# Applied Sensor-Assisted Monte Carlo Localization for Mobile Wireless Sensor Networks

Salke Hartung Arne Bochem Andreas Zdziarstek Dieter Hogrefe

Institute of Computer Science, Telematics Group
University of Goettingen, Germany
{hartung, bochem, hogrefe}@cs.uni-goettingen.de

## Abstract

Localization is a mandatory requirement in almost all sensor network applications. In order to avoid power-hungry GNSS solutions like GPS, alternative localization algorithms based on anchor nodes are developed. Unfortunately, many proposed solutions focus on static networks and do not account for mobility. An often cited and further improved approach is Monte Carlo Localization (MCL), which is one of the first methods accounting for total mobility in the network. In this paper we propose and practically evaluate Sensor-Assisted Monte Carlo Localization (SA-MCL), which is designed to bypass situations of temporary connection loss to anchor nodes due to changing network topologies. SA-MCL uses additional sensor information to update the position estimation of a node in case no anchor nodes are available. We show by both practical field tests and simulations that our dead reckoning approach can successfully account for situations without anchor node information and reduces the localization error by about 58 %. Our practical implementation is performed on IRIS sensor motes which are mounted to the backside of radio controlled cars.

## Categories and Subject Descriptors

C.2 [**Computer Systems Organization**]: COMPUTER-COMMUNICATION NETWORKS; C.2.1 [**COMPUTER-COMMUNICATION NETWORKS**]: Network Architecture and Design—*Distributed networks, Wireless communication*

## General Terms

ALGORITHMS, EXPERIMENTATION, MEASUREMENT, PERFORMANCE

*Keywords*

Wireless Sensor Network, Mobility, Localization, Range-Free, Monte Carlo Localization

International Conference on Embedded Wireless Systems and Networks (EWSN) 2016
15–17 February, Graz, Austria

ISBN: 978-0-9949886-0-7

## 1 Introduction

Data collected in Wireless Sensor Networks (WSNs) will often be rendered useless if it is not associated to the physical location in space of the corresponding sensor mote. In tracking applications like wildlife monitoring the position itself might be the data of interest valuable to researchers. Although global navigation satellite systems (GNSSs) like GPS solve the localization problem for many systems like portable computers, digital cameras and mobile phones, they are not suitable for the application in WSNs. Sensor motes are designed to be extremely energy-efficient and tend to reach very small manufacturing sizes. GNSSs however are very power-hungry components, require an additional antenna and increase the production and deployment costs of WSNs. Furthermore, GNSSs are known to work only in outdoor scenarios with clear line of sight to the navigation satellites. Sensor network applications are manifold and therefore it cannot be guaranteed that the requirements for running GNSSs can be fulfilled. To avoid the usage of GNSSs the concept of anchor nodes has been developed. These special nodes are the only ones in the network which are always aware of their position either by using a GNSS or by having a static position which can be determined during deployment. Anchor nodes are sending out location announcements which basically include a unique id and the positioning information of the anchor node. Simple nodes receiving these announcements use a localization algorithm to estimate their position based on the information of the anchor nodes. These estimations are always subject to a certain localization error $\varepsilon_{\mathtt{loc}}$. The aim of all localization algorithms is to keep $\varepsilon_{\mathtt{loc}}$ as low as possible without introducing big additional hardware or computational requirements.

Localization algorithms can be categorized into two major groups [3, 11]. Range-based approaches always require active participation of the simple nodes in terms of determining distances to anchor nodes or angles from incoming radio signals [4, 2]. A very common technique to estimate the distance to an anchor node is using the Received Signal Strength Indicator (RSSI). Based on the assumption that the power of the received signal is proportionally decreasing with increasing distance to the sender it is theoretically possible to estimate the distance to the sender by using a suitable propagation model. However, the RSSI is subject to lots of different impacts and heavily depends on the physical circumstances in the network including antennas, weather con-

ditions, obstacles which might block the signal, etc. Previous studies of the RSSI indicate that this technique is too unsteady to achieve reasonable results for usage in localization techniques [12, 23, 1, 19]. The second group are range-free algorithms. Range-free localization is usually based on connectivity only and does not require any active measurements conducted by the simple nodes. Consequently, range-free localization is often much easier to implement and has lower deployment costs.

A popular representative of range-free localization is Monte Carlo Localization (MCL) [13]. MCL was originally developed for the usage in robotics and has been adapted by Hu and Evans for Wireless Sensor Networks in 2004. The key idea of MCL is to represent the probability distribution of the position of the sensor mote as a set of weighted samples. Each sample represents a possible location of the mote. Bayesian filtering is used to eliminate impossible samples, i.e. samples which are not in communication range to anchor nodes. The average of all samples passing the filter is the localization estimation of the node. The most important contribution of MCL is that it is especially designed for mobile WSNs, i.e. all nodes including anchors are allowed to move arbitrarily during network operation time.

A common problem of all localization techniques based on anchors information in mobile WSNs is the steadily changing network topology, which might lead to complete connection loss to all anchor nodes. In this case no location estimation is possible. In this work we are extending MCL using a dead reckoning technique to bypass these situations using additional sensor information from a gyroscope, a magnetometer and an accelerometer. In contrast to many other works we implement our approach called Sensor-Assisted Monte Carlo Localization (SA-MCL) on real hardware and evaluate it in a mobile sensor network testbed consisting of radio controlled cars. Furthermore, to analyze important metrics like scalability we provide excessive simulation studies and compare all results to the original MCL algorithm. In summary, our contributions are as follows:

1. We propose SA-MCL, an approach on top of MCL to bypass situations without anchor information
2. We provide an excessive comparison study of SA-MCL and MCL based on network simulation
3. We introduce our mobile wireless sensor network testbed.
4. We implement and evaluate MCL and SA-MCL on real hardware.

The rest of the paper is structured as follows. After introducing necessary terms and revising MCL in Section 2, we review related work in Section 3. In Section 4 we explain our extension SA-MCL. Section 5 gives a detailed report of our simulation studies. Our mobile WSN evaluation of SA-MCL is provided in Section 6. Finally, we conclude our work in Section 7.

# 2 System Model and Review of MCL

In this section we describe the system model used in this paper and introduce necessary terms. Further, we review the original MCL algorithm to provide a basic understanding for the reader.

## 2.1 Terms

- Localization: The term localization refers to the process of determining the own position in space. Position and location are equally used in this paper.
- Seed nodes: A seed node is a node in the network capable of determining its position on its own, e.g. by using GPS, and is distributing this information at regular intervals as a broadcast message.
- Simple nodes: All other nodes in the network which are trying to localize themselves are called simple nodes. They are the majority of the network and do not have the capability to determine their location without additional information.
- Location announcement: A broadcasted message by a seed node including its own location, a timestamp and an identifier to be able to distinguish between multiple received location announcements.

## 2.2 Known Parameters and System Model

This paper assumes all nodes are homogenous, i.e. they share the same hardware capabilities including antennas, processors, memory, etc. Seed nodes only differ by being equipped with GPS and otherwise share the same parameter set. The only parameters known by all nodes in the system are $v_{\max}$ and $r$.

- $v_{\max}$
  The maximum velocity a node can have. A node does not necessarily have to move with $v_{\max}$ all the time. It is just an upper bound.
- $r$
  The maximum communication range. Every node $i$ can assume that the distance to another node $j$ is $\leq r$ if $i$ can receive messages from $j$.

All nodes are assumed to move arbitrary in a deployment area of limited dimensions. The boundaries of the deployment area are insuperable. In general the mobility in our system can be described using the random waypoint model: all nodes will move to an arbitrary destination and rest for a maximum time of $t_{\text{wait}}$ before choosing a new waypoint and starting over.

