_{3y}^2) - \frac{\phi_{S_2}S_{2x}^2(\phi_{S_1} - \phi_{S_3})}{\phi_A - \phi_{S_2}}$$
$$\vdots$$
$$2\phi_{S_n}(S_{nx}e_x + S_{ny}e_y) - \frac{2\phi_{S_2}S_{2x}e_x(\phi_{S_1} - \phi_{S_n})}{\phi_{S_1} - \phi_{S_2}}$$
$$= \phi_{S_n}(S_{nx}^2 + S_{ny}^2) - \frac{\phi_{S_2}S_{2x}^2(\phi_{S_1} - \phi_{S_n})}{\phi_{S_1} - \phi_{S_2}}$$

The equations above indicate that $\phi_{S_1} \neq \phi_{S_2}$. The case of equality of ϕ_{S_1} and ϕ_{S_2} is discussed in the next paragraph. For now, we assume that $\phi_{S_1} \neq \phi_{S_2}$ and therefore neglect the denominator as soon as all terms are of the same denominator. If all terms are reordered, we get a system of linear equations of the form $Ax = b$, where

$$A = \begin{bmatrix} 2(\phi_{S_3}S_{3x}\Gamma + \phi_{S_2}S_{2x}(\phi_{S_3} - \phi_{S_1})) & 2\phi_{S_3}S_{3y}\Gamma \\ \vdots & \\ 2(\phi_{S_n}S_{nx}\Gamma + \phi_{S_2}S_{2x}(\phi_{S_n} - \phi_{S_1})) & 2\phi_{S_n}S_{ny}\Gamma \end{bmatrix}$$

$$b = \begin{bmatrix} \phi_{S_3}(S_{3x}^2 + S_{3y}^2)\Gamma - \phi_{S_2}S_{2x}^2(\phi_{S_1} - \phi_{S_3}) \\ \vdots \\ \phi_{S_3}(S_{3x}^2 + S_{3y}^2)\Gamma - \phi_{S_2}S_{2x}^2(\phi_{S_1} - \phi_{S_3}) \end{bmatrix}$$

For better readability we substituted $(\phi_{S_1} - \phi_{S_2})$ with Γ in A, b respectively. This system can be solved using a standard least-square approach: $E = (A^T A)^{-1}A^T B$, where E is the location estimation of the event. When the inverse matrix cannot be calculated, the location cannot be computed and the multilateration fails. This can happen if $\phi_{S_1} = \phi_{S_2}$. This is however no restriction, as in the case of $\phi_{S_1} = \phi_{S_2}$ the ratio of the intensities is 1 and the position of E lies on the vertical line through the middle of $\overline{S_1, S_2}$. The intersection of this vertical line with any of the participating circles results in the possible locations of event E. Consequently, in the case of $\phi_{S_1} = \phi_{S_2}$ the matrix is not calculated and the location is estimated using the intersection of the vertical line with any two independent circles derived from the intensities.

V. Simulations

In this section we present first simulation results of the broadcast as well as of the localization algorithm. To evaluate the two algorithms we implemented both in Matlab. All simulations were run over 20 seeds with a 95% confidence interval. The simulations described in this section share a common scenario. This standard scenario consists of 300 sensor nodes randomly distributed in a square area with sides of 100 meters. The radio range R of each sensor node is ten meters, which results in an average connectivity of nine neighbor nodes.

A. Evaluation of the DDB protocol

To evaluate the DDB broadcast protocol, we implemented the triangle metric along with the additional area metric and a simple flooding algorithm in Matlab. A detailed comparison of the DDB protocol to other broadcast protocols can be found in [13]. In this paper we will only investigate the performance of our new metric compared to the additional area coverage metric and a simple flooding protocol. It is to remark that the results we gained match well the results presented in [13] simulated with the Qualnet network simulator [15]. The thresholds for the protocols were chosen as follows: The additional area coverage (DDB_{AC}) metric uses a retransmission threshold RT of 22% of AC_{MAX}. This means a node using the DDB_{AC} metric cancels its retransmission when the additional area it covers with a rebroadcast is below 22% of AC_{MAX}. A value of 22% is chosen as the maximal loss of the DDB triangle ($DDB_{Triangle}$) approach is intrinsic 22%.

