# Automated WLAN Calibration with a Backtracking Particle Filter

Moritz Kessel and Martin Werner
Mobile and Distributed Systems Group
Ludwig-Maximilians-University Munich
Munich, Germany
moritz.kessel@ifi.lmu.de

***Abstract*—Location information is one of the most important information sources in ubiquitous computing scenarios. However, a cheap and global indoor positioning solution offering a sufficiently high accuracy and precision for most ubiquitous computing applications without much calibration effort is not yet available. In this paper we present a backtracking particle filter for sensor fusion of accelerometer, magnetometer and WLAN signal strength measurements on a smartphone, offering high indoor tracking accuracy and precision. Even more, we show that backtracking leads to high quality track information where no WLAN is available. This track information is sufficiently accurate to provide for automated calibration or even the creation of a complete new WLAN fingerprint database. A ground truth position is not needed at all. The particle filter and the WLAN calibration technique are evaluated with high quality ground truth in a test environment at our site and the feasibility of the algorithms is demonstrated.**

***Keywords*-Fingerprinting Calibration, Dead Reckoning, Smartphone Localization.**

## I. Introduction

Location information is maybe the most crucial source of knowledge for ubiquitous computing applications. In outdoor applications location estimates can easily be computed by GPS receivers, since the accuracy of few meters is often sufficient for pervasive services such as navigation. However, this is not the case in indoor environments. GPS signals suffer from attenuation and multipath effects, offering low accuracy or no position capabilities in indoor environments. Especially the GPS error in altitude estimation can lead to localizing a target on a wrong floor which for example renders navigation aids useless. Hence, indoor localization typically relies on other technologies.

In the past years a multitude of technologies and methods for indoor positioning has been proposed which can be divided into two major classes of systems. One class enables highly accurate indoor positioning with high expenses for infrastructure and/or sensors. Ultrasonic, ultra wide band or laser-based systems belong to this class. Those are often deployed in single rooms, small areas, or on few mobile items, where the benefit compensates the expenses. The second class of systems offers a lower accuracy, but relies on existing or cheap infrastructure and/or cheap sensors, and thus can be deployed in a larger area. Application scenarios include indoor location-based services in large buildings such as museums, airports, or hospitals, e.g., pedestrian navigation or asset tracking. Here, the underlying technology is often WLAN, cheap inertial sensors, or a combination of both.

The advantage of WLAN-based positioning is the often existent and, anyway, a quite cheap infrastructure of wireless access points (AP). But the accuracy of few meters requires a time-consuming calibration effort which needs to be carried out beforehand and additionally in the case of infrastructural changes involving the APs or the structure of the building. This effort is avoided when using cheap inertial sensors such as the accelerometer, compass, or gyroscope of a smartphone, but these sensors are quite inaccurate. When used for dead reckoning, the error induced by the sensors grows very quickly. Hence, systems based on inertial sensors need external recalibration techniques such as map matching or WLAN positioning. Furthermore, systems based only on inertial sensors and map matching can only provide a position estimate after some time, so real time positioning systems often rely on additional information sources such as WLAN.

In this paper we combine WLAN-based positioning in form of a fingerprinting approach with two inertial sensors of a smartphone, i.e., digital compass and accelerometer, for accurate real-time indoor positioning. The combination is achieved with a sequential importance resampling (SIR) particle filter, in which the accelerometer and compass readings are used for step detection with step direction determination in the prediction phase of the filter. WLAN signal strength information is the input for weighting the particles in the filter's update phase. Particles crossing walls are eliminated by using an efficient bitmap based map matching algorithm. We show that the particle filter is sufficiently accurate for ubiquitous computing applications, if up-to-date WLAN fingerprint data is available. We also show that without WLAN support, the particle filter converges much slower especially in cases of building symmetries and that over time errors can add up and reduce the accuracy of the filter with increasing tracking time.

In the case when no or only deprecated WLAN fingerprint information is available, we propose an implicit WLAN calibration method based on dead reckoning using the inertial sensors only. We store the track in form of all surviving particles and their parents for backtracking, and thus retrieve high quality tracks of the targets. While, due to backtracking, the quality of a track increases over time, we propose to use

978-1-4673-1954-6/12/$31.00 ©2012 IEEE

the backtrack instead of the track calculated online for implicit WLAN calibration. We show that a database of WLAN fingerprints can be kept up-to-date without the need of personal interaction or feedback and even demonstrate that a completely new fingerprint database can be generated automatically without the high effort of manual calibration. One application scenario of our system is a fingerprint database which is generated and kept up-to-date by the users of the system, carrying out positioning and implicit calibration without user interaction at the same time.

The structure of the paper is as follows. The next section presents related work in the field of indoor positioning, particle filter-based map matching and automated WLAN calibration. Then, in Section III the positioning algorithm is described in detail, also giving a short overview of the statistical background, followed by a detailed description of the algorithm for implicit WLAN calibration. Here, backtracking is introduced and described how it is used for keeping an existing fingerprint database up-to-date as well as for creating a totally new database. Our system was experimentally evaluated in a test environment at our site, which is presented together with the obtained results in Section V. Afterwards, the results are discussed and the paper is concluded.

## II. Related Work

The topic of indoor positioning and tracking is deeply investigated in academic and industrial research and a very active research topic concerning ubiquitous computing. A vast variety of technologies and algorithms have been proposed and still no solution exists that offers satisfactory position information for every use case or scenario.

