Spray: A Multi-Modal Localization System for Stationary Sensor Network Deployment

Niklas Wirström
SICS,
Sweden.
niwi@sics.se

Prasant Misra
Robert Bosch Center for Cyber Physical Systems,
Indian Institute of Science, India.
prasant.misra@rbccp.org

Thiemo Voigt
Uppsala University and SICS,
Sweden.
thiemo@sics.se

***Abstract*—We present a localization system that targets rapid deployment of stationary wireless sensor networks (WSN). The system uses a particle filter to fuse measurements from multiple localization modalities, such as RF ranging, neighbor information or maps, to obtain position estimations with higher accuracy than that of the individual modalities. The system isolates different modalities into separate *components* which can be included or excluded independently to tailor the system to a specific scenario. We show that position estimations can be improved with our system by combining multiple modalities. We evaluate the performance of the system in both an indoor and outdoor environment using combinations of five different modalities. Using two anchor nodes as reference points and combining all five modalities, we obtain RMS (Root Mean Square) estimation errors of approximately 2.5 m in both cases, while using the components individually results in errors within the range of 3.5 and 9 m.**

I. Introduction

We target the problem of localizing sensor nodes in *stationary* sensor networks, in which no or limited infrastructure exists. Location information plays an important role in numerous wireless sensor network (WSN) applications [12] because it provides a context for interpreting sensed data, or is often even the data that needs to be sensed. In a stationary sensor network, this information can, in many cases, be specified manually, or be obtained through the Global Positioning System (GPS), or other Global Navigation Satellite Systems (GNSS). Manual localization, however, can be too time consuming to be practical for large scale deployments, or when this task is time critical, such as in rescue, firefighting, or military scenarios. Moreover, GNSS signals are not available everywhere. Examples of such cases are indoor environments, dense forests, or mountain regions. Another example in which they might not be applicable is military applications in which they might not be considered trustable, both for their vulnerability to jamming [16], and the fact that they are controlled by external organizations.

Consequently, localization methods that rely on local measurements are more desirable in these cases [4]. These methods, however, also suffer from a number of inherent problems, such as measurement inaccuracy that arises from noise, multi-path effects, blockages, interference, clock drifts, or environmental effects [5], [10], [30]–[32]. There are currently no measurement methods that produce reliable results in all types of operational conditions, and especially not when nodes are not in line-of-sight (LOS). Moreover, in absence of existing localization infrastructures, it is typically necessary to manually specify a number of reference points. It is important that the required number of such reference points is kept to a minimum, to limit the amount of manual work. Although individual local measurements typically are of low accuracy, the final position estimation can be improved by combining measurements from several measurement modalities.

In this paper we present *Spray*, a particle filter based WSN localization system. The two main differences between *Spray* and other particle filter based approaches is that: 1) it estimates stationary sensor node positions without relying on existing localization infrastructure, as opposed to tracking moving objects using an existing infrastructure with known node positions [6], [11], and 2) it provides a component based framework into which multiple localization modalities can be easily incorporated, in contrast to other approaches that tightly couple a single modality to the fusion algorithm [8]. This enables *Spray* to be tailored to different deployment scenarios in which different sets of modalities are available.

We empirically evaluate *Spray* in real-world, indoor and outdoor experiments using different combinations of 5 different modalities. We show that, in many cases, the accuracy can be improved significantly by combining localization modalities, and by combining all modalities, we obtain node position estimation accuracy of around 2.5 m, using only 2 anchor nodes as reference points. We also compare our system, using a single modality, to a traditional localization approach and demonstrate that it performs well also in this basic case.

The main target applications for *Spray* are time critical applications in which position information of sensor nodes are needed, and GNSS systems cannot be used, or are undesired for other reasons. Applications include firefighting and other rescue scenarios in which rescue workers leave a trace of sensor nodes behind them. These systems are often referred to as *breadcrumb systems* [13]. The sensors can be used for tracking the rescue workers' position, and to monitor the surroundings, e.g., to detect major temperature changes, falling objects, or moving people. This can be valuable information to monitor how a fire is spreading in a building, to determine if paths have potentially become blocked, or to localize people to be rescued. In these scenarios GNSS might not be applicable because of two reasons: 1) GNSS signals might not be available, due to indoor conditions, and 2) it might be considered too vulnerable to jamming, in case of, e.g., a terror attack. A second example is a police, or military application in which a building or a region rapidly needs to be secured by monitoring people

978-1-4799-4937-3/14/$31.00

entering, exiting or moving inside it. In many cases, a position accuracy on the order of a few meters is sufficient in these scenarios.

The rest of the paper is organized as follows: Section II presents the related work. In Section III, we describe the internals of *Spray*. Section IV presents the evaluation results, and finally, Section V concludes the paper.

II. Related Work

For decades, localization has been of high interest in the field of robotics where most of the associated problems can be divided into two separate categories: *tracking* and *navigation*. Solutions to both these problems typically rely on a motion model that describes the dynamics of the object to be localized, and information fusion techniques that combine information from multiple sensors. For this purpose, it is common to use Bayesian filters, such as Kalman filters (KF), extended Kalman filters (EKF), or particle filters (PF) [1]. A drawback with KF is the assumption that the system and measurement processes are linear. This fails to hold in a variety of realistic scenarios. EKF approximates this linearity assumption by assuming the measurement process to have a Gaussian probability density. Although, particle filters are considerably more computational intensive than KF and EKF, they are more modular because they represent probability distributions by a set of discrete points that can be manipulated individually.

Stationary WSN localization is different from tracking and navigation in *two* fundamental aspects. *First*, in a robot navigation application, the robot can move and improve position estimations by using new sensor readings in combination with the motion model. For static node localization, there is no motion involved. This means that there are no ways to obtain a second set of measurements from a second location if the current measurements are inaccurate or are coherent with multiple different locations. *Second*, nodes that are far away from anchors might have to be estimated recursively in the sense that their positions are estimated based on previously estimated positions of other nodes, which results in the propagation of measurement errors. Together, these two factors makes it non-trivial to apply the robot localization techniques to the problem of static WSN localization.