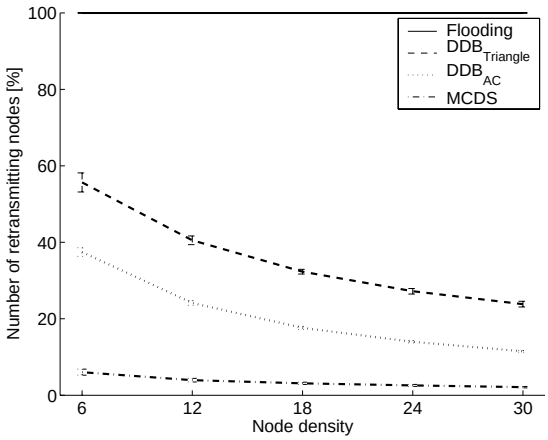

Fig. 7: Percentage of retransmitting nodes with different DDB metrics

To evaluate the performance of the different broadcast metrics, the network density is varied. The number of nodes in the network are altered from 300 up to 1100 in steps of 200 nodes resulting in an average number of neighbors from 9 to 33. The broadcast was always initiated by sensor node one. To have a benchmark for our algorithms we additionally implemented a minimum connected dominating set (MCDS) algorithm which computes the minimal set of nodes that is

necessary to reach all nodes within the network. The MCDS algorithm calculates the optimal solution for the broadcast problem, but is only computable with the knowledge about the whole network topology. The results of the DDB simulations are depicted in Fig. 7. The results of the delivery ratios of the different protocols are not depicted, as the delivery ratios are 100% in all simulations over all seeds. The DDB_{AC} as well as the $DDB_{Triangle}$ protocol decreased the number of retransmitting nodes in the network considerably, especially when the network density is high. Surprisingly DDB_{AC} needed almost 20% less retransmitting nodes than $DDB_{Triangle}$ over all node densities, resulting in less than 25% of retransmitting nodes as soon as the number of neighbors approximates 15 neighbors. The performance of the MCDS algorithm is by far the best. The difference can be explained by the retransmitting overhead of DDB for nodes close to the network area border. As the DDB algorithm is receiver based it has no knowledge about a possible area border and all nodes close to the border retransmit their message even as there are no additional nodes reachable.

We conclude that the DDB_{AC} protocol is the most suited protocol as long as the resource constraints on the nodes are not too restrictive. In other cases the $DDB_{Triangle}$ approach seems to be a reasonable alternative. A main advantage of the DDB protocol architecture is the absence of any states. The rebroadcast decision solely depends on the position of a node and of a function to assess that position. The results gained in these initial simulations encourage us to use the DDB protocol for the notification message distribution.

B. Evaluation of the ILA algorithm

In order to simulate noisy measurements on the sensor boards we implemented the inverse square law with a signal attenuation of $\frac{1}{d^2}$, where d is the distance, as error model. This model is for example appropriate for sound propagation. Intensity errors that are caused by noisy sensor readings are modeled according to the following formula:

$$err(\omega) = 1 \pm \lambda N(0,1) \cdot \omega \tag{4}$$

The error err depends on the derived intensity ω as well as the square distance d^2 between the event source and the measuring sensor node. Its amplitude is adjusted via the parameter λ. $N(0,1)$ is a normal distribution with mean zero and standard deviation one. According to (4) errors are normally distributed around the intensity whereas the amplitude of the deviation depends on the square distance a sensor node is away from an event and λ.