Many pedestrian indoor positioning systems rely on WLAN fingerprinting algorithms [5], [2], [18], [6] which offer position estimates with sufficient accuracy (i.e., 1-3m) while utilizing the existing WLAN infrastructure and therefore avoiding high expenses. Fingerprinting is a pattern matching method working in two phases: In a calibration phase a fingerprint database is created by the collection of received signal strength indicators (RSSI) from existing access points (AP) at certain reference positions. A fingerprint is the tuple consisting of reference position, eventually a direction, and a measurement vector consisting of RSSI and AP pairs at a specific time/interval. In the second phase (called online or positioning phase), the positioning is carried out by comparing a vector of current measurements consisting also of RSSI and AP pairs with the previously stored values from the database. Some algorithms calculate the position as the reference position of the nearest fingerprint in signal space [2] or the average of the $k$-nearest neighbors (kNN) in signal space [5]. Here one can additionally decide whether a weighted average with the distance in signal space as weight should be computed and whether missing AP information should be ignored or penalized. Other algorithms utilize Bayesian methods [18], [6] based on probability distributions derived by multiple measurements over a length of time. A Gaussian model of RSSI distribution and independence of signal strengths from different APs is often assumed to compute the location of a target based on the probability of observing a certain measurement at different locations. COMPASS [6] addresses the problem of attenuation effects caused by the body of a user. In the calibration phase, fingerprints for several selected directions (typically each $45°$ or $90°$) are collected at reference positions and the users orientation, derived by a digital compass, is stored additionally to the reference position in the fingerprint. In the positioning phase, the user's orientation is also measured and only fingerprints with a similar orientation estimate are used for positioning. Widyawan et al. developed a particle filter for combining signal information from WLAN and wireless sensor networks and, in the absence of inertial sensors, applied a pedestrian movement model in the prediction phase [15]. Liu et al. give a good overview of existing positioning systems based on radio frequency technologies in [9].

Other pedestrian indoor positioning systems calculate the position of a user based on inertial sensors such as accelerometer or gyroscope. Woodman and Harle show in [16] that a building model can compensate the drift of inertial sensors and use a particle filter for a combination of foot mounted accelerometer, gyroscope and map filtering. They show that after some time the user can be accurately localized without initial position information. However, they report problems with building symmetry and add coarse WLAN positioning for obtaining an initial position fix in their dead reckoning system. Evennou and Marx compare a Kalman and a particle filter for fusing location estimates of a WLAN fingerprinting algorithm with accelerometer data [4]. While the particle filter offers a better positioning accuracy the biggest performance gain with respect to accuracy and precision was achieved by the combination of inertial data and WLAN. Widyanwan et al. propose in [14] a backtracking particle filter in a pedestrian dead reckoning system for fusing foot-mounted inertial measurement units (IMU) with a building model, but neither utilize WLAN for positioning nor their system for WLAN calibration. There are also approaches for SLAM (Simultaneous Localization and Mapping) based on inertial sensors only. Robertson et al. present a system called FootSLAM which utilizes a particle filter to learn the accessibility map of a building [12]. The authors utilize only foot-mounted inertial sensors and probabilistic map in form of a grid of hexagons. In this grid, the possible transitions between neighboring cells are learned by observing the probabilities approximated by the particle filter.

In recent years, many indoor positioning systems have been adopted or developed for deployment on cell phones. Smartphones offer many additional sensors which can be integrated with means of sensor fusion algorithms such as Kalman or particle filters to enhance the position accuracy of existing systems. Martin et al. present one of the first WLAN positioning systems, which integrates both calibration and positioning on a mobile phone [10]. They use various nearest neighbor algorithms, but do not use additional sensors. Schäfer et al. demonstrate the usability of a particle filter on a smartphone in real-time with the help of a location model

specifically adjusted for particle filtering [13].

There exist also approaches for automated WLAN fingerprint calibration. Ledlie et al. propose in [7] a crowd sourcing approach with user feedback for WLAN calibration. The system combines positioning with calibration, but relies on an active community which is willing to interact with the system for calibration. Chintalapudi et al. present a genetic algorithm which is able to calculate a WLAN fingerprint database from user data without pre-deployment calibration and without map knowledge or knowing the position of access points [3]. Accuracy and precision of the system are quite high, but at a high calculational effort. Finally, Woodman and Harle propose in [17] to use position estimates calculated by a particle filter combining foot-mounted IMU with a building model for WLAN fingerprint map generation. They show that the system offers a sufficiently accurate database to enable a mesh-based WLAN position correction scheme in their particle filter for reducing the problem of building symmetries. Rai et al. recently developed a similar approach for WLAN calibration [11], also integrating a backtracking particle filter. While the methodology is similar to our approach, they do neither describe how new measurements should influence the creation of a new fingerprint database nor how a deprecated database is brought up to date.

The contribution of our approach is twofold. First, we integrate WLAN fingerprinting with compass and accelerometer data by the means of a particle filter on a smartphone. Furthermore, we make use of the performance gain with respect to accuracy and precision induced by backtracking to automatically and implicitly calibrate a WLAN fingerprint database. For kNN fingerprinting, we give an algorithm for the creation of a new database and the adaption of an existing database with the help of new measurements.