For this reason, many other schemes have been proposed for static WSN node localization. Some of these are based on the assumption that nodes have LOS, existing infrastructures, or specific geographical information [3], [5], [9], [17]–[19], [21], [24], [28]. Moreover, most approaches rely on mathematical models that are difficult to modify to incorporate more than one localization modality [15], [25], [27]. Within the field of WSN, Bayesian filters have mostly been used for tracking or navigation applications [2], [6], [10], [11], [26], [29], out of which few exploit the modularity of particle filters [6], [11], [22]. There are approaches that use particle filter based solutions for *stationary WSN node localization*, but these are limited to location estimation based on a single modality: Rudafshani et al. [23] rely on a PF to estimate locations based on range-free information; Huang et al. [7] use a decentralized PF to estimate node positions (in a potentially mobile WSN); and Ihler et al. [8] use non-parametric belief propagation, a generalization of PF, for the same purpose.

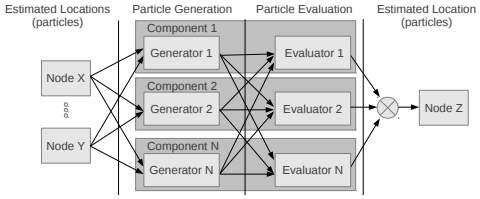


Fig. 1. *Spray* decouples the localization modalities from the fusion algorithm by encapsulating each modality in an individual component. Each component has, potentially, its own dedicated particle evaluator and generator, which are used to weight, and generate new particles, respectively.

In *Spray*, we build on the same ideas, but take a next step to extend to *multiple* localization modalities to minimize the final position estimation errors caused by imperfections in measurements of individual modalities. In contrast to [7], [8], [23], *Spray* provides a framework in which new localization modalities can be defined in terms of generators, evaluators, and a voting mechanism to determine the order in which node positions will be estimated. Moreover, their proposed algorithms are only evaluated by simulation, while we evaluate *Spray* in real-world experiments.

III. System Architecture

The core of *Spray* consists of a centralized particle filter based algorithm in which the particles represent possible node locations. Here, we explain the basics of particle filters in general, and then we describe the particle filter used in *Spray*.

A. Particle Filters

A particle filter represents the state s_t of a system at time t, by a set of N discrete samples known as particles $\{p_t^k\}_{k=1}^N$, each with an associated weight w_t^k. The state is updated by generating a new set of particles $\{p_{t+1}^k\}_{k=1}^N$ based on a system model, and the current set of particles $\{p_t^k\}_{k=1}^N$. In the case of navigation and tracking, this system model is typically a motion model. At each time t, a measurement y_t of the world is performed (e.g., by measuring the distance to walls, etc.,), and each particle p_t^k is weighted based on the likelihood for the system of being in state p_t^k and observing the measurement y_t. The weights are, subsequently, normalized such that $\sum_k^N w_t^k = 1$. The system state can then be estimated by $\hat{s}_t = \sum_{k=1}^N w_t^k p_t^k$. An additional step known as *resampling* is typically performed at specific intervals. It consists of selecting N particles *with replacement* such that each particle p_t^k is selected with a probability w_t^k. The particles are then, typically, reweighted such that each weight $w_t^k = 1/N$.

B. The Spray Particle Filter

To enable *Spray* to be tailored to different scenarios in which different sets of measurement modalities are available, the system represents each measurement modality as a separate *component* (See Fig. 1). Each component has an associated particle *evaluator*, which is used to weight particles according to their coherence with the measurements corresponding to that modality. Components can also have an associated particle *generator*, which is used to generate particles in regions where the true node position is likely to be, according to the measurements. The use of generators is simply an optimization that limits the number of generated particles. Without this

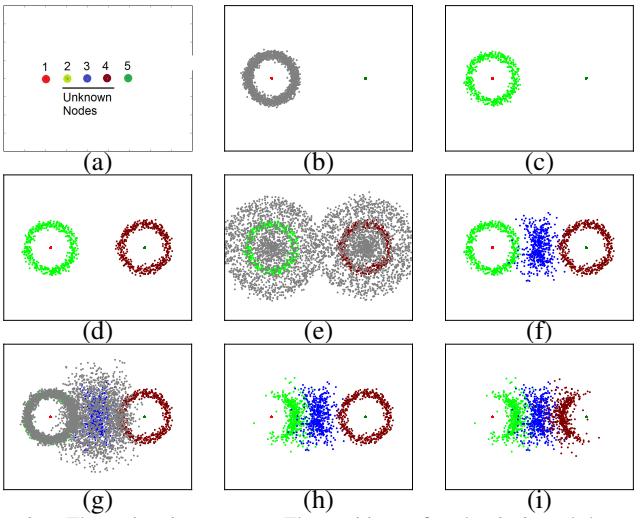


Fig. 2. The estimation process. The positions of nodes 2, 3, and 4 are to be estimated based on range measurements. Nodes 1 and 5 are anchors. The process consist of alternating between generation, evaluation, and resampling of particles.

optimization, particles would have to be generated over the entire area of interest. Some components do not implement generators, such as the map component because it would not result in an optimization, since particles would have to be generated over, potentially, very large regions. Components only define how measurements are used, and are agnostic how they are collected.

As mentioned in Section II, WSN localization of stationary nodes is different from traditional navigation or tracking applications in that it involves only nodes that do not move once they are deployed, and that it must handle nodes for which there are no direct measurements available from the anchor nodes. Here follows an overview of the estimation process.