The simulations in this section share the common scenario parameters proposed above. Furthermore, a number of parameters are varied: number of sensor nodes, standard deviation of the measurement error, and reception radius. The event is always localized at position (50, 50). The reception radius D determines the distance until which the event is observable. In our simulations the reception radius varies from 10 m to 40 m. The amplitude of the standard deviation is adjusted over λ and its value varies from 0 to 25%. All simulations have in common that location estimations that are farther away from the calculating sensor node than the reception radius are discharged. This restriction is reasonable, as the event could not have been sensed by a sensor board if it is farther away than the reception radius. Very erroneous location estimations are seldom, but possible as a normal distribution is used in the error model, which permits very high deviations.

1) Influence of the distance on the accuracy: In these initial simulations we investigate the influence of the reception radius D. The location estimation is thereby performed on any sensor board in the reception area bounded by the reception radius and the location estimation error is averaged over all computations. We varied the reception radius accordingly to the values defined in the last section. In Fig. 8 the results of these simulations are depicted. The location estimation error is in all subsequent figures denoted in percentage of the radio range R.

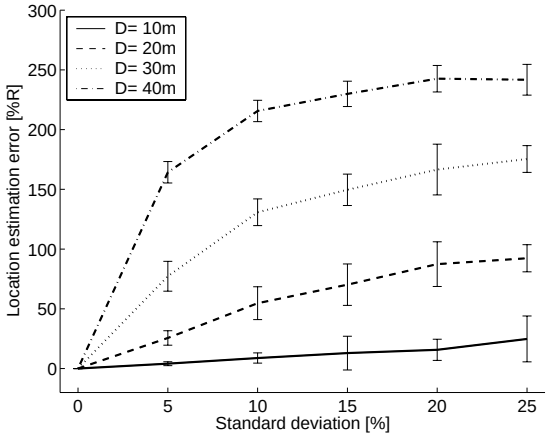

Fig. 8: Location error with variable reception radius D

The gained results indicate that the position estimation error is acceptable as long as the distance to the event is not too far and the sensor readings are not too noisy, i.e. λ is not too high. This result is not surprising and enforces our choice to elect the closest sensor node as winner and perform the location estimation on it. In the next simulation section we will explore the ILA performance if only the winner node performs the position estimation.

2) Position estimation at winner node: In this simulations we vary the number of nodes in the network to investigate the influence of the network density on the location estimation accuracy. 300, 500, and 1000 sensor nodes are simulated what results in an average connectivity of 9, 15, and 31 sensor nodes. We expect a better performance in denser networks, as the average distance between winner node and event source becomes smaller.

The results of the simulations are depicted in Fig. 9. The location estimation error is in all simulations between one meter and five meters. Thereby, the majority of the calculations supply location estimations less than two meters away from the exact event position. The large confidence intervals in these