## III. Localization with a Particle Filter

In this section we explain our smartphone-based system for localization and tracking in detail. The localization is done by a SIR particle filter, a sequential Mote Carlo method for implementing a recursive Bayesian filter. Following [1], those filters recursively estimate the state vector of a dynamic system (e.g., the location, velocity and direction of movement of a pedestrian) based on noisy observations stored in a measurement vector. To be able to estimate the state, two different models are necessary. The first describes the evolution of the state with time and will therefore be called prediction model. The second model describes the influence of the measurements on the state estimation and is called measurement or update model in the following.

The prediction model relies on the probability $p(x_k|x_{k-1})$ to end up in a certain state $x_k$ at a discrete point in time $k$ based on the previous state $x_{k-1}$. This probability can be derived by equation (1):

$$x_k = f_{k-1}(x_{k-1}, v_{k-1}) \tag{1}$$

Here, $f_{k-1}$ denotes a function modeling the state transition in time based on the previous state $x_{k-1}$ and some independent identical distributed (i.i.d.) noise $v_k$.

The measurements $z_k$ are related to the state according to the measurement model which is described by equation (2):

$$z_k = h_k(x_k, n_k) \tag{2}$$

Here, $h_k$ is a known function of the state and some i.i.d. measurement noise $n_k$. From equation (2), a likelihood function $p(z_k|x_k)$ can be derived, which represents the probability to retrieve the measurement $z_k$ in a state $x_k$.

Using the Capman-Kolmogorov equation (3), with $z_{1:k} = \{z_1, ..., z_k\}$ being all measurements up to time $k$,

$$p(x_k|z_{1:k-1}) = \int p(x_k|x_{k-1})p(x_{k-1}|z_{1:k-1})dx_{k-1} \tag{3}$$

together with the Markov assumption $p(x_k|x_{k-1}, z_{1:k-1}) = p(x_k|x_{k-1})$ and Bayes theorem

$$p(x_k|z_{1:k}) = \frac{p(z_k|x_k)p(x_k|z_{1:k-1})}{p(z_k|z_{1:k-1})} \tag{4}$$

with the normalizing constant

$$p(z_k|z_{1:k-1}) = \int p(z_k|x_k)p(x_k|z_{1:k-1})dx_k \tag{5}$$

one can calculate a posterior distribution based on the prior and a new measurement $z_k$.

In the case of particle filters, the probability distribution function (pdf) is approximated by a set of weighted samples called particles. These particles $\{x_k^i\}$ and their associated weight $\{w_k^i\}$ are sequentially updated according to equations (3) and (4). The weight $w_k^i$ of a particle $i$ represents the probability $p(x_k = x_k^i|z_{1:k})$ of the target being in the state $x_k^i$. The sum of the weight of all particles is 1, so the state $x_k$ of the system at a discrete point in time $k$ is the weighted sum of the states of all particles $\{x_k^i\}$:

$$x_k = \sum_i^N x_k^i \cdot w_k^i \tag{6}$$

### A. Initialization, Prediction, Resampling, and Update

In the following paragraphs, we describe a particle filter based method for indoor localization. Four different kinds of information are used for our filter: Accelerometer data, digital compass data, WLAN signal strength information, and a bitmap model of the building. The state of the system (and therefore also the state of each particle) consists of two pixel coordinates x and y on a bitmap, the floor number (also standing for the number of the corresponding bitmap), a direction $d$, and a step length $l$.

The particle filter algorithm consists of several phases. Starting with the initialization, where a first set of particles is created, the following phases depend on the incoming data. When a step is detected, the prediction phase is carried out according to the prediction model. Then follows a resampling step, where some particles are removed or splitted. Whenever new signal strength measurements (achieved by an active scan) are available, the update phase is executed recomputing the weight of all particles according to the measurement model.

For initialization of our particle filter, a predefined number $N$ of particles is created according to one of the following initialization schemes: The first scheme uniformly distributes the particles across a certain area. In the second case, all particles are created at the same pixel coordinates and the floor number of the corresponding map. The last scheme distributes the particles according to a two dimensional Gaussian distribution centered at an initial WLAN position estimate with a variance indicating the precision of the estimate. Note that particle weights are initialized to be equal and so the density of particles is also an indicator for the probability of being at a certain position.

The prediction phase depends on step detection, step length estimation, step heading detection, and map information. Step detection is achieved by interpreting the accelerometer readings in a similar way as in [8]. Based on a sliding window of low-pass filtered consecutive accelerometer readings (which is approximately 1s with our smartphones), a step is detected whenever the vertical acceleration drops by more than $2\text{ms}^{-2}$ within the sliding window. Those readings are not considered for step detection twice, so when a step is detected, previous readings are discarded. This mechanism ensures that a single large drop in vertical acceleration measured by consecutive accelerometer readings is not interpreted as multiple steps. Note that in contrast to [8], we use use rotation matrices to calculate vertical component of the acceleration using the phone's orientation in the real world. Whenever a step is detected, a trigger for the particle filter prediction phase is fired. In this phase, each particle $\{x_{k-1}^i\}_{i=1}^N$ is moved in the direction $d_{k-1}$ of the current compass reading (perturbed by a Gaussian random variable $\theta_{k-1}^i$ drawn from $\mathcal{N}_{0,\sigma_\theta}$) for the current step length $l_{k-1}$ (perturbed by a Gaussian random variable $\lambda_{k-1}^i$ drawn from $\mathcal{N}_{0,\sigma_\lambda}$). Equation (7) is the concrete application of equation (1) for each particle:

$$x_k^i = x_{k-1}^i + (l_{k-1} + \lambda_{k-1}^i) \begin{pmatrix} \cos(d_{k-1} + \theta_{k-1}^i) \\ \sin(d_{k-1} + \theta_{k-1}^i) \end{pmatrix} \quad (7)$$

Note that we disturb each particle separately instead of creating a number of new particles for each old particle according to the pdf of the prediction model, because the number of particles would grow exponentially and with a sufficiently high number of particles the posterior pdf will be a good approximation of equation (3). Furthermore, it should be mentioned that the step length is recalculated during positioning from the weighted sum of the step length of each particle. The thought behind is that particles with unlikely step length tend to die sooner or later due to map matching or update from a WLAN position, so even an initially unknown personal step length will after some time be approximated.