1) An Overview of the Estimation Process: The estimation phase is an iterative process. In each iteration, each node's position is estimated, one at a time, until all unknown nodes have been positioned. Let us consider the single modality scenario shown in Fig. 2(a) where three nodes with unknown positions (nodes 2, 3, and 4) are to be localized with the help of two anchor nodes (nodes 1 and 5). Let us also assume that only one range measurement for each immediate neighboring pair of nodes is available.

Initially, particles for node 2 are generated based on the known position of node 1, and the range measurement between node 1 and node 2 (Fig. 2(b)). Then, as shown in Fig. 2(c), a subset of these particles are selected to represent the temporary estimate of node 2. However, with only one range measurement, we cannot obtain a better estimate than this circular cloud. The same process is subsequently executed for node 4 using the range measurement to node 5 (Fig. 2(d)). Then, node 3 is estimated based on the estimated positions of nodes 2 and 4, and their corresponding measurements to node 3 (Fig. 2(e)). Thereafter, a subset of these particles are selected based on the weights assigned to them based on both measurements (Fig. 2(f)). This concludes the first iteration with a temporal position estimate computed for all nodes. In the second iteration, all estimated node positions will be refined. Fig. 2(g) shows the particles generated for node 2 in this iteration. This time, the measurement to node 3 is also used, as we now have a position estimation for it. Fig. 2(h) shows the selected subset after evaluation. The corresponding procedure is then carried out for node 4, and the result is shown in Fig. 2(i).

As illustrated by the example scenario, an important aspect of the filter is the order in which node positions are estimated. One reasonable approach is to start with the nodes that have direct measurements available to an anchor node, and then, recursively, continue with nodes that have direct measurements to previously estimated nodes, and so on. Different modalities may provide measurements between different pairs of nodes, and therefore, it is not possible to define a generic estimation order that is optimal with respect to all modalities. In *Spray*, the estimation order is based on special priority weights that are assigned to nodes by the different components. The geometric mean is then computed from the individual weights to form the total weight for each node. We use the geometric mean, rather than the arithmetic mean, because of its insensitivity to differences in scale between the elements averaged.

2) Individual Node Estimation: We now explain the process of estimating the position of a single node as illustrated in Fig. 1. In this description, we assume that each node location is represented by N particles, each with a position and a weight, and that all weights are normalized such that their sum is equal to 1.

The process of estimating the position for node i is as follows: (1) Each generator generates a number of particles. (2) Each evaluator e computes a weight z_e^k for each particle p^k. Evaluators also compute weights for particles that have been generated by other components. The manner in which an evaluator computes these weights depends on the nature of its associated modality. (3) The geometric mean is then computed to estimate the total weight for each particle p^k according to the following equation: $w_i^k = \prod_{e=1}^{n}(z_e^k)^{\frac{1}{n}}$, where n is the number of evaluators. The reason for using the geometrical mean is that it is insensitive to difference in scale of the weighting functions of the different components. (4) Finally, the particles are resampled with replacement as described in Section III-A, such that N particles are selected to represent the node's location.

C. Implemented Components

In this section, we describe the internal structure of the implemented components. Components estimate node positions, potentially based on previously estimated positions of other nodes. This involves selecting reference particles, from the node with an already estimated position, for both generation and evaluation of the particles for the node whose position is to be estimated. For example, if we have a range measurement r between node i and node j, and node j already has an estimated position, we select a reference particle from node j, and generate a new particle at a distance r from it. This reference particle can be selected in many ways. Let us assume that the position of node j is already estimated, and that we want to estimate the position of node i. One approach, using the same notation as in previous sections, is to use all particles of node j to evaluate each particle of node i by computing the weights as: $z_e^k = \sum_{q=1}^{N} w_j^q f(p_i^k, p_j^q)$. Here, e denotes the evaluator corresponding to a specific measurement from node

i, and $f(\cdot)$ is the weighting function for a particular evaluator. However, if M is the total number of generated particles, this requires MN computations per measurement. This can, in many cases, be too computationally intensive to be practical. Instead, we use a single particle $p_j^{mod(s+k,N)}$ as a reference, where $s \sim \mathcal{U}(1, N)$ is a uniformly distributed random offset and evaluate the weighting function as $f(p_i^k, p_j^{mod(s+k,N)})$. This results in only M computations per measurement. Although, there is a risk that an outlier particle is selected as reference, the consequences of mis-weighting single particles are small when the number of particles is high.

Spray makes it possible to define new components using only a few number of lines of component specific code. The components described below are implemented using 4 lines for each generator, between 4 and 9 lines for the evaluators, and between 21 and 40 lines for the ranking mechanisms. The ranking mechanisms are the functions used to compute the evaluation order of the nodes.

1) The Range Component: Given a range measurement r^* and a reference particle q, the range component generates a new particle p according to Equation (1):

$$p = q + r\,[\cos(\alpha) \quad \sin(\alpha)] \tag{1}$$

where $r \sim \mathcal{N}(r^*, \sigma^{gen})$, and $\alpha \sim \mathcal{U}(0, 2\pi)$. Here, $\mathcal{N}$ and $\mathcal{U}$ denote the normal and uniform distributions, respectively. The result is a circular cloud of particles around node j with mean radius r and width determined by σ^{gen}. The range component's evaluator weights a particle p by first computing its distance $d_{p,q} = \| p - q \|$ to the reference particle q, and then, computing the weight using the probability density function $f_{\mathcal{N}}(\cdot)$ of the normal distribution with mean r and standard deviation σ^{eval} as $z = mf_{\mathcal{N}}(d_{p,q}, r^*, \sigma^{eval})$, where $m = 1/f_{\mathcal{N}}(0, r^*, \sigma^{eval})$ is a normalization factor.