## 2.3 Monte Carlo Localization

The MCL algorithm has been adapted from the area of robotics [10] and presented for the usage in WSNs by Hu and Evans [13]. The core idea of MCL is to represent the probability distribution of a nodes' location as a set of weighted samples, $L$, where each sample $l_t$ represents a possible location of the node at time $t$. Impossible samples are filtered based on made observations, i.e. the received location announcements.

The initial set, $L_0$, is selected by choosing random locations in the whole deployment area. A node will always

1: **procedure** MCL
2: $L_t = \{\}$
3: **while** $size(L_t) < N_{\text{sample}}$ **do**
4: $R = \{l_t^i | l_t^i$ from $p(l_t|l_{t-1}^i), l_{t-1}^i \in L_{t-1}\}$
5: $\forall\, i, 1 \leq i \leq N$
6:
7: $R_{\text{filtered}} = \{l_t^i | l_t^i$ where $l_t^i \in R \wedge p(o_t|l_t^i) > 0\}$
8: $L_t = \text{choose}(L_t \cup R_{\text{filtered}}, N)$
9: **end while**
10: **end procedure**

Figure 1: Original MCL algorithm [13].

maintain a fixed number of samples, $N_{\text{sample}}$, to guarantee enough variability while still limiting the computational overhead. The MCL algorithm shown in Figure 1 computes the sample set $L_t$ using the information of sample set $L_{t-1}$ and observations of seed nodes, $o_t$, available at time $t$. To account for uncertainty about the node movement behavior the algorithm includes a *prediction step* (line 4) in which a new sample is drawn from a circular sampling area with radius $r_{\text{sarea}} = v_{\max} \times t_{\text{check}}$ around its current position given by a transition equation $p(l_t|l_{t-1})$. As explained above, in MCL $v_{\max}$ is the maximum velocity of a node. $t_{\text{check}}$ is the time between two localization attempts of a node. The probability of the current location given the previous location estimation is given by a uniform distribution [13], where $d(..)$ denotes the Euclidean distance between two samples as shown in Equation(1).

$$p(l_t|l_{t-1}) = \begin{cases} \frac{1}{\pi \times r_{\text{sarea}}{}^2}, & \text{if } d(l_t, l_{t-1}) \leq r_{\text{sarea}} \\ 0, & \text{if } d(l_t, l_{t-1}) > r_{\text{sarea}} \end{cases} \quad (1)$$

The set generated in the prediction step is put into the *filtering step* (line 7) which uses the observations $o_t$ to filter impossible node locations from the sample set. Each node keeps track of its first-hop neighbor seeds $S$ and of its second-hop neighbor seeds $T$. The filtering condition for a sample $l$ is given in Equation (2).

$$\text{filter}(l) = \forall s \in S, d(l,s) \leq r \wedge \forall s \in T, r < d(l,s) \leq 2r \quad (2)$$

Samples passing the filter are assigned a weight of 1, all others a weight of 0. Samples with a weight of 0 are ignored in further re-sampling steps. If more than $N_{\text{sample}}$ samples have been generated, $N_{\text{sample}}$ random samples are chosen from the current set (line 8). The process is repeated until $|L_t| >= N_{\text{sample}}$ (line 3). The final step is to compute the position estimation by calculating the weighted average of the sample set, i.e. the average of all samples with a weight of 1. For a more detailed description of MCL we would like to refer the reader to the original paper [13].

## 3 Related Work

Existing work on improvements of MCL mainly focuses on enhancing the filtering step either by introducing more sophisticated sample weighting or by defining more precise filtering conditions.

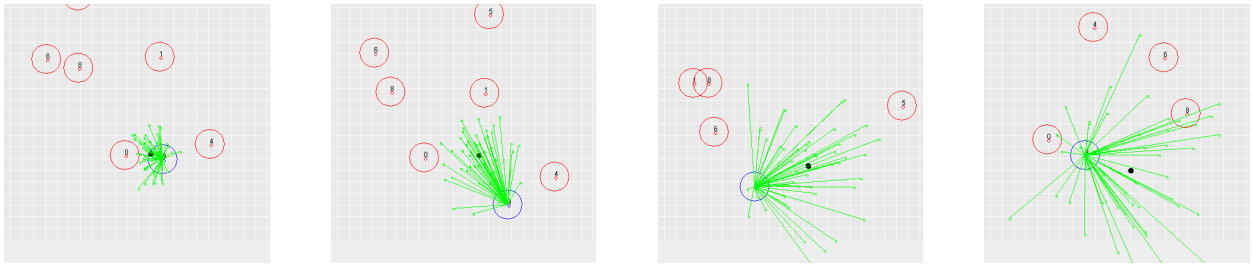
Rudafshani and Datta [20] take neighboring information into consideration for calculating sample weights. This means not only seed nodes contribute to the localization process of a node, but all neighbors of a node do. The authors present two versions of their algorithm: MSL* calculates weights for all of a neighbor's samples, while MSL tries to reduce the therefore emerging computational and communication overhead by calculating a single weight for every neighbor. Hence, in this work a closeness value is introduced which describes the quality of a location estimate. The closeness value is used to identify neighbors which seem to have a good estimation of their own position and therefore provide more valuable information to the node which tries to localize.

Zhang et al. propose WMCL [24] (weighted MCL) which provides further improvements for the MCL algorithm family. A bounding box is constructed to reduce the area from which new candidate samples are drawn. According to the authors this results in less overhead when the new sample set is computed. In addition to that, WMCL makes use of neighborhood position estimations to increase the localization efficiency. The authors state that their algorithm works better in static scenarios than previous solutions without tuning special parameters as done in MSL/MSL*. A real implementation on MICAz motes [7] is provided in a small testbed to evaluate a static scenario.

Teng et al. [22] show how a single mobile seed node can be used to localize all nodes in a static sensor network in their approach called MA-MCL. Applications for these scenarios might be rare and can only be expected when sensor nodes are deployed in a rush so that there is no time to calibrate the position. The sensors are not expected to move after deployment. The seed node is the only node equipped with GPS and will move randomly in the whole deployment area. Since the unknown nodes do not move, they generate new samples based on their current position only in a static square area with side length $\beta$ which is a tunable parameter of the algorithm and is by default set to $\beta = 0.1r$ where $r$ is the communication range of the simple node. The authors compare their work to the solution given in [20] and show that in these special scenarios their algorithm outperforms MSL and MSL*. However, they created a single point of failure situation: if the mobile seed fails for any reasons, the whole network will not be able to localize.

All of the mentioned extensions for MCL are working well, assuming that seed information is always present, i.e. seed nodes are always in communication range. Our approach differs from all of the above, since we are accounting for temporary situations in which no seed nodes are present at all. Furthermore, except for WMCL none of the approaches was implemented on real hardware not to mention a proper evaluation in a mobile WSN testbed.

Aside MCL, other range-free solutions have been proposed. Zhong and He introduce the regulated signature distance (RSD) [25] which acts as an additional metric for connectivity based approaches. The authors follow the assumption that RSSI is not a reliable estimator for determining positions on its own, but can be used as an additional indicator to improve connectivity based solutions.



(a) t=5 (b) t=10 (c) t=15 (d) t=20

Figure 2: Sample set degeneration of MCL.

Klingbeil and Wark proposed using a combination of sensor data originating from a IMU and a set of static seed nodes to form a network able to monitor motions and paths of walking people in an indoor scenario [16]. Their approach of detecting movement and heading is similar to the one used in this paper, but the localization algorithm is executed on a central personal computer. Therefore, this solution is not suited for spontaneously formed networks where all nodes are mobile.

Evaluations of algorithms for WSNs performed in mobile test beds are rare in general. Kafle et al. recently proposed a dynamic mobile sensor networking platform [15]. Their work can be understood as a framework for all kinds of sensor network applications, which might encourage researchers to put more focus on real implementations.