In addition to propagating the particles, we apply an efficient bitmap-based map matching. When the path of a particle (implemented as line-painting on the bitmap) includes a non-white pixel, a collision with an obstacle has occurred and the weight of the particle is set to zero. Note that this requires a white representation of walkable space on the bitmap which includes for example the removal of doors or room names possibly depicted on the bitmap.

After the prediction phase, we a resampling algorithm is carried out including the removal of particles with a weight below a probability threshold and the splitting of particles with a large weight to avoid degeneracy problems as described in [1]. Due to real time requirements, we restrict the maximum number of particles to the number of initial particles $N$. Therefore, a particle is split, whenever its weight is at least 50% higher than $N^{-1}$ and the current number of particles is smaller than the maximum number of particles. A particle $x_k^i$ is split in $n$ new particles, where $n = w_k^i N$ is a rounded natural number such that all new particles have a weight below $N^{-1}$. Furthermore, a particle is discarded, when its weight is below $N^{-2}$.

In addition to the map matching, we propose a position correction technique based on WLAN positioning. When using only map matching techniques, one can sometimes observe a multimodal distribution among the particles where the particle cloud is divided into multiple clusters of particles. This effect can be reduced by including additional information generated by a WLAN positioning algorithm which positioning result is independent of the elapsed time since the positioning started (e.g., by a fingerprinting approach). Coarse location information is sufficient for weighting the particles according to a measurement model. We propose a weighted kNN algorithm from which we deduce a probability $p(x_k|z_k)$ as in equation (4). With weighted kNN and a sufficient large number of neighbors $k$, we construct a bivariate Gaussian distribution $\mathcal{N}_{m_x,m_y,\sigma_x,\sigma_y}$ based on the weighted mean $(m_x, m_y)$ of the $k$ nearest neighbors and their standard deviation $(\sigma_x, \sigma_y)$ to that mean. For each particle $x_k^i$, we calculate

$$p(x_k^i|z_k) = \mathcal{N}_{m_x,m_y,\sigma_x,\sigma_y}(x_k^i) \quad (8)$$

and combine this probability with the prior weight $w_{k-1}^i$ of the particle:

$$w_k^i = p(x_k^i|z_k) w_{k-1}^i \quad (9)$$

Finally, we normalize the weight of each particle. Note that the update could also have been achieved by using a Bayesian WLAN positioning system and equation (4) or any other positioning system which can be used to construct a time independent pdf. Since the measurement model is in this case used only for correcting drifts over time, a coarse positioning method is sufficient. Furthermore, the system should also be able to offer reliable WLAN correction in the case when only few signal strength measurements for each fingerprint are available. Since this is the case when we apply the automated calibration, we stick to weighted kNN instead of a Bayesian approach in the measurement model.

Tracking accuracy and precision of this particle filter are described in the evaluation. However, in the described form, it requires WLAN fingerprinting for the measurement model and thus a time consuming calibration phase. In the next section we describe an enhancement called backtracking for our particle filter offering high quality track history that can be used for automated WLAN calibration.

## IV. Backtracking for WLAN Calibration

When using position data for WLAN calibration, the problem arises that the accuracy of the system used for calibration should be better than the capabilities of the algorithm used for WLAN positioning. One could think that this renders the WLAN positioning superfluous, but imagine an expensive positioning system build on top of a moving entity, e.g., a robot. The system is too expensive to be offered to a large number of pedestrians for example in a public building, but can generate high quality fingerprint maps for cheap WLAN positioning. These in turn can be used to offer low cost localization services to other users. In the following, we assume that no training data for WLAN positioning is available even when WLAN signals are received.

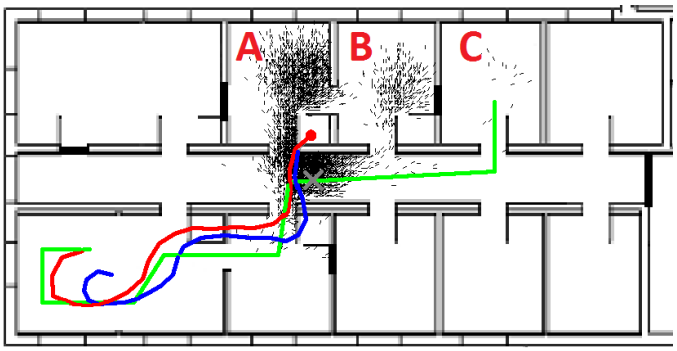


Fig. 1. Ground truth (light green), SIR particle filter track (dark blue), backtracking particle filter track (red), current position (grey cross), and particle distribution during positioning.