2) The Step Count Component: In some scenarios, sensor nodes are deployed by foot. By counting the steps between node locations, it is possible to get an upper limit of the distance between two nodes. This information is, typically, not enough by itself to give accurate node estimates, but together with e.g. range measurements, it can improve the estimation accuracy by complementing range measurements that have large errors due to multi-path effects. With n as the number of steps and l as the step length in meters, the generator of the step count component generates particles on a disc with radius nl from the reference particle q as in Equation (1), but with $r \sim \mathcal{U}(0, nl)$. The corresponding evaluator weights a particle p by first computing the distance $d_{p,q}$ to the reference particle q, as in Section III-C1, and then computing the corresponding weight $z = 1$ if $d < nl$, and $z = mf_{\mathcal{N}}(d, nl, \sigma^{eval})$ otherwise. Again, m is a normalization factor such that $z = 1$ when $d = nl$. This way, particles that are within the radius nl are given a weight of 1. Weights are then decreased at a rate depending on σ^{eval}.

3) The Dead Reckoning Component: The dead reckoning component combines step counting with directional measurements for each step. Given that the directional measurement β_i for the ith step and the total number of steps n, the displacement can be computed as: $y = nl \sum_{i=1}^{n} [\cos(\beta_i) \quad \sin(\beta_i)]$. Using the polar representation $[r^* \quad \alpha^*]$ for y, the dead reckoning particle component generates particles according to Equation (1), but with $r \sim \mathcal{N}(r^*, \sigma_r^{gen})$ and $\alpha \sim \mathcal{N}(\alpha^*, \sigma_\alpha^{gen})$. The evaluator weights a particle p using the *bivariate* normal distribution $f_{BVN}(\cdot)$, a reference particle q and its polar representation $[k \quad \gamma]$, according to Equation 2, in which we have assumed that the errors of r^* and α^* are mutually independent. m is, again, a normalization factor.

$$z = mf_{BVN}\left(\begin{bmatrix} k - r^* \\ mod(|\alpha^* - \gamma|, \pi) \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} \sigma_r^{eval} & 0 \\ 0 & \sigma_\alpha^{eval} \end{bmatrix}\right). \tag{2}$$

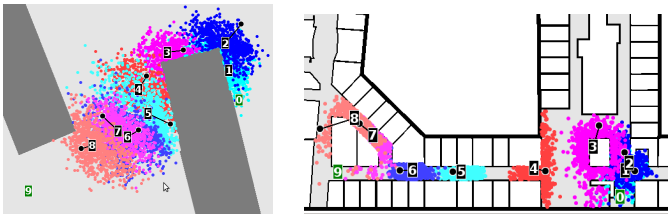
4) The Non-Proximity Component: The non-proximity component exploits the fact that if two nodes are not within each others radio range, it is also unlikely that they are physically close to each other. Nevertheless, this assumption can be false if, for example, there is a thick wall between the nodes. The non-proximity component has no associated particle generator as a very high number of particles can be needed to cover the regions regarded as possible. The evaluator weights particles based on a reference particle q, and a limit h as $z = 1$ if $d_{p,q} > h$, and $z = mf_{\mathcal{N}}(d_{p,q}, h, \sigma^{eval})$ otherwise. This is very much the opposite to the evaluator of the step counting component. The result is that particles outside a circle of radius h are assigned a weight $z = 1$, and weights decrease within this circle with a rate determined by the parameter σ^{eval}.

5) The Map Component: The map component has no particle generator, for the same reasons as the non-proximity component. The evaluator weights particles according to the region in which they are positioned on a graphical map. It is binary in the sense that it assigns a weight of 1 to particles in regions defined as possible, and 0 otherwise. Maps can contain useful localization information. In some cases maps can be automatically extracted from the Internet. In an outdoor deployment, for example, all regions covered by buildings can be defined as impossible. The key benefit of using maps is that they can limit the solution space for position estimation. This is particularly true if there are nodes without direct measurements to anchors, in which case positions must be derived from previous position estimates.

IV. Evaluation

In this section, we evaluate *Spray* in both an outdoor and indoor environment. The measurements are made using inexpensive COTS products with low accuracy in many cases. More accurate measurements can presumably be obtained using more advanced hardware. The purpose of the experiments, however, is *not* to evaluate the performance of different individual modalities in the two scenarios, nor to compare the results from the two scenarios with each other. What we want to show is that *although* measurements are imperfect, we can obtain relatively high localization accuracy by combining multiple modalities using *Spray*. To demonstrate the independence on specific hardware, we also use different hardware setups in the two scenarios.

In Section IV-A, we present the experimental settings and the two scenarios used for evaluation. In Sections IV-B through IV-D, although *Spray* is agnostic to how measurements are obtained, we present the methods we use to collect measurement data, and present micro-benchmarks for the range



(**a**) (**b**)

Fig. 3. An outdoor (**a**) and an indoor (**b**) scenario. Black dots show the true node positions, and the squares containing numbers show the estimated positions. The lines between the dots and the squares, show the relation between position estimations and the true positions.

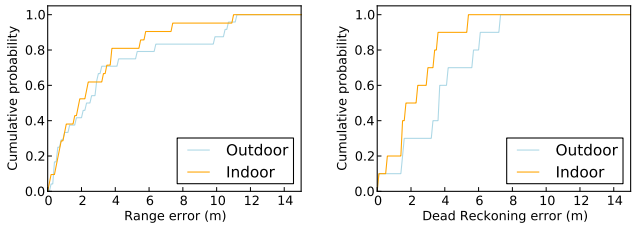


Fig. 4. Estimation errors of the ranging **(left)**, and dead reckoning **(right)** methods. The ranging errors are given as the difference between true and estimated distances between nodes. For both scenarios, 75-80 % of the range measurements are below 5 m. For dead reckoning, the error is given as the difference between the true and estimated absolute positions of the nodes. 70-90 % of the estimation errors are below 5 m.

and dead reckoning components. The purpose is to report the quality of the input measurements. In Section IV-E and IV-F, we investigate how different internal parameter settings affect the performance of *Spray*. In Section IV-G through IV-I, we evaluate our system by: 1) Comparing *Spray*, using only the range component, to a multilateration based algorithm. 2) Analyzing the contribution of the different components by evaluating all possible combinations. 3) Using simulated results to investigate in which cases benefits can be expected from fusing measurements. 4) Evaluating *Spray* using a single anchor as reference point.