# 4 Sensor-Assisted MCL

In this section we first analyze the behavior of MCL in situations with no seed information present and propose our dead reckoning improvement named Sensor-Assisted Monte Carlo Localization (SA-MCL) afterwards.

## 4.1 Problem Statement

Mobility in WSNs continuously generates new network topologies. The network might be divided into different parts and single nodes could be isolated completely. Depending on the number of seed nodes in the network the complete coverage of the deployment area cannot be guaranteed. Eventually, simple nodes are confronted with situations without seed information. In this case a simple node can only execute the prediction step of MCL. Consequently, for longer periods of seed node absence the MCL sample set will degenerate, i.e. the samples are spread over the whole deployment area as illustrated in Figure 2. The green bars indicate the distance from the real position of the node to the samples of its MCL sample set. Due to repeated MCL prediction without MCL filtering the samples are spread more and more in the deployment area over time. As a consequence, the position estimation indicated by the black dot is getting worse and worse.

In our work we are focusing on bypassing these situations by using additional sensor information. In the imminent event of losing contact to all seed nodes a combination of sensor information from an accelerometer, a magnetometer and a gyroscope is used to update the nodes' position based on the last location estimation.

## 4.2 Design

The core idea of SA-MCL is to record the path of a node relative to the last known position if no seed information is available. Therefore, a simple node in SA-MCL has to be equipped with additional sensors:

1. A **magnetometer** is a simple compass module able to measure the strength of the magnetic field of the Earth. Consequently, it can be used to determine the heading of an object in degrees where magnetic North is fixed at 0°.
2. An **accelerometer** is a device able to measure proper acceleration, i.e. the acceleration relative to free fall. Accelerometers usually provide 3-axis-measurement and therefore are suitable for usage in 3D space navigation. By integrating the accelerometer measurements the velocity of a moving object can be determined [21, 6].
3. A **gyroscope** measures angular velocity along up to three axis. Given a certain orientation of a node, the gyrometer can be used to calculate the relative difference in orientation from the last known point.

The three introduced sensors are often combined on a single chip and are very cheap as they are mass production components which are also used in smart phones or laptops.

By combining the sensor information a mote can roughly record the path it is traveling. If it needs to localize again and did not receive any location announcements, it will freeze the state of the MCL sample set by moving all samples according to the recorded path. The procedure is illustrated in Figure 3. At time $t_0$ the node loses contact to all seed nodes and needs to localize again at time $t_4$. It is able to estimate its movement according to the gathered sensor data at time $t_1$ and $t_2$ and therefore will move all samples relatively to the last known position at time $t_4$.

The pseudo code of SA-MCL is shown in Figure 4. The main difference to MCL is that SA-MCL is keeping track of the number of observations $|o_t|$. If no new observations are obtained, the sensor data will be gathered instead ($\Delta x$ and $\Delta y$) and the traveled distance will be added to all samples in $L_{t-1}$ to retrieve $L_t$. Otherwise, the default MCL algorithm is executed. Since using additional sensors consume energy, they are only activated if necessary. This will be the case if the number of observations tends to reach 0. Once the

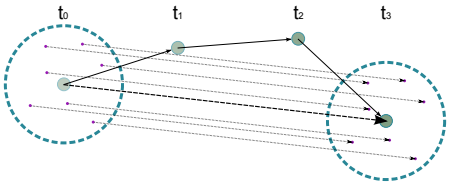



Figure 3: Samples moved accordingly to the recorded path.

node receives new location announcements the sensors can be turned off again. The important part, which is simplified in the pseudo code for the ease of presentation, is the implementation of the functions `getMovementXfromSensors` and `getMovementYfromSensors`. Later sections describe the process of gathering the sensor information in more detail.

1: **procedure** SA-MCL
2: $L_t \leftarrow \{\}$
3: **if** $|o_t| < 1 \wedge sensorsActive$ **then**
4: $\Delta x \leftarrow$ getMovementXfromSensors()
5: $\Delta y \leftarrow$ getMovementYfromSensors()
6: $\forall\, l_t \in L_t$ do
7: $l_t.x \leftarrow l_t.x + \Delta x$
8: $l_t.y \leftarrow l_t.y + \Delta y$
9: **else**
10: **while** $size(L_t) < N$ **do**
11: $R = \{l_t^i | l_t^i$ from $p(l_t|l_{t-1}^i), l_{t-1}^i \in L_{t-1}\}$
12: $\forall\, i, where 1 \leq i \leq N$
13:
14: $R_{\text{filtered}} = \{l_t^i | l_t^i$ where $l_t^i \in R \wedge p(o_t|l_t^i) > 0\}$
15: $L_t = choose(L_t \cup R_{\text{filtered}}, N)$
16: **end while**
17: **if** $|o_t| < 2$ **then**
18: $sensorsActive \leftarrow true$
19: **else**
20: $sensorsActive \leftarrow false$
21: **end if**
22: **end if**
23: **end procedure**

Figure 4: SA-MCL algorithm.

## 5 Simulation Evaluation

Certain important metrics of WSN algorithms are difficult to evaluate on real hardware without access to an already deployed and operating network. For instance, the scalability of an algorithm can only be tested with hundreds or even thousand of nodes which usually goes beyond the scope of feasibility. In these cases network simulation provides an easy solution to do an initial evaluation which can be extended by real experiments.

Our simulation evaluation of SA-MCL is performed in the original Java simulator provided by Hu and Evans [13]. Although the simulator lacks several basic features like path fading models or packet collision handling, using the same evaluation tool leads to optimal comparability. Implementing our approach in network simulation software is very simple compared to working with real sensor information from accelerometer, gyroscope and magnetometer. In the simulator, the current position and the next waypoint of a node as well as its current velocity are always known, i.e. it is very easy to calculate the orientation of a node and the distance it traveled. It would be unrealistic to assume that in a real implementation the additional sensors would be able to deliver perfect data. Therefore, to account for imprecise sensor data due to noise or improper hardware quality a sensor error is introduced in our implementation. This error determines an artificial deviation from the otherwise perfectly calculated orientation and traveled distance.

In our simulations we focus especially on low seed density situations. In contrast to Hu and Evans [13] we define a more expressive metric for the seed density. Instead of considering only the ratio of seed nodes and simple nodes, we take all important parameters into account which impact the seed node coverage of the deployment area, i.e. the number of seed nodes $N_{\text{seed}}$, the radio range $r$ and the simulation area $A_{\text{sim}}$. Equation (3) shows how we define the seed density.

$$\rho_{\text{seed}} = \frac{N_{\text{seed}} \times (2r)^2}{A_{sim}}. \tag{3}$$

The basic idea is to cover the deployment area $A_{\text{sim}}$ with $N_{\text{seed}}$ squares with an area defined by the radio range $r$. Although modeling the transmission coverage as a square is an unrealistic and massively simplified model, our definition of seed density is much more expressive than other metrics which do not account for the dimensions of the deployment area or different radio ranges.

### 5.1 Simulation Parameters

Table 1 lists the different simulation parameters and their default values, which have been examined in order to evaluate SA-MCL. The simulation software of Hu and Evans does not provide a real unit system. All parameters are given with regard to the radio range $r$. A velocity of 0.4 means that between two localization approaches the node will move by a distance of $0.4 \times r$. For the ease of presentation we assume $r$ is given in meters.