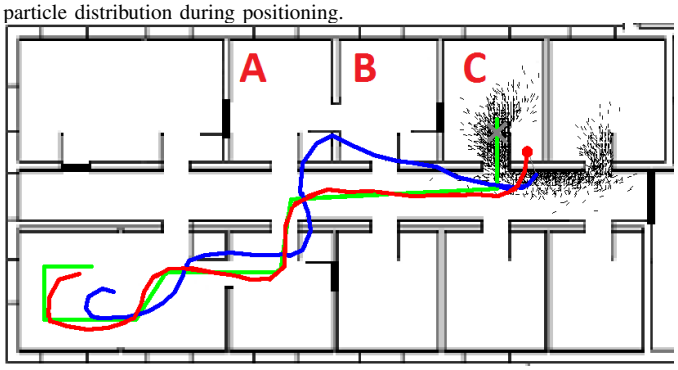


Fig. 2. Ground truth (light green), SIR particle filter track (dark blue), backtracking particle filter track (red), current position (grey cross), and particle distribution some time later than Figure 1.

In our case, we have a quite accurate inertial positioning system which can use additional WLAN information to speed up the convergence of particles and reduce the possibility of a growing error over time in a dead reckoning system. The properties of a particle filter without WLAN correction initialized with a uniform distribution, however, will lead to multiple clusters of particles which might over some time eventually be reduced to a single cluster. The surviving particles have traveled along a path that with a high probability resembles the true path of the tracked entity and thus this path provides a higher accuracy than the particles provided at a specific tracking time. Since a higher accuracy of the calibration system should lead to a higher accuracy of the calibrated system, we apply a technique called backtracking to our particle filter which computes for a given set of particles the trajectory each particle has followed since the start of tracking. Figure 1 and Figure 2 show the advantage of backtracking compared to a common particle filter. The dark blue line depicts the track estimated by the particle filter, including the data of a large cluster of particles dying at a wall in the room A or B (see Figure 1). The red line shows the track updated by the backtracking algorithm, which ignores all dead particles and thus is nearer to the real track in light green (see Figure 2) ending in room C.

In the case of a SIR particle filter, particles with a small weight vanish and particles with a large weight may be split in several particles. To be able to keep record of all preceding particles, we implemented the backtracking technique by introducing a pointer for each particle which refers to the parent particle. To achieve maximal backtracking information, we create a new particle for each current particle in the resampling phase (multiple particles in the case of splitting the current particle). The new particle refers to its parent particle which in turn is removed from the list of active particles. Particles which die have no further children and thus can be discarded completely (including their predecessors). In addition to keeping track of the particles, we store the results of consecutive scans for WLAN signal strength while walking through the building. The measurements can be mapped to a set of particles which represents the state of the system at the measurement time.

During the calibration a WLAN fingerprint database, two possible scenarios need to be distinguished. Either a possibly deprecated database needs to be updated or kept up-to-date or a totally new database needs to be created when no prior information about the WLAN infrastructure is available. In the latter case, one can only rely on other sensors to estimate the position while at the same time scanning actively for WLAN access points. The information gathered from scanning and the position information at the point in time when each scan is finished can be combined to build a reference database. In the first case, an existing database might need to be updated due to changes in environmental information. This way, the existing WLAN fingerprint database can still be used for coarse positioning and new information needs to be introduced to the system slowly to prevent from false calibration due to measurement errors, which could render the whole database useless for positioning. This is a common problem arising in comparable scenarios of unsupervised learning.

We propose to solve this problem statistically by interpolating the new ($\text{rssi}_{\text{new}}$) and the old signal strength information ($\text{rssi}_{\text{old}}$) according to a specific weight $\omega$:

$$\text{rssi} = \omega \cdot \text{rssi}_{\text{new}} + (1 - \omega) \cdot \text{rssi}_{\text{old}} \tag{10}$$

This approach also works for Bayesian fingerprinting techniques, where in addition to the mean value a standard deviation $\sigma$ is stored for the signal strength of each visible access point. Here, the following equation (11) is used for

updating the old standard deviation $\sigma_{\text{old}}$ in the case of new information:

$$\sigma = \sqrt{(1-\omega)\cdot\sigma_{\text{old}}^2 + \omega\cdot(\text{rssi}-\text{rssi}_{\text{new}})^2} \qquad (11)$$

The weight $\omega$ is an indicator how fast the database adopts to changes in the measured signal strength. Signals from new access points are added without weighting. RSS values from APs which were not detected in new measurements are decreased by $(\text{rssi}_{\text{old}} - (-100))/\omega$ and removed, when rssi drops under $-99$ which is below the sensitivity of most smartphones.

In contrast to other systems, we model fingerprints as those points in coordinate space, where the signal strength measurements were recorded. Additionally, we store the orientation of the smartphone to be able to consider the attenuation effect of WLAN signals caused by a user's body in our system. For WLAN calibration, we carry out scanning in the direction of each main axis of the building storing each result as a separate fingerprint. So the question arises which fingerprints should be updated in case of new measurements, or where a fingerprint should be created in the case of creating a new database.

As a simple algorithm we propose to update the nearest fingerprint which is nearer than a predefined distance threshold to the position at measurement time and faces into the direction of the same building axis as the current measurement. If no such fingerprint exists, a new fingerprint is created. This way, the grid size can easily be regulated by setting the distance threshold. In the case of only few new measurements, this method will very slowly adapt the database, so one could also think of updating all surrounding fingerprints.