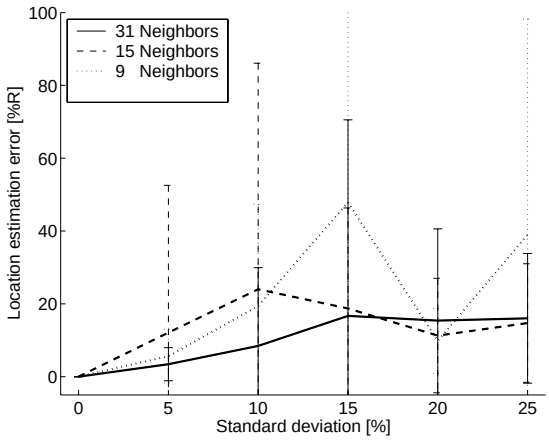

Fig. 9: Location estimation of winner

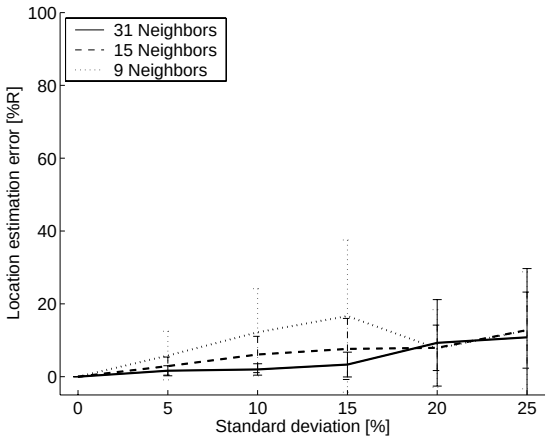

Fig. 10: Location estimation of winner including its vicinity

simulations indicate that the calculation limitation on only one sensor node is quite error-prone. This is substantiated by the sensibility of the ILA algorithm. The location estimation works well as long as there are no outliers among the measurements. On the other hand, if some measurements are very noisy an accurate event location estimation with only one sensor node fails, as the least square method used cannot handle very noisy sensor readings. We therefore will take the position estimations of the one-hop neighbor nodes of the winner also into consideration. The according refinements and simulations are discussed in the next subsection. Another possibility was to investigate other, less error-prone approaches to compute the location estimation. This remains to be done in future work.

3) Position estimation enhanced with information from winner vicinity: The results gained in the last section have shown that the location estimation on only one sensor board is in general not accurate enough. Consequently, we enhance the computation on the winner node with the position estimations calculated in its immediate vicinity. The necessary information is provided by the DEA algorithm proposed in Section III. The computation instruction is as follows: The winner node calculates the mean value and the standard deviation of its own position estimation as well as of the estimations of all of its neighbor nodes. It disregards then all estimations that are more than standard deviation away from the mean and computes the location with the remaining estimations. Furthermore, the location estimation fails if the standard deviation is more than half the mean value. This could happen in a noisy environment where a reliable location estimation is no longer given. Enhanced algorithms to operate in such scenarios will be investigated in future work.

The simulation results (see Fig. 10) show that the mean error as well as the standard deviation are considerably diminished with the algorithm proposed in this section. The node density influences the location estimation positively, but even with a rather low node connectivity of in average 9 neighbor nodes, we get feasible results. We conclude our evaluation with these first results, which indicate that a intensity-based localization on the winner node is possible. It moreover performed well in the simulations done so far. The results encourage us to continue our work and finally provide an efficient, distributed and accurate event detection and localization mechanism.

VI. Conclusions

We introduced two algorithms in this paper, namely the distributed election-winner notification algorithm (DENA) and the intensity-based localization algorithm (ILA). The combination of both algorithms builds a framework to efficiently and accurately detect and localize events. The dependence of the ILA algorithm on merely the derived intensity of an event on a sensor node constitutes its generic applicability as well as its weak binding on sensor hardware. In this paper we have shown that the performance of the ILA algorithm verified our expectations. It was shown that the location of an event can be computed in a distributed manner without need to gather any information on a sink node and that the accuracy of the event location improves the closer to the event the ILA algorithm is performed. Concerning the DENA algorithm, we have shown that the notification message is distributed efficiently by the dynamic broadcast protocol (DDB). The number of retransmitting nodes is decreased considerably with both DDB metrics tested so far. It remains to mention that both, the ILA and the DENA algorithm work close together in our approach. This is reasonable, it does however not restrict the applicability of the key functionality of both algorithms on their own.

VII. Future work

In future work we will further evaluate both algorithms. Special interest will be focused on the additional delay the DENA causes and on its ability to minimize the number of retransmitting nodes. At time, the DENA algorithm causes every node in the two-hop neighborhood of the winner node to respond. The possibility to query only a subset of these nodes will be considered. Furthermore, an implementation of the framework using the OMNeT++ [16] network simulator has been started. We will compare our framework to other

event detection and localization schemes. Thereby, we will focus on energy and bandwidth consumption. At time, the ILA algorithm needs the information of all neighboring nodes. In future work we will investigate if a subset of these neighbor nodes results in sufficiently accurate results. In that context, we will also investigate appropriate techniques to filter outliers. To deal with erroneous sensor measurements we currently use a linear mean-square approach. We will consider the usage of more sophisticated non-linear least square methods [17]. Finally, we will perform extensive sensor measurements on real hardware to obtain sophisticated error models and we will check if the algorithm is also feasible when applied on moving events and with multiple event sources.