The simulation scenario consists of a square with side length $a = 500\,\text{m}$ and a total of 300 nodes including seed nodes. Following Equation 3 a seed density of $\rho_{\text{seed}} = 1.6$ corresponds to about 40 seed nodes. For each simulation aspect one of the simulations parameters is changed while the others maintain their default value. We are interested in evaluating different network sizes, radio ranges, sample set cardinalities and seed node densities. Furthermore, we examine the behavior of SA-MCL assuming a certain sensor error to account for limited resolution of the additional sensors.

| Parameter | Meaning | Default |
|---|---|---|
| $N_{\text{nodes}}$ | Total number of nodes | 300 |
| $\rho_{\text{seed}}$ | Density of seed nodes | 1.6 |
| $v_{\text{max}}$ | Maximum velocity of nodes | 0.4 |
| $N_{\text{sample}}$ | Number of maintained samples | 25 |
| $r$ | Radio range | 50 |
| $\varepsilon_{\text{sensor}}$ | Error in sensor data | 20 % |

Table 1: Evaluated simulation paramters for SA-MCL.

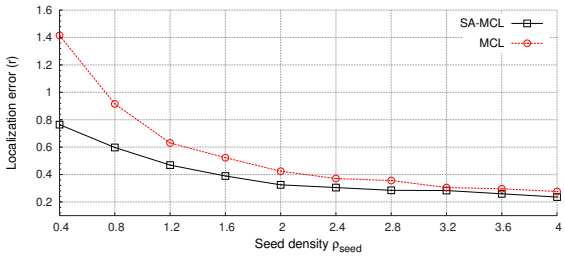



Figure 5: Localization error for different seed densities.

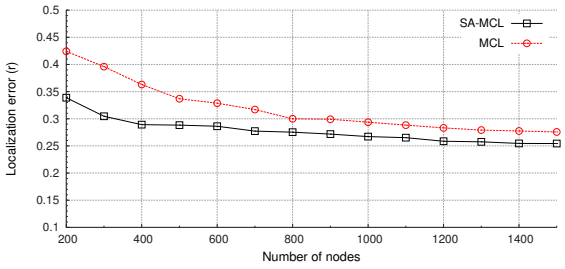



Figure 7: Localization error for different amounts of nodes.

## 5.2 Simulation Results

The following paragraphs present the localization error $\varepsilon_{\text{loc}}$ in terms of $r$. As in [13] it is calculated as the Euclidean distance of the real position $p_{\text{real}}$ to the estimated position $p_{\text{est}}$ multiplied by $r$ as shown in Equation (4).

$$\varepsilon_{\text{loc}} = \sqrt{(p_{\text{real}}.x - p_{\text{est}}.x)^2 + (p_{\text{real}}.y - p_{\text{est}}.y)^2} \times r \qquad (4)$$

### 5.2.1 *Effect of Different Seed Node Densities*

The key aim of SA-MCL is to account for situations in which the seed node density is low. In Figure 5 we show how using the additional sensor information helps SA-MCL to perform much better compared to MCL, especially if $\rho_{\text{seed}} < 2$. For this experiment the number of seed nodes is constantly reduced to achieve lower seed densities. It can be found that SA-MCL outperforms MCL by up to 40 % in low seed density situations. For higher seed densities MCL and SA-MCL tend to converge to a single graph as there are only few situations left where no seed information is available, i.e. only MCL is executed. The graph can also be read in a second way: assuming one wants to keep a certain level of localization error, it is possible to reach this level with considerably less seed nodes in SA - MCL compared to MCL. This is an important fact for applications in which nodes are not expected to be recovered after their mission, because the deployment costs can be drastically reduced if it is possible to reduce the number of costly seed nodes.

### 5.2.2 *Effect of Different Sample Set Cardinalities*

Several different sample set cardinalities, represented by the parameter $N_{\text{sample}}$, are examined to find a suitable number of samples which need to be maintained for satisfying results. The outcomes are shown Figure 6. Since the sample set cardinality heavily affects the computational overhead of the algorithms the aim is to maintain as few samples as possible. For only 1 maintained sample the localization error is very high, since a single sample not passing the filtering step results in an empty sample set, which makes a location estimation impossible. As soon as the number of samples is increased, the localization error is drastically reduced. It can be found that for both algorithms it is sufficient to keep a sample set cardinality of 25 to 30 samples. In accordance with [13] at a sample set cardinality of $N_{\text{sample}} = 50$ no more significant improvement can be found. The reason is that the additional samples do not provide any further information about the location of the node.

### 5.2.3 *Scalability in Large Networks*

An important aspect for algorithms used on devices with restricted computational capabilities is scalability. Scalability describes the ability of an algorithm to handle a growing amount of input while avoiding an increase of computational power necessary to solve the task [5]. An algorithm is considered to be well scaling if it is able to handle large inputs equally or with similar efficiency. In networking algorithms this means that an algorithm working on tiny-sized networks must perform equivalently good if the number of network participants is increased to hundreds or thousands of nodes.

Fortunately, the concept of range-free localization in association with the broadcast nature of wireless communication allows perfect scalability. In fact, the number of simple nodes a seed node can serve is not limited by computational means. Since the localization algorithm is executed on every single node, no increase of computational resources can be noted. The computational power required is only affected by the number of location announcements received and the sample set cardinality. The aim of any operator of a WSN will be to keep the number of seed nodes as low as possible. As a consequence, it is not likely to have networks with a huge amount of seed nodes. Furthermore, as seen previ-

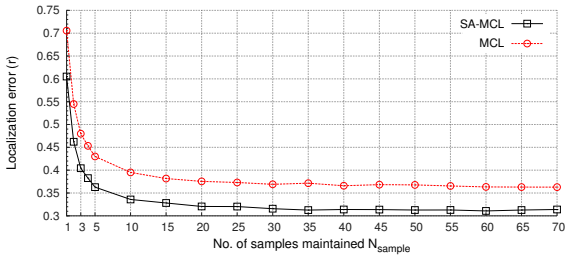



Figure 6: Localization error for different sample set sizes.

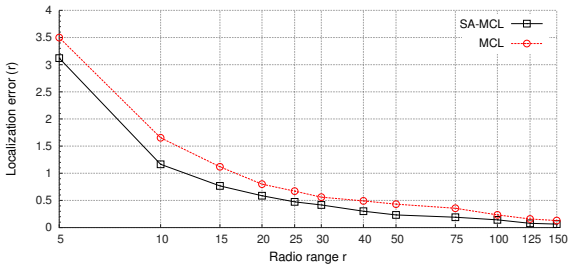



Figure 8: Localization error for different radio ranges.

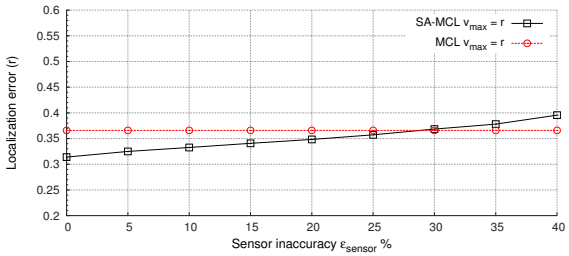



Figure 9: Localization error for different sensor precisions.

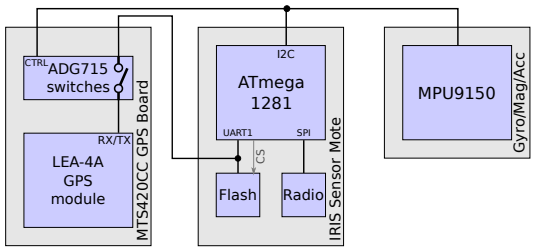



Figure 10: Data bus connections on the mobile node. Power supply lines not shown.

ously, there is no need to increase the sample set cardinality to more than 25-50 samples, as no further decrease of the localization error can be expected. This means both parameters affecting the computational overhead of a node are of no consequence.