The proposed algorithm is also applicable in the case of fingerprints being areas, e.g., labeled and trained data for a Naive Bayesian classifier. It presents a way of introducing more weight on current measurements as on older ones, adapting a training database to changes in the environment.

## V. Evaluation

We tested the algorithms in a office environment at our site. All information was gathered with a HTC Desire smartphone. All 64 reference positions for the WLAN fingerprint database were situated on walkable space. At each reference position, four fingerprints were created by scanning for access points and measuring the received signal strength (RSS) while for each single fingerprint the smartphone pointed only in the direction of one of the main axes of the building. This procedure resulted in 256 fingerprints in total (the gray dots in Figure 3).

Furthermore, six tracks were recorded (see Figure 3): While walking along a certain path through the test environment, the readings of the accelerometer and compass were stored together with a timestamp in a file. Both sensors generated readings at a rate of approximately 5Hz. In addition, continuous scans for RSS values were executed (with a rate of approximately 1Hz), enriched with a timestamp of the measurement time, and then also stored in a file. At the same time we saved a timestamp when passing certain reference positions on the path, i.e., at all turns and at the end of the path, which were marked on the ground. This way, high quality ground truth position data was collected.

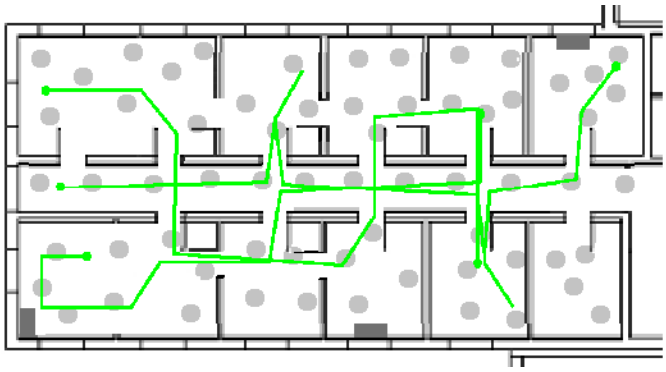

Fig. 3. Reference positions (gray dots) and access points (gray rectangles), as well as ground truth tracks (green) for testing. Each displayed track was recorded starting at the left and at the right side.

The evaluation is structured as follows. First of all, a short evaluation of the components for step detection and step direction estimation is given. Then the positioning performance of the system using all three components in the particle filter together with map matching is given, followed by an evaluation of accuracy and precision of the system without WLAN correction. Finally, the performance gain by utilizing the backtracking approach is evaluated and the feasibility of the algorithm for calibrating WLAN fingerprint maps is demonstrated by comparing the accuracy of our system using a manually calibrated database with the accuracy when using the automated approach.

### A. Step Detection and Step Direction Estimation

The step detection algorithm is one of the most essential parts of our particle filter, since its results highly depend on the quality of the prediction model and in some cases, e.g., for the creation of a new fingerprint database for WLAN positioning, the algorithm will rely only on the prediction model. The algorithm was tested with two different datasets, each consisting of accelerometer readings recorded while walking 50 steps along a corridor. The success rate of the algorithm was 100%, meaning that it correctly recognized 50 steps in each dataset. No matter how good the step detection is actually working, the step length still needs to be estimated inducing an error to our system which needs to be corrected. In our case, this is done by map matching and WLAN correction.

Similar important as step detection is the step heading detection, which is achieved with the help of a digital compass. We tested the accuracy of the orientation information by comparing high quality ground truth track to a track computed solely on the basis of compass data: Starting at the same position as ground truth, we simulated movement with an empirically solved constant speed in the direction of the compass readings. Figure 4 shows a certain delay in the case of abrupt turning, but the overall accuracy of the smartphone's compass is quite good.

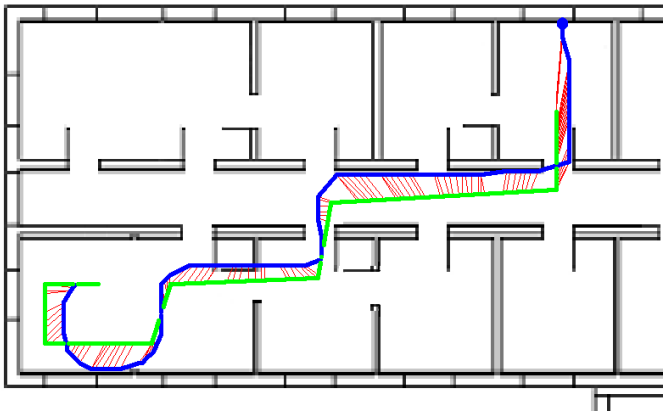

Fig. 4. The real track in light green, the estimated track in dark blue, and error vectors are the thin red lines in between.