A. Experiment Settings

We evaluate *Spray* in two different scenarios: An outdoor, and an indoor scenario shown in Fig. 3(a) and (b), respectively. In the outdoor scenario the dark gray regions represent buildings, and the gray region in the indoor scenario it constitutes a corridor in an office area. The distances between nodes 0 and 9 in the outdoor and indoor scenarios are 31 m and 43 m, respectively. Both scenarios fit well with the military and police applications outlined in Section I. The indoor scenario also fits well with the fire fighter scenario. Each experiment in this section consists of 100 different runs using the same setup (i.e. component parameters, number of particles, anchor nodes, etc). For each run, *Spray* is executed for 40 iterations. For each iteration we compute the RMS (Root Mean Square) error as a single error measurement for all the nodes in the scenario. We then compute the mean for each iteration over the 100 runs, resulting in a single average RMS value for each iteration for a specific setup. We refer to this error as the *mean RMS error*. As an error measurement for a whole setup, we take the average of the mean RMS errors for the 30 last iterations, and present this value together with the corresponding standard deviation.

B. The Range Measurements

We use the multi-channel ranging method we developed earlier [20]. This method, an extension of Mazomenos' method [14], is not as precise as some cross-correlation methods under LOS conditions using sound [15], but shows better performance in NLOS conditions. Moreover, it allows for the use of a standard IEEE 802.15.4 radio transceiver, which is commonly available on WSN motes. In the outdoor experiment, we use the STM32W RF Control kit development boards from STMicroelectronics. The stm32w has an integrated IEEE 802.15.4 radio and a 24 MHz crystal oscillator. Ranging with these nodes was performed without external antennas. In the indoor experiment, we use MSP430-based sensor nodes with a CC2420 radio, a 8 MHz crystal oscillator, and external antennas. Range measurements are performed by letting each sensor node, one at the time, perform measurements to the other nodes. Naturally, measurements will only be obtained for nodes within range. The absence of range measurements is used as information for the non-proximity component. The empirical cumulative distribution functions (CDF) for the ranging errors, for both the indoor and outdoor experiments, are shown to the left in Fig. 4. We observe that 75-80 % of the measurements are below 5 m, and that all ranging errors are below 11 m.

C. Dead Reckoning Measurements and Step Counting

Although dead reckoning is not a stationary modality per se, it can be used in breadcrumb systems by tracking the person deploying the network, in combination with information about when nodes are deployed. The dead reckoning measurements are collected using an Android smart phone, running a customized application, attached to the person deploying the sensor nodes. The application logs both the accelerometer, and magnetometer values (i.e., the absolute direction) from the phone to identify steps and compute the direction of each step. The phone is kept in a horizontal position, pointing forward, during the whole procedure. All nodes are visited in a single path. The same step measurements are also used for the step counting component. The right part of Fig. 4 shows the CDFs of the errors of the absolute position estimations computed from the raw dead reckoning measurements for both scenarios. This is computed without the use of *Spray*, which is equivalent to using *Spray* with only the dead reckoning component with $\sigma_r^{gen} = \sigma_\alpha^{gen} = 0$. We use node 0 as the only reference point. We observe that 70-90 % of the measurements are below 5 m, and that all errors are below 8 m.

D. Maps

Fig. 3 shows the maps used in the two scenarios. The light-gray regions define the node placement zones. For the outdoor experiment, only parts that are known not to be buildings are regarded as possible locations. This information can easily be extracted from services available for free on the Internet. The map for the indoor scenario, however, contains much more detailed information which cannot be obtained automatically from the Internet. On the other hand, emergency exit maps are available for many buildings. These can be used in combination with drawing program-like tools to define regions where nodes can be positioned. In this case, there is a trade-off between the number of nodes that are to be deployed and the extra work of defining the possible regions.

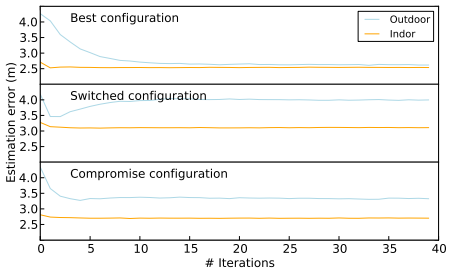


Fig. 5. Different Parameter Configurations: The evolution of the mean RMS error for 40 iterations, using the best, interchanged, and compromise parameter configurations.

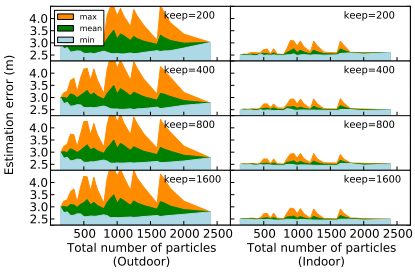


Fig. 6. The impact of different number of generated particles depends on for different numbers of particles that are kept to represent each node position.

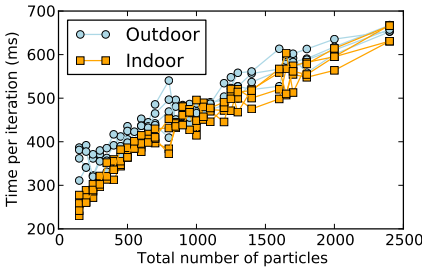


Fig. 7. Average time to complete an iteration, for different number of generated particles. It is approximately linear and does not differ significantly between the two scenarios.