To emphasize this Figure 7 shows the localization error for different amounts of nodes, $N_{\text{nodes}}$. The number of simple nodes is constantly increased, the number of seed nodes remains constant. Interestingly, the localization error is even slightly decreasing for both algorithms. The reason for that is the two-hop approach of MCL. Every location announcement of a seed node is forwarded once by all nodes receiving it. With an increased number of total nodes in the network more location announcements will be broadcasted again. Therefore, more nodes which would be isolated otherwise are still receiving location announcements. However, the effect of increasing the amount of nodes in the network is only very subtle and reaches a stable level at about 1200 nodes in this scenario. Further increase of the total number of nodes does not have any effect on the localization error.

#### 5.2.4 *Effect of Different Radio Ranges*

Following the definition of the seed density given above the radio range $r$ is an important parameter for the seed node coverage in the network. There is a trade-off situation between the localization error and the radio range: the smaller the radio range is, the lower the localization error will be given a suitable number of seed nodes to cover the network. Unfortunately, many more seed nodes are required to cover the whole network and to reach reasonable seed densities with smaller radio ranges. Figure 8 shows the localization error for MCL and SA-MCL for different radio ranges of both seed nodes and simple nodes. To provide a sufficient seed node coverage even for low radio ranges in this experiment 200 simple nodes and 100 seed nodes are used. It can be found that for both algorithms the localization error will increase if the radio range increases.

#### 5.2.5 *Effect of Errors in Sensor Data*

The main problem which could arise in SA-MCL is imprecise sensor data. While all electronic sensors have a limited resolution, they might also be affected by external impacts. For instance, a magnetometer is always influenced by strong magnetic fields which even pervade possible countermeasures like magnetic shielding. To account for imprecise hardware a sensor error, $\varepsilon_{\text{sensor}}$, is introduced in the simulations ranging from 0 to 40 %. The results are shown in Figure 9. MCL does not make use of sensor information, therefore the results do not change and are given as a constant line for reference. It can be found that even when increasing the sensor error up to 30 % SA-MCL performs better than MCL.

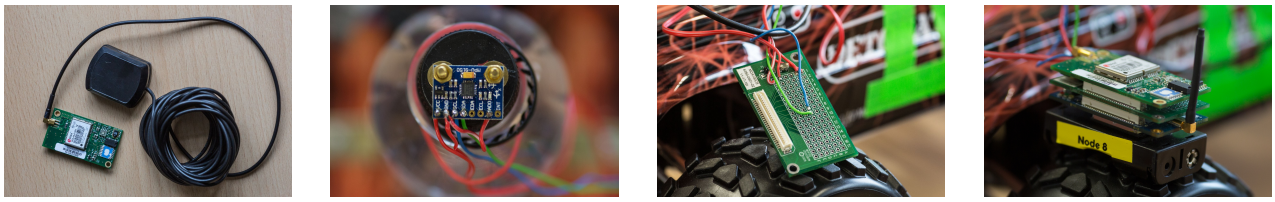
## 6 Field Test

Going beyond simulation, we build a practical implementation of MCL and SA-MCL and experimentally evaluate its performance in a field test. In this part of the paper, we first describe our mobile wireless sensor network testbed which is based on radio controlled cars (RC cars). Secondly, we describe the implementation of our approach and particularize how the sensor data is gathered. Finally, we describe our approach of parameter optimization and present our final results.

### 6.1 Hardware Setup

The hardware platform used for the field test is comprised of a small radio controlled electric vehicle carrying a sensor mote connected to a GPS receiver, accelerometer, gyroscope (angular rate sensor) and magnetometer. The sensor mote is an IRIS Wireless Measurement System manufactured by MEMSIC Inc. The mote's main board contains an Atmel ATmega1281 MCU, an 802.15.4/ZigBee-compliant 2.4 GHz radio and a 512 kB flash ROM for storing measurement results [18]. It also includes an expansion connector for connecting additional sensor boards. GPS functionality is provided by the Crossbow Technology MTS420CC sensor board (Figure 11a) which contains a LEA - 4A GPS receiver module by uBlox AG which is connected to the mote's UART1 serial port through the expansion connector (Figure 11c)[9]. As the flash chip and the GPS module are connected to the same UART, bus arbitration is necessary. This is done via a separate chip-select line for the flash chip [8] and via I2C-controlled ADG715 switch ICs for the GPS board (see Figure 10 for an overview).

An Invensense MPU9150 9 - axis motion sensor as shown in Figure 11a provides accelerometer and gyroscope sensors as well as the AKG8975 magnetometer which is included on-chip [14]. A breakout board containing the MPU9150 is mounted on a vertical plastic spacer giving about 20 cm of clearance from the vehicle's body using non-magnetic (brass) screws to minimize magnetic interference originating from the frame and motor. It is connected to the mote's $I^2C$-bus through wires and an interface board plugged into the expansion connector. The completed sensor mote stack including the mote itself, the GPS board and the expansion board is shown in Figure 11d.



(a) GPS module (b) MPU-9150 (c) Expansion board (d) Full mote stack

Figure 11: Mobile node hardware components.

Power is supplied to the mote and all sensors from the vehicle's 7.2 V NiMH-battery through a LM2596-based DC-DC step-down circuit, whose output has been manually adjusted to be in between 3.290 V and 3.300 V using a MASTECH M9803R multimeter [17].

The final car assembly is shown in Figure 12.

## 6.2 Motion Sensing

While estimating the compass heading of a sensor node by employing a simple approach that uses only a magnetometer, several challenges arise. Sensor noise leads to unsteady, fluctuating sensor data even at rest. Magnetic interference both from the RC car and the environment can lead to flawed magnetometer data. Finally, tilting the magnetometer introduces noticable errors and can easily occur while driving.

Sensor noise can, to some degree, be suppressed by means of low pass filters. Magnetic interference from the RC car is compensated by mounting the magnetometer on a spacer, as seen in Figure 12, keeping it away from the magnetic fields originating from the RC car. To mitigate magnetic interference from the environment and situations in which the magnetometer is tilted, accelerometer and gyrometer data is employed. Additionally, the accelerometer is used to determine if the sensor node is moving, which is important for running SA-MCL.

The gyrometer sensor provides angular velocity data, relative to the local orientation of the sensor node. This makes it possible to update a known orientation estimate using gyrometer data. To do this, the relative orientation offset has to be rotated to an absolute orientation offset and added to a previous estimate.

By measuring the vector of gravity using the accelerometer, it becomes possible to calculate pitch and roll of the sensor node while at rest. Together with the yaw reading

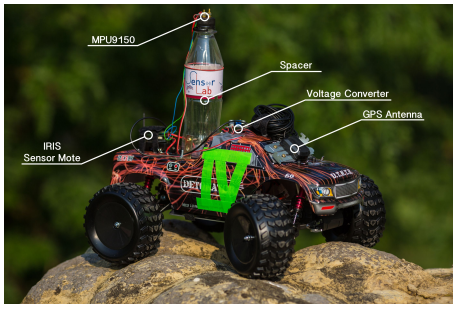



Figure 12: Fully assembled RC car mobile node.

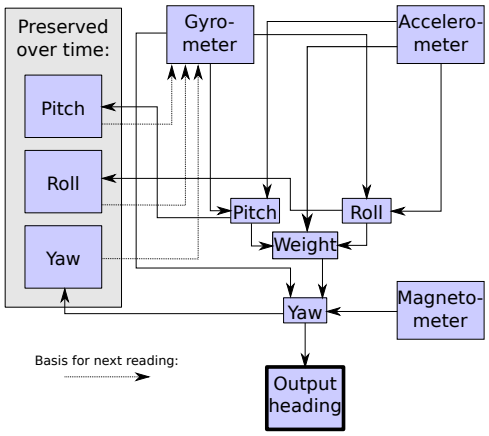



Figure 13: Motion sensing overview.

from the magnetometer, these values provide a base orientation of the sensor node that can be updated using the gyrometer. Since the gyrometer's accuracy is not impacted by acceleration, magnetic fields or tilt, a weighted average of the gyrometer's yaw estimate and the magnetometer yaw estimate provides a robust estimate of the sensor node's compass heading.