### B. SIR Particle Filter Performance

In this section, the overall performance of the system is described in terms of accuracy and precision. The accuracy of the system is given in form of the mean position error to ground truth and the error at the end of the track, while the precision is given in form of the standard deviation. The system was run 50 times with 10000 particles for each single track while utilizing all available information, which was the smartphone's accelerometer and compass, a bitmap, and WLAN signal strength information for comparison with a high resolution fingerprint database. Note that the applied algorithm for WLAN position determination is very coarse and kNN with $k = 10$ was utilized for determination of the parameters in equation (8). Since we use the WLAN positioning only as weighting scheme for ruling out unlikely particles a coarse position method helps to compensate for accumulating errors while at the same time avoiding severe degeneration of the particle cloud. Furthermore, an initial step length of 1.0m was used which is larger as might have been expected for a common pedestrian. This way the strength of the WLAN update phase of the filter becomes obvious, since it offers good accuracies even in case of over- or underestimating the step length. Furthermore, is this a good way to demonstrate the capabilities of the step length adaption mechanism described in Section III-A. For disturbing step length and direction with Gaussian noise, $\sigma_\lambda$ was set to 0.2m and $\sigma_\theta$ was chosen to be $30°$. While the deviation of the step length offers support for different persons with a varying step length, the quite large deviation of the compass compensates for the lag in case of abrupt turns or sudden short time disturbances caused by electromagnetic fields.

Averaged over all tracks, the accuracy of the SIR particle filter was 1.10m and the standard deviation 0.86m. The average error at the end of track was 0.71m, denoting a good accuracy gain (in comparison to the overall average error) due to map matching and WLAN calibration which compensates for inaccurate step length and step heading estimation.

When no WLAN fingerprint information is available, the system has to rely on step detection, step heading detection and map matching alone. Therefore, three other scenarios were investigated where after the three initialization schemes, i.e., a coarse WLAN position, a uniform distribution across the whole area, and a fixed known starting location, only prediction and resampling was carried out. In the case of no WLAN at all, the coarse WLAN position can be seen as an approximation to a similar positioning method, e.g., cellular positioning or Bluetooth proximity detection. Table I shows the results concerning mean error, end error, and standard deviation averaged over all tracks.

TABLE I
OVERVIEW OF TRACKING ACCURACY AND PRECISION

| *Initialization* | Error | Std. deviation | End error |
|---|---|---|---|
| **UNIFORM** | 6.30 m | 3.02 m | 2.30 m |
| **Coarse WLAN** | 3.70 m | 2.06 m | 2.23 m |
| **FIXED** | 1.33 m | 0.78 m | 1.06 m |
| **WLAN correction** | 1.10 m | 0.86 m | 0.71 m |

As can be seen in Table I, the accuracy and precision of all results without WLAN correction stays below the accuracy of the complete system with WLAN correction. Note that the tracks are quite short and the many open doors in the model lead to only few restrictions in the possible movement of particles. This can be seen in the large mean and end error concerning the initialization with a uniform distribution or the coarse WLAN position. The main influence on the end error comes from one single track, where an end error of more than 6m (UNIFORM and Coarse WLAN) could be recognized (compare Figure 5). In the other tracks, the algorithm was able to recognize the true path before the end (see Figure 6 as an example). As expected, the algorithm works best in the case of a known starting position (see FIXED).

Figure 6 and Figure 5 show two problems a positioning system has which supports only dead reckoning. The first problem is known as stabilization or convergence of particles through map matching after an initial uniform distribution. When the initial starting position is not know and a uniform distribution is assumed, movement and movement direction lead after some time to a single cluster of particles. During this period of time, the estimated position as the weighted mean of the particles draws closer to the real position. This can be observed in Figure 6, where the estimated path in blue approaches the green path during positioning. This effect can be delayed or hindered by building symmetries where multiple possibilities of ways correspond to the measured movement. This can be seen in Figure 5 where even a wrong hypothesis has most of the particles weight assembled.

It can be concluded that the algorithm offers highly reliable position information when WLAN correction is applied, the starting location is known, or, in the case when no or only a coarse starting location is known and no WLAN correction is possible, after some movement time through the building until the map matching technique leads to a single cluster of particles. This time depends on the traveled distance, the number and location of turns, and the symmetry of the

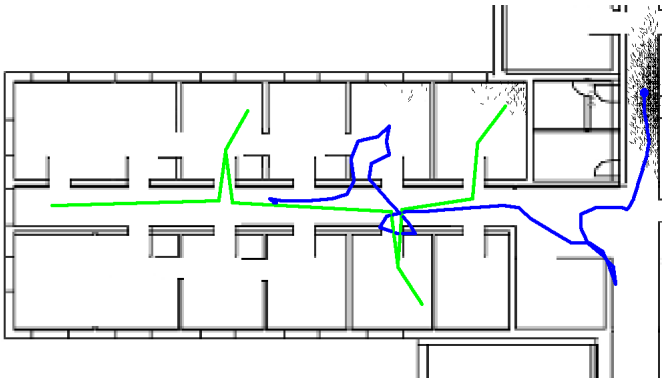

Fig. 5. The real track in light green and the estimated track starting with a uniform distribution in dark blue.

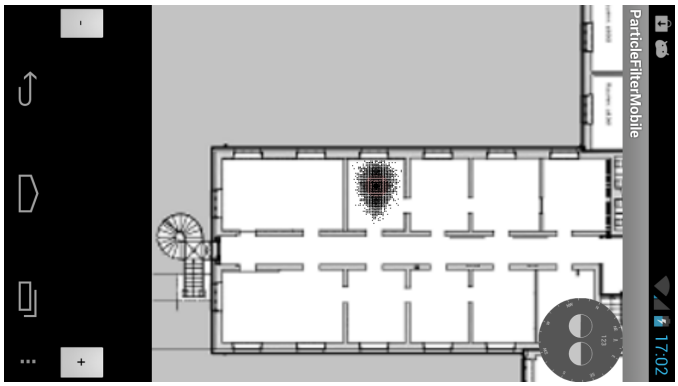


Fig. 7. The Android version of the SIR particle filter.

building. In the case of the test tracks the distance was in between 15 and 30 meter (uniform distribution) or 10 to 20 meter (coarse starting position) and in one track not successful at all with 3 three unbalanced clusters of particles remaining after 42 meter (compare Figure 5).