REFERENCES

[1] G. Simon, G. Balogh, G. Bap, M. Maróti, B. Kusy, J. Sallai, A. Lédeczi, A. Nádas, and K. Frampton, "Sensor network-based countersniper system," in *SenSys*, Baltimore, Maryland, USA, November 2004.

[2] Y. Zou and K. Chakrabarty, "Sensor deployment and target localization in distributed sensor networks," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 3, no. 1, pp. 61–91, 2004.

[3] S. Guha, R. N. Murty, and E. G. Sirer, "Sextant: A unified node and event localization framework using non-convex constraints," in *MobiHoc'05*, Urbana-Champaign, Illinois, USA, May 2005, pp. 205–216.

[4] M. Kumar, L. Schwiebert, and M. Brockmeyer, "Efficient data aggregation middleware for wireless sensor networks," in *IEEE International Conference on Mobile Ad-hoc and Sensor Systems*, Fort Lauderdale, Florida, USA, October 25-27 2004, pp. 1579–1581.

[5] S. Li, S. H. Son, and J. A. Stankovic, "Event detection services using data service middleware in distributed sensor networks," in *ISPN'03*, Palo Alto, USA, April 2003, pp. 502–517.

[6] T. Abdelzaher, B. Blum, D. Evans, J. George, S. George, L. Gu, T. He, C. Huang, P. Nagaraddi, S. Son, P. Sorokin, J. Stankovic, and A. Wood, "Envirotrack: Towards an environmental computing paradigm for distributed sensor networks," in *Proc. of 24th International Conference on Distributed Computing Systems (ICDCS)*, Tokyo, Japan, Mar. 2004, to appear.

[7] N. Niculescu and B.Nath, "Ad hoc positioning system using aoa," in *Proceedings of IEEE INFOCOM Conference on Computer Communications*, San Francisco, CA, USA, 2003.

[8] S. Capkun, M. Hamdi, and J. P. Hubaux, "Gps-free positioning in mobile ad hoc networks," in *Proceedings of HICSS*, January 2001, pp. 3481–3490.

[9] M. Heissenbüttel, T. Braun, T. Bernoulli, and M. Wälchli, "BLR: Beacon-less routing algorithm for mobile ad-hoc networks," *Computer Communications Journal*, vol. 27, no. 11, pp. 1076–1086, July 2004.

[10] K. Langendoen and N. Reijers, "Distributed localization in wireless sensor networks: a quantitive comparison," *Computer Networks*, vol. 43, no. 4, pp. 499–518, 2003.

[11] D. Niculescu, "Positioning in ad hoc sensor networks," *IEEE Communications Magazine*, vol. 18, no. 4, pp. 24–29, July/August 2004.

[12] A. Savvides, H. Park, and M. B. Srivastava, "The n-hop multilateration primitive for node localization problems," *Mobile Networks and Applications*, vol. 8, pp. 443–451, 2003.

[13] M. Heissenbüttel, T. Braun, M. Wälchli, and T. Bernoulli, "Optimized stateless broadcasting in wireless multi-hop networks," in *IEEE Infocom 2006*, Barcelona, April 23-29 2006, to appear.

[14] S. Ni, Y. Tseng, Y. Chen, and J. Sheu, "The broadcast storm problem in a mobile ad hoc network," in *Proceedings of the ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM)*, 1999, pp. 151–162.

[15] Qualnet, "Scalable network technologies (snt)." [Online]. Available: http://www.qualnet.com

[16] A. Varga, "Omnet++ discrete event simulation system." [Online]. Available: http://www.omnetpp.org/

[17] W. Navidi, W. S. Murphy, and W. Hereman, "Statistical methods in surveying by trilateration," *Computational Statistics and Data Analysis*,, vol. 27, pp. 209–227, 1998.