E. Component Parameter Settings

Components are typically associated with a set of parameters that determine the behavior of both generation and evaluation of particles. An example of this is the standard deviation σ^{gen} of the ranging component, that determines the width of the particle cloud. The choice of parameter settings profoundly affects the performance of *Spray*. We run a parameter sweep for different combinations of values for all the σ-parameters defined in Section III-C. We select the best configuration for each of the two scenarios and run it 100 times. The top figure in Fig. 5 shows how the mean RMS error evolves over the 40 iterations for these configurations when nodes 0 and 9 are used as anchors. The errors have converged after approximately 5 iterations in the outdoor scenario, while almost instantaneously in the indoor scenario. Moreover, the two scenarios yield comparable results. The middle figure in Fig. 5 shows the result when the two configurations have been interchanged such that the best configuration for the indoor scenario is applied to the outdoor scenario, and vice versa. As expected the errors increase in both cases. Notably is also that the estimation error for the outdoor scenario decreases to a certain point, and then starts to increase again. The errors do, however, converge to a reasonable level in both cases. We now derive a compromise configuration by selecting the configuration that minimizes the sum $e_{in}(c_i)^2 + e_{out}(c_i)^2$, where $e_{in}(c_i)$, and $e_{out}(c_i)$ are the errors for the indoor, and outdoor scenarios, respectively, resulting from applying configuration c_i. The bottom figure in Fig. 5 shows the result of applying this configuration to both scenarios. The performance is improved in both scenarios compared to the switched-configuration setting. This shows that, for these scenarios, it is possible to find a common configuration that yields reasonable performance.

F. The Optimal Number of Particles

We investigate how the estimation accuracy depends on the number of particles that are generated for each measurement, and the number of particles that are kept, i.e., resampled to represent a single node position. We refer to these parameters as `gen` and `keep`, respectively. Each component that has a particle generator (i.e. the range, dead reckoning, and the step counter components, in our setup), has its own `gen` parameter that specifies the number of particles to generate for each measurement. We perform a parameter sweep over all different combinations of `gen` in the domain (50, 100, 200, 400, 800) at different settings for the `keep` parameter (200, 400, 800, 1600). Fig. 6 shows the results of this sweep. The x-axis of the figure indicates the sum of the different `gen` values of all components. That is, if the range, step counting, and dead reckoning components, in a particular sweep instance, use a value of `gen` equal to 50, 400, and 100, respectively, the sum is 550. This is however, not the *actual* number of particles that is generated, since that also depends on the number of measurements that exists for each component. In general, multiple sweep instances result in the same sum. The three stacks in each sub-figure, represent the max, mean, and min estimation error for all the sweep instances resulting in the `gen` sum indicated by the x-axis. Hence, the min curve represents the best configuration for a given sum. From the min curves, we see that, for the outdoor scenario, the estimation error for the best configuration decreases with the number of particles up to a certain level. Then it starts to increase again.

The performance for the indoor scenario, on the other hand, seems independent of the different values of `gen` and `keep`. A reason for this could be that the possible locations, as defined by the map, represent, mostly, relatively narrow corridors, rendering a low number of particles sufficient. From the figure, we also see that varying `gen` and `keep` within these ranges has little impact on the measurement error (only in the order of cm). The peaks in the max stack, representing the worst configuration for a given sum, shows that some configurations yield relatively high errors. These correspond to when few particles are generated for the dead reckoning component, and many are generated for, at least one of the range and step counting components. The dead reckoning measurements, as will be seen in Section IV-H, are the most accurate measurements, and if a comparatively low number of particles is generated for this modality, the probability of selecting many *worse* particles is high.

Fig. 7 shows the average time it takes to complete an iteration, i.e. to estimate all the node positions a single time, for different number of generated particles on a quad core 3.4 GHz PC. The x-axis representation is the same as in the previous figure. The time per iteration is approximately linear in the total number of generated particles, and does not differ significantly between the two scenarios. The average iteration time for a total of e.g. 1000 particles is approximately 500 ms. Running the filter for 20 iterations, as suggested in Section IV-E, results in a total estimation time of approximately 10 s, which is an acceptable time frame for the targeted applications.

G. Spray vs. Multilateration

Here, we compare *Spray* to a multilateration based algorithm. Multilateration is a standard method for estimating

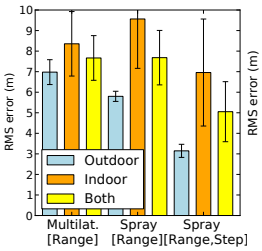


Fig. 8. *Spray* vs. multilateration. *Spray* produces comparable results to the tested multilateration algorithm. Adding a second component to *Spray* improves the estimation performance significantly.

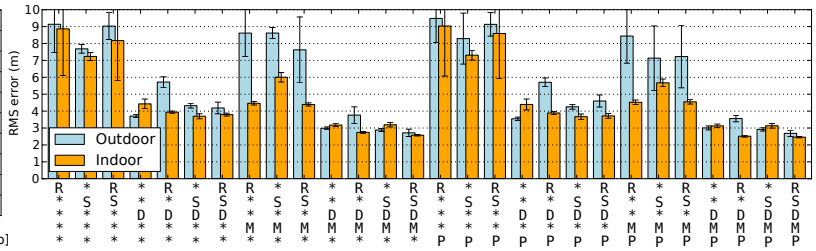


Fig. 9. The contribution of individual components. The letters R, S, D, M, and P denotes the **R**ange, **S**tep counting, **D**ead reckoning, **M**ap, and non-**P**roximity, respectively. For both scenarios, the best combination is the one including all five components. The eight best include both D and M.