The weights are adjusted according to which adverse conditions are detected. If the sensor is moving or tilted, accelerometer and magnetometer receive lower weights. When not tilted or in motion, the gyrometer is aggresively weighted less and the magnetometer is used to reset the orientation, eliminating accumulated sensor drift.

In addition to the situational weighting approach, low pass filters are employed to reduce the noise of sensor data.

An overview of the approach can be seen in Figure 13.

## 6.3 Software Implementation

Due to the limited processing power and working memory, care has to be taken to avoid the use of expensive floating point calculations as much as possible. While anchor nodes receive their location information from GPS sensors attached to the extension connector, the longitude and latitude information is projected into a cartesian coordinate system by means of the Universal Transverse Mercator (UTM) map projection. The coordinates are centered around those of the site where the experiment takes place and stored in a 16 bit fixed point format with a resolution of approximately 3 mm. This allows relatively efficient computation of distances even within the limits of an 8 bit microcontroller without any significant loss in precision.

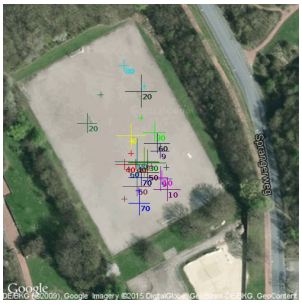


Figure 14: Screenshot of live view mode.

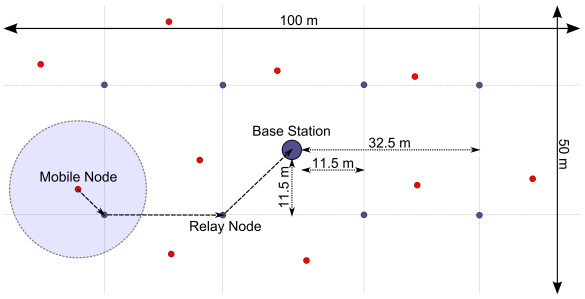



Figure 15: General field test setup.

Our implementation is able to complete one run of MCL with fifty samples within 70 ms to 80 ms on IRIS sensor motes. When running SA-MCL and using the previously described dead reckoning approach, the runtime is reduced even further to 6 ms to 7 ms, as no resampling of samples has to be performed. Both algorithms are executed directly on the hardware platform.

MCL and SA-MCL are both run once per second after which an announcement message is broadcast, containing anchor node information. GPS readings are also taken at a rate of one per second, while motion information, i.e. compass heading and a flag that determines if the sensor is moving at all, is measured about five times per second. All events such as measured sensor data and received or sent messages are logged to flash memory to allow for detailed analysis of everything that happened during the experiment.

Furthermore, our implementation features a live view mode which displays the positions of all sensors in the testbed as show in Figure 14. The live view mode is not required for determining the performance results of our evaluation, but provides a comfortable way to quickly detect problems.

## 6.4 Methodology

The following paragraphs describe how our field test is carried out.

### *6.4.1 Field test setup*

Ten remote controlled cars, which are configured as described above, are driven over a flat sports field, controlled by volunteers. The dimensions of the experimental area are 100 m × 50 m. The drivers of the cars are instructed to imitate the random waypoint mobility model. This means a car is supposed to drive a more or less straight line, pause for a short time, change direction and start over. To maintain control of the car and ensure safe driving the maximum velocity is limited to $v_{\max} \approx 6$ km/h. Data is collected over a period of about 18 min to 20 min.

To enable the live view mode it is necessary to forward the positioning information, which otherwise is stored only in every motes' flash memory, to a central base station. Therefore, a grid of relay nodes is required. These relays do not have any other functionality than forwarding packets to the base station. Figure 15 shows the field test setup with the base station located in the center, eight relay nodes arranged in a grid and the mobile nodes are allowed to move arbitrarily on the field.

To allow fine grained control of the nodes' radio ranges, an artificial limit is introduced by means of an RSSI cut-off value. If a node receives a location announcement, it will always record it in its flash memory log. However, if the RSSI is below a certain, configurable threshold, the announcement will not be processed further by MCL. Experimentally determined RSSI values for different ranges are given in Table 2. We are aware that this mapping is only a very rough estimation and does not hold true in general. However, packets misjudged based on this method are considered to be a consequence of the varying antenna characteristics on real hardware. To provoke many situations without seed information, the RSSI cutoff is set to 29, which corresponds to a radio range of ≈ 5 m according to our provided table.

To maximize the usable data gained from the experiment, all seed nodes also run both MCL and SA-MCL, i.e. every seed node is a simple node at the same time. To avoid a car localizing itself based on its own location announcements, announcements originating from itself are ignored.

### *6.4.2 Ground truth*

All cars are equipped with GPS sensors to allow determining the localization error with respect to the reference value provided by the GPS readings. We are aware of the fact that GPS introduces a localization error on its own. However, more precise reference systems usually involve multiple cameras and are beyond our budget. Seed nodes will announce their position also based on their current GPS information.

### *6.4.3 Data logging*

All data collected during the field test, including GPS data, sent and received location announcements and location estimations of MCL and SA-MCL are stored in the flash memory of each node. Using the logs gained from the field experiments, it becomes possible to run further simulations with very high fidelity based on the collected data afterwards.

## 6.5 Results

After performing the field test, we evaluate the performance of SA-MCL and MCL according to various error metrics.

| RSSI | 50 | 33 | 26 | 12 | 9 | 6 |
|---|---|---|---|---|---|---|
| Dist. | 1 m | 4 m | 8 m | 16 m | 22 m | 26 m |

Table 2: Mapping of RSSI value to distance.

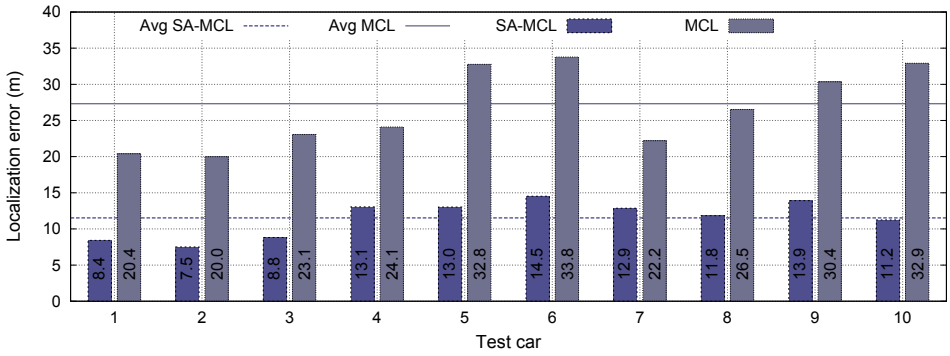



Figure 16: Absolute Localization Error.

### 6.5.1 *Absolute localization error*

Again, the absolute localization error is calculated as the Euclidean distance between the estimated position and the real position given by our GPS ground truth data. Our field test clearly shows that the performance of SA-MCL is superior to that of MCL. A per-car breakdown of the absolute localization error is given in Figure 16.

The average absolute localization error of SA-MCL during this experiment is 11.37 m, while the average absolute localization error for MCL is 27.1 m, which corresponds to an average improvement of 58 %. The maximum improvement is as high as 66 %, while even in the worst case there is still an improvement of 46 %.

### 6.5.2 *Grid localization error*

Intuiting that SA-MCL and MCL performance is different in different regions of the field test area, besides the absolute localization error another type of metric is developed, called grid error. For this error measure, the test field is subdivided into cells on a grid. The cells have a dimension of $\approx 3\,\text{m} \times 3\,\text{m}$. For each cell the average error at the times that the GPS coordinates are within the given cell is determined.