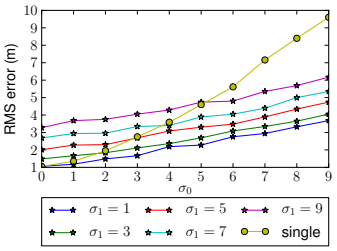


Fig. 10. Simulated results: comparison between one and two modalities for different measurement error distributions.

positions based on range measurements. First, we compare *Spray* to multilateration, using only the ranging component. We do this to show that even without additional information, such as maps or dead reckoning, *Spray* yields comparable results. Then, we incorporate the step counting component to demonstrate the benefits of using multiple modalities. For multilateration, we set up an equation system similar to the one described by Savvides et al. [25], and use the Gauss-Newton method to solve it. The performance of the Gauss-Newton method is sensitive to the choice of initial values (i.e. the initial guess of where nodes are). We use initial positions that are randomly distributed within the range of 5 m from the true position. To prevent the system from oscillating, we use a *shift cutting* factor of 0.1 to rescale the update vector in the Gauss-Newton method.

Using only range measurements, at least three anchors are necessary to perform localization. Therefore, in this experiment, we use three anchor nodes (nodes 0, 2 and 9) for both algorithms in both scenarios. The total errors for *Spray* are computed as described in Section IV-A, and the errors for the multilateration algorithm are computed similarly: We run the algorithm 100 times for each scenario. Each run consist of 2000 iterations to ensure convergence. After each run, we compute the RMS error for the non-anchor nodes in the last iteration. We then take the average of the 100 RMS values, and use the total mean RMS error, together with the corresponding standard deviation. The two leftmost bars in the left graph of Fig. 8 represent the RMS errors for the multilateration algorithm for the outdoor and indoor scenarios, respectively. The third bar is the mean of the first two. The next three bars show the RMS error for *Spray*, using only the ranging component. The two systems perform similar on average. *Spray* performs better in the outdoor scenario, but worse in the indoor scenario. The three rightmost bars in the figure show the RMS error of *Spray* when the step counting component is added. This improves the accuracy for both scenarios, decreasing the mean error with approximately 2.5 m. Although the step counting measurements are similar in nature to ranging measurements, there is no straight forward way to incorporate this information into the multilateration algorithm, because step counting defines a disk as possible locations, in contrast to a circle as defined by the the range measurements.

H. The Contribution of Individual Components

We now turn to investigate the contribution of the individual components. We run all possible combinations of components for both the indoor and outdoor scenarios. In this experiment we use two anchors: node 0, and node 9.

Fig. 9 shows the result of each combination that contains at least one particle generator. That is, there is for example no bar representing only the map component, because it does not have a particle generator, and hence cannot be used alone. In the figure, the letters D, P, M, S, and R, represent **D**ead reckoning, non-**P**roximity, **M**ap, **S**tep count, and **R**anging, respectively. With only two anchors, using only range measurements results in two possible symmetrical solutions. For this reason ranging alone cannot be expected to perform well in this setting. Neither step counting nor non-proximity can help breaking this symmetry. On the other hand, adding directional information, such as dead-reckoning, or eliminating some possible solutions using a map can significantly improve accuracy.

The most important result from this experiment is that the highest estimation accuracy is obtained by combining all components. The results also show that the dead reckoning component is the single most important component. Removing it, always results in errors of over 7 m in the outdoor scenario. In the indoor scenario, however, reasonable accuracy can still be obtained by combining the map and range components. The reason is possibly because of the fine grain information of the map. Moreover, the top eight combinations include both the dead reckoning, and the map component. We also note that including an additional component of lower accuracy, can in some cases actually decrease the performance, and that it is difficult to see a pattern for when this happens. For example, adding range to step counting decreases performance, while adding range to the combination of step counting and map increases performance.

To investigate this, we simulate a scenario in which measurement errors are drawn from the normal distribution with different variances. We simulated 20 nodes deployed according to a random walk. We use three anchors (nodes 0, 10, and 19). Nodes are deployed in numerical order, each node is placed at a uniformly random distance between 10 and 15 m from the previous node, in a uniformly random direction. Figure 10 shows a comparison of the median errors between using one and two modalities. The median is used to decrease the impact of outliers resulting from node topologies, generated by the random walk, which are impossible to solve using three anchors. The line labeled as *single* shows the simulated results when using a single modality that has a zero-mean normally distributed range error, with standard deviation σ_0 according to the x-axes. The other lines show the results from using two modalities with standard deviations σ_0 and σ_1 respectively. The results show that it is possible, to a certain degree, to

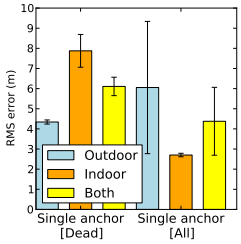


Fig. 11. Estimation errors for localization using a single anchor. Using the dead reckoning component alone, gives lower errors in the outdoor case, while a combination of all components give lower errors both on average, and in the indoor case.

gain performance by including a second modality even if it is of lower accuracy. For example, at $\sigma_0 = 3$, we see that adding a second modality with standard deviation of $\sigma_1 = 5$ or lower, improves accuracy compared to using a single modality. Moreover, using two modalities each with $\sigma_0 = \sigma_1 = 9$ gives a final accuracy comparable of using a single modality with standard deviation $\sigma_0 \approx 6.5$.

I. Single Anchor Localization

We now evaluate the system with a single anchor node (node 0), in the two scenarios. We use two different settings: One with only the dead reckoning component, and the other with all components. The purpose is to see to what extent the performance in the single-anchor case, can be attributed to the dead reckoning component alone. We compare to dead reckoning because it is the only component, of those presented in this paper, that can theoretically be used for localization with a single anchor. The results are shown to the right in Fig. 11. The two scenarios yield very different results: Errors decrease when all components are used in the indoor scenario, but increase in the outdoor scenario. The error, when averaged over both scenarios is, however, lower when all components are used as indicated by the last bar in each group in the figure. That is, without a priori information of how different measurements will perform in a new environment, combining multiple modalities gives the highest expected performance, according to these results. The fact that the performance in the dead reckoning case differ significantly from that reported in Fig. 4 is because in that case a setting equivalent to using $\sigma_r^{gen} = \sigma_\alpha^{gen} = 0$, giving a single point as position estimation. In this experiment, we use the parameter settings resulting from the experiments in Section IV-E. This generates crescent shaped distributions over possible locations, which we believe is typically more desirable than a single point representation, in the targeted applications.

V. Conclusion

In this paper, we presented *Spray*: a multi-modal localization system for WSN deployments of stationary nodes. Our results showed that the position accuracy can be improved significantly in many cases, by combining different localization modalities. The best performance, in both our scenarios, was obtained by combining all components. We have also seen that for *Spray* to perform optimally, the parameter configurations for the components is important. We have shown that for the two scenarios evaluated here, it is possible to find a common such configuration that yields good performance in both cases.

Acknowledgements

This work has been partly performed within the FP7 Evarilos project (grant agreement 317989).

References

[1] N. Ahmed, et al. Detection and tracking using particle-filter-based wireless sensor networks. *Transactions on Mobile Computing* 9, 2010.

[2] I. Amundson and X. D. Koutsoukos. A survey on localization for mobile wireless sensor networks. MELT'09, 2009. Springer-Verlag.

[3] H. Chang, J. Tian, T. Lai, et al. Spinning beacons for precise indoor localization. SenSys '08, 2008. ACM.

[4] L. Girod, et al. The design and implementation of a self-calibrating distributed acoustic sensing platform. SenSys '06, ACM.

[5] L. D. Girod. *A Self-Calibrating System of Distributed Acoustic Arrays*. PhD thesis, University of California, Los Angeles, 2005.

[6] J. Hightower and G. Borriello. Particle filters for location estimation in ubiquitous computing: A case study. Ubicomp, 2004, Springer-Verlag

[7] R. Huang and G. V. Záruba. Location tracking in mobile ad hoc networks using particle filters. J. of Discrete Algorithms, 5(3), 2007.

[8] A. T. Ihler, et al. Nonparametric belief propagation for self-localization of sensor networks. *IEEE J.Sel. A. Commun.*, 23(4), 2006.

[9] M. Kushwaha, K. Molnár, et al. Sensor node localization using mobile acoustic beacons. MASS'05, IEEE, 2005.

[10] B. Kusý. *Spatiotemporal coordination in wireless sensor networks*. PhD thesis, Vanderbilt University, 2007.

[11] A. LaMarca, Y. Chawathe, et al. Place lab: Device positioning using radio beacons in the wild. Pervasive 2005, Springer-Verlag.

[12] A. Lédeczi and M. Maróti. Wireless sensor node localization. *Philosophical Transactions of the Royal Society A*, 370(1958):85–99, 2012.

[13] H. Liu, J. Li, et al. Automatic and robust breadcrumb system deployment for indoor firefighter applications. MobiSyS, 2010, ACM.

[14] E. B. Mazomenos, et al. A two-way time of flight ranging scheme for wireless sensor networks. EWSN'11, 2011, Springer-Verlag.

[15] P. Misra, W. Hu, M. Yang, and S. Jha. Efficient cross-correlation via sparse representation in sensor networks. IPSN '12, 2012. ACM.

[16] M. Mukhopadhyay, et al. Augmentation of anti-jam gps system using smart antenna with a simple doa estimation algorithm. *Progress In Electromagnetics Research*, 67:231–249, 2007.

[17] D. Niculescu and B. Nath. Ad hoc positioning system (aps). GLOBECOM '01, 2001 IEEE.

[18] D. Niculescu and B. Nath. Dv based positioning in ad hoc networks. *Journal of Telecommunication Systems*, 22:267–280, 2003.

[19] G. Oberholzer, et al. Spiderbat: Augmenting wireless sensor networks with distance and angle information. IPSN '11, ACM/IEEE.

[20] P. Pettinato, N. Wirström, J. Eriksson, and T. Voigt. Multi-channel two-way time of flight sensor network ranging. EWSN'12, Springer-Verlag.

[21] N. B. Priyantha. *The Cricket Indoor Location System*. PhD thesis, Massachusetts Institute Of Technology, 2005.

[22] A. Rai, et al. Zee: zero-effort crowdsourcing for indoor localization. Mobicom '12, ACM.

[23] M. Rudafshani and S. Datta. Localization in wireless sensor networks. IPSN '07, 2007. ACM.

[24] A. Savvides, et al. Dynamic fine-grained localization in ad-hoc networks of sensors. MobiCom '01, ACM.

[25] A. Savvides, et al. The bits and flops of the n-hop multilateration primitive for node localization problems. WSNA '02, ACM.

[26] A. Smith, et al. Tracking moving devices with the cricket location system. MobiSys '04, ACM.

[27] A. So, Y. Ye. Theory of semidefinite programming for sensor network localization. SODA '05, SIAM.

[28] R. Stoleru, et al. A high-accuracy, low-cost localization system for wireless sensor networks. SenSys '05, ACM.

[29] C. Taylor, A. Rahimi, et al. Simultaneous localization, calibration, and tracking in an ad hoc sensor network. IPSN '06, 2006. ACM.

[30] C. D. Whitehouse. The design of calamari: an ad-hoc localization system for sensor networks. Master's thesis, UC Berkeley, 2002.

[31] C. D. Whitehouse. *Understanding the Prediction Gap in Multi-hop Localization*. PhD thesis, UC Berkeley, 2006.

[32] G. Zhou, T. He, S. Krishnamurthy, and J. A. Stankovic. Impact of radio irregularity on wireless sensor networks. MobiSys '04, 2004. ACM.