The resulting grid is then plotted with color coded errors in the style of a heat map. The grid error plot showing the averages of all ten cars can be seen in Figure 17, with the average error given in meters according to the scale. At a glance, it becomes obvious that especially at the outskirts of the field test area, SA-MCL outperforms MCL. This is likely due to a lower seed node density in those regions, which results in MCL's location estimates rarely entering those areas. The effect would be even more drastic on a larger test field.

### 6.5.3 *Optical trace*

Figure 18 shows a partial plot of the path taken by one of the RC cars overlayed on a satellite picture of the experiment's locale over a period of 3.5 min. The figure illustrates a period in which no seed information is available to the node. The yellow line corresponds to the location provided by the GPS sensor, while the red line gives the location estimate of SA-MCL and the green line shows the estimate calculated by MCL. Only a part of the full path is given, as a plot of the full path would make it hard to visually discern details due to its complexity.

It can be found that SA-MCL is almost perfectly imitating the real path of the car as given by the GPS data. Using the combined data of all sensors to determine the new heading SA-MCL is able to react immediately to changes in orientation. MCL on the other hand is unable to account for the missing seed information and therefore jumps from its last known location at the top left corner directly to the next determined location after receiving seed information again at the bottom of the test field.

### 6.5.4 *Current Draw*

One of the most quoted reasons for justifying the usage of localization algorithms instead of GPS is the high current draw of the latter. However, only few researchers provide a study of the energy consumption of their algorithms, often due to a missing real implementation. We measured the current flow of the fully assembled node when executing MCL,

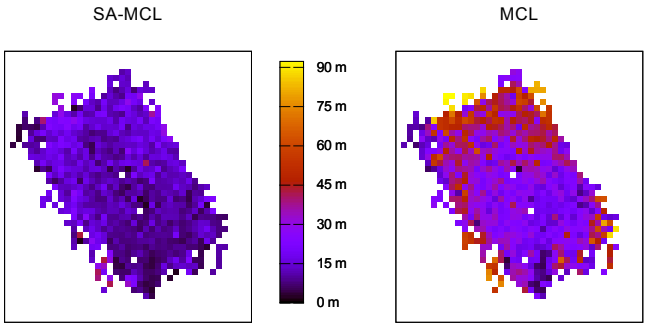



Figure 17: Average grid error over all cars.

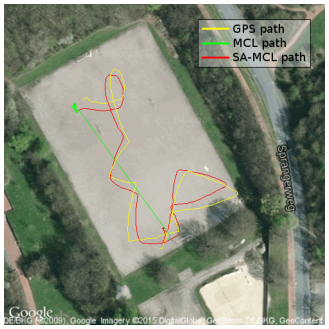



Figure 18: Path taken by RC car during experiment.

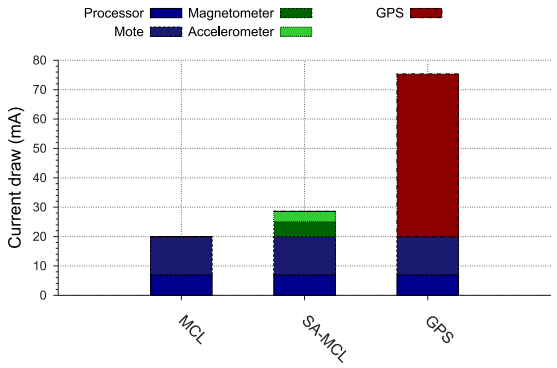



Figure 19: Current draw for MCL, SA-MCL and GPS.

SA-MCL and when using GPS with a MASTECH M9803R multimeter [17]. The current draw is measured over a period of 1 min with a frequency of 4 Hz. The gathered values are averaged to produce the final results which are shown in Figure 19. It can be found that the GPS device indeed is consuming a lot of power while the current draw introduced by our additional sensors is reasonably low. To be precise, the power consumption overhead of SA-MCL is about 8.6 mA compared to 55.38 mA for the GPS device. All in all, the total current draw for a node running MCL is about 20 mA which includes powering the mote itself. For SA-MCL the current draw is about 28.59 mA and 75.38 mA for a mote equipped with GPS respectively. This means, the consumed power of a node localizing using GPS is about 3 times higher.

## 6.6 Further evaluation

Based on the data and GPS traces collected in the field test, we are able to run further offline experiments to analyze the performance of SA-MCL and MCL with different parameters. The logs stored in the nodes' flash memory are fed into an especially designed simulator based on the code running on the actual sensor nodes. In this way, it becomes possible to evaluate the performance of the algorithms when configured with different parameter sets.

The main parameters of interest are the radio range, which is represented by the RSSI cut-off value, and the speed value, which describes how fast the samples are moved in SA-MCL when no seed information is available. The quality of this parameter is crucial for the performance of SA-MCL. The value of $v_{\max}$ also depends on this parameter and should be set to twice the value of the speed parameter, to ensure that the randomly resampled particles in the particle filter on average spread with a speed corresponding to that of the speed parameter.

Figures 20 and 21 show the absolute localization error in meters for both algorithms with different parameters. SA-MCL is examined with speed values in the range 0.075 m/s to 3 m/s with a step size of 0.075 m/s and RSSI cut-off values in the range $[0,40]$. It can be found that SA-MCL is robust against a decrease in radio range as long as its speed value is configured to match the average speed of the sensor node while in motion. In contrast, the localization error of MCL strongly increases if the radio range and thus the seed density decreases.

Figure 22 shows the performance of SA-MCL in comparison to MCL. This comparative error is calculated as the ratio between the average absolute error of SA-MCL and MCL as given in Equation(5). Here, it can be seen that speed settings below the optimal value still at worst make SA-MCL's performance approach that of MCL, but do at no point make it perceivably worse, while well calibrated values lead to consistently superior performance.

$$\varepsilon_{\text{loc }\%} = \frac{\varepsilon_{\text{loc SA-MCL}}}{\varepsilon_{\text{loc MCL}}} \times 100\,\% \tag{5}$$

# 7 Conclusions

In the final section of this work we would like to summarize the contents of this paper and list some possible improvements and extensions for future work. Supplemental material including the source code of our implementation and videos of the field test can be downloaded[1].

## 7.1 Summary

In this paper we presented our approach of using additional sensor information to implement a dead reckoning localization extension for the Monte Carlo Localization (MCL) algorithm. Our solution called Sensor-Assisted Monte Carlo Localization (SA-MCL) accounts for temporary situations without seed information due to connection loss caused by changing network topologies in mobile WSNs. By freezing the MCL sample set and moving all samples according to the motion information gathered from accelerometer, gyroscope and magnetometer we are able to localize a node relative to its last known position. Extensive simulation studies showed that our approach performs well even in large networks of hundreds of nodes. More importantly, the simulations indicate that we are able to reduce the localization error drastically in low seed density networks.

[1] http://filepool.informatik.uni-goettingen.de/publication/tmg/2016/material_ewsn2016.zip.

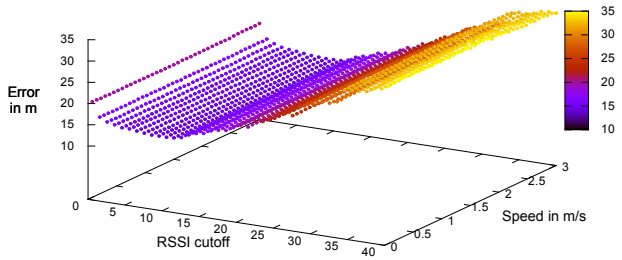



Figure 20: Absolute MCL error with varying parameters.

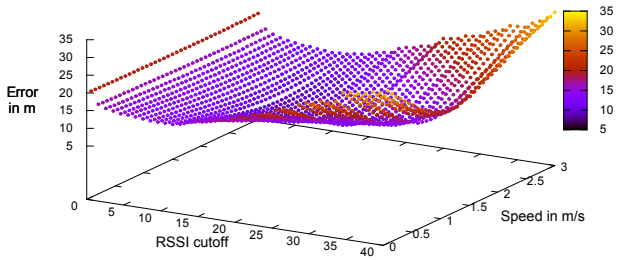



Figure 21: Absolute SA-MCL error with varying parameters.

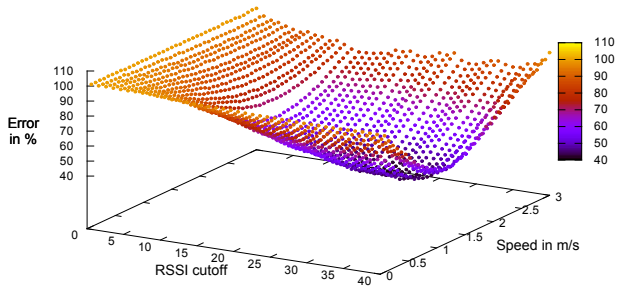



Figure 22: SA-MCL error percentage compared to MCL.

Our main contribution is to provide a real implementation of SA-MCL on mobile sensor motes which are mounted on radio controlled cars. According to our initial simulations the experimental results confirm that SA-MCL outperforms MCL in low seed density situations. Our approach can be used to either reduce the localization error drastically or to save costly seed nodes while maintaining the same level of localization error. The detailed logs of our field test can be used to run further simulation based on real GPS and network traces.

## 7.2 Future Work

In future research we would like to extend our work on mobile WSN testbeds. We believe that conducting experiments on real hardware is much more expressive compared to other studies conducted using simulation software only. The main problem in our testbed is the manpower required to drive the cars. We are interested in an autonomously operating system which controls the car without a real driver. Such a system could provide the basis not only for evaluating localization algorithms, but for all algorithms in mobile WSNs including routing, data aggregation, etc. which could encourage researchers to test new proposals and solutions more often on real hardware.

Besides our mobile WSN testbed in general, we are interested in combining our approach with previous works on improving MCL, which focus mainly on improving sample filtering as described in Section 3. None of the mentioned works was evaluated in a mobile sensor network testbed. Therefore, comparing them against each other and further improving them using SA-MCL can be a future contribution.

Additionally, SA-MCL could easily be extended to work in 3D space. The main challenges for validating such an extension in the real world would be measuring the ground truth locations and the a more complex experimental setup, replacing cars with flight capable vehicles.

# 8 Acknowledgments

We would like to thank all colleagues, who assisted during the field test. This work has been partly supported by the Simulation Science Center project sponsored by the Volkswagen Foundation and the state Lower Saxony, Germany.

# 9 References

[1] O. G. Adewumi, K. Djouani, and A. M. Kurien. RSSI Based Indoor and Outdoor Distance Estimation for Localization in WSN. In *14th IEEE International Conference on Industrial Technology (ICIT)*, 2013.

[2] S. Al-Jazzar, H. Strangeways, and D. McLernon. 2-d angle of arrival estimation using a one-dimensional antenna array. In *Proceedings of the 22nd European Signal Processing Conference (EUSIPCO), 2014*, pages 1905–1909, Sept 2014.

[3] I. Amundson and X. D. Koutsoukos. A survey on localization for mobile wireless sensor networks. In *Proceedings of the 2nd International Conference on Mobile Entity Localization and Tracking in GPS-less Environments*, MELT'09, pages 235–254, Berlin, Heidelberg, 2009. Springer-Verlag.

[4] F. Bellili, S. Ben Amor, S. Affes, and A. Samet. A new importance-sampling ml estimator of time delays and angles of arrival in multipath environments. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2014*, pages 4219–4223, May 2014.

[5] A. B. Bondi. Characteristics of Scalability and Their Impact on Performance. In *Proceedings of the 2nd International Workshop on Software and Performance (WOSP)*, WOSP '00, pages 195–203, New York, NY, USA, 2000. ACM.

[6] CH Robotics. *AN-1007 Estimating Velocity and Position Using Accelerometers*, October 2012.

[7] Crossbow Technologies. *MICAz Sensor Mote Platform*, Feb. 2015.

[8] Crossbow Technology Inc. *IRIS OEM Edition Hardware Reference Manual*, 2007.

[9] Crossbow Technology Inc. *MTS/MDA Sensor Board Users Manual*, 2007.

[10] F. Dellaert, D. Fox, W. Burgard, and S. Thrun. Monte carlo localization for mobile robots. In *Proceedings of the IEEE International Conference on Robotics and Automation, 1999*, volume 2, pages 1322–1328 vol.2, 1999.

[11] G. Han, H. Xu, T. Duong, J. Jiang, and T. Hara. Localization algorithms of wireless sensor networks: a survey. *Telecommunication Systems*, 52(4):2419–2436, 2013.

[12] S. Hartung, H. Brosenne, and D. Hogrefe. Practical RSSI Long Distance Measurement Evaluation in Wireless Sensor Networks. In *2013 IEEE Conference on Wireless Sensors (ICWiSe)*, Kuching, Malaysia, 2013.

[13] L. Hu and D. Evans. Localization for Mobile Sensor Networks. In *10th Annual International Conference on Mobile Computing and Networking (MobiCom 2004)*, pages 45–57, Phildadelphia, USA, 2004.

[14] Invensense Inc. *MPU9150 Product Specification*, 2011.

[15] V. Kafle, Y. Fukushima, and H. Harai. Design and implementation of dynamic mobile sensor network platform. *IEEE Communications Magazine*, 53(3):48–57, March 2015.

[16] L. Klingbeil and T. Wark. A Wireless Sensor Network for Real-Time Indoor Localisation and Motion Monitoring. In *Proceedings of the 7th International Conference on Information Processing in Sensor Networks*, IPSN '08, pages 39–50, Washington, DC, USA, 2008. IEEE Computer Society.

[17] MASTECH. *M9803R Bench Multimeter User Manual*, 2014.

[18] MEMSIC Inc. *IRIS datasheet*, 2010.

[19] A. T. Parameswaran, M. I. Husain, and S. Upadhyaya. Is RSSI a Reliable Parameter in Sensor Localization Algorithms - An Experimental Study. In *28th International Symposium on reliable distributed Systems*, 2009.

[20] M. Rudafshani and S. Datta. Localization in wireless sensor networks. In *6th International Symposium on Information Processing in Sensor Networks (IPSN), 2007*, pages 51–60, 2007.

[21] K. Seifert and O. Camacho. *Application Note AN3397*. Freescale Semiconductor, February 2007.

[22] G. Teng, K. Zheng, and W. Dong. MA-MCL: Mobile-Assisted Monte Carlo Localization for Wireless Sensor Networks. *IJDSN*, 2011, 2011.

[23] R.-H. Wu, Y.-H. Lee, H.-W. Tseng, Y.-G. Jan, and M.-H. Chuang. Study of characteristics of rssi signal. In *IEEE International Conference on Industrial Technology (ICIT), 2008.*, pages 1–3, April 2008.

[24] S. Zhang, J. Cao, C. Li-Jun, and D. Chen. Accurate and energy-efficient range-free localization for mobile sensor networks. *IEEE Transactions on Mobile Computing*, 9(6):897–910, 2010.

[25] Z. Zhong and T. He. Rsd: A metric for achieving range-free localization beyond connectivity. *IEEE Transactions on Parallel and Distributed Systems*, 22(11):1943–1951, Nov 2011.