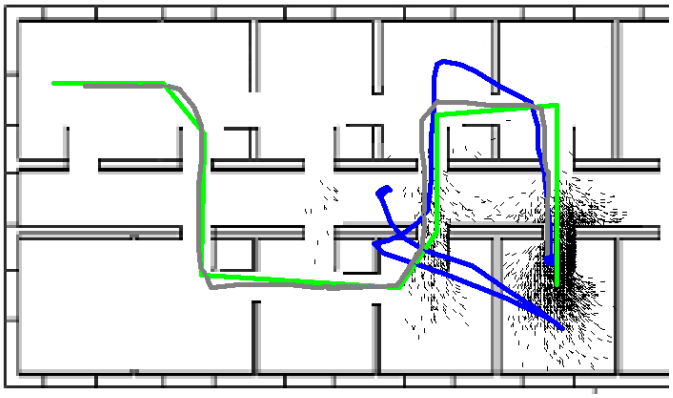

Fig. 6. The real track in light green, the estimated track starting with a uniform distribution in dark blue, and the backtrack in grey.

Before the performance of backtracking is analyzed, the running times of the SIR particle filter need to be given. The system is designed for real time tracking on mobile phones and there also exists a prototypical implementation for Android. However, the evaluation of performance was carried out on a desktop computer for the sake of repeatability. The running time is given as a quotient of the sensing time and the processing time. The sensing time is the time needed to walk along a track and collect measurements while the processing time is the time needed for all calculations. On the desktop environment with a 2.27 GHz processor running on a single core, the quotient was between 0.1 and 0.06 meaning that the precessing was at least 10 times faster than the calculations.

On the mobile phones, no direct time calculation was carried out since the processing steps and thus the load depend on the incoming sensor data. This means for example that when no step is detected no prediction phase needs to be carried out and so no calculations occure. On a Samsung Galaxy Nexus smartphone with Android, real time tracking in a large building was not a problem with up to $10^5$ particles. See Figure 7 for a screenshot of the mobile positioning client.

### C. Performance Gain of Backtracking

For calibrating a WLAN fingerprint database, a highly accurate and precise position system is necessary. Our system satisfies these requirements when WLAN correction is available, the starting location is known, or the user has traveled some time through the building. However, the first item relies on already available and at least somewhat calibrated WLAN fingerprint data rendering calibration dispensable. The second item is seldom found in indoor positioning systems, but could be available in form of a known entry point to the building. In most cases, one has to rely on the user traveling through the building until only one hypothesis (i.e., one cluster of particles) survives. The whole information gathered until that moment can neither be used for calibration nor for any indoor location based services directly.

We explained the concept of backtracking which we apply to solve this problem. We compare the tracking accuracy and precision of the SIR particle filter with the backtracking particle filter. Note, however, that the backtracking takes place after the track has been finished and thus is not applicable as online tracking algorithm. Table II shows the results concerning mean error, end error, and standard deviation averaged over all tracks with one remaining hypothesis at the end (therefore only 5 tracks are included).

TABLE II
OVERVIEW OF BACKTRACKING ACCURACY AND PRECISION

| *Initialization* | Error | Std. dev. |
|---|---|---|
| **UNIFORM** | 1.28 m | 0.62 m |
| **Coarse WLAN** | 1.23 m | 0.46 m |
| **FIXED** | 0.94 m | 0.47 m |
| **WLAN correction** | 0.66 m | 0.38 m |

As can be seen, the mean accuracy is even better than the end accuracy of the SIR particle filter (even when also ignoring the track with multiple hypotheses remaining at the end). Note also the increase of tracking accuracy in the case of available WLAN correction. An example of a backtrack is given in

Figure 6. We consider the accuracy adequate for automated WLAN calibration. Furthermore, the whole track of a user can be handled independently of the initialization method.

### D. Automated WLAN Calibration

The algorithm for automated WLAN calibration is tested in two different scenarios. In the first scenario, an old and outdated WLAN fingerprint database is updated with the help of two calibration tracks, each one covering every room in the test area. The time needed to create these tracks (i.e., walking through every room twice) was about 200 seconds. We then compare the tracking results achieved with the old data with those based on the updated database. Then a completely new database is created also based only on those two tracks. For testing the automated WLAN calibration, we again recorded six tracks following the paths depicted in Figure 3.

The old database was created approximately six months before the new data was recorded. During these months, a wall was removed between two rooms, the ceiling lining was removed in the corridor and some access points changed. Therefore, significant changes in the fingerprint database could be observed as expected. Furthermore, the database had no fingerprints in one room where two paths led through. The mean tracking error of the system using WLAN correction was 3.31m, the mean end error 2.05m, and the mean standard deviation 1.58m, which is far worse than the accuracy of a calibrated system.

We then used the algorithm described before for updating the old database with $\omega = 0.33$. For the update, we initialized both calibration tracks with a uniform distribution and only considered dead reckoning (i.e., prediction and resampling without the measurement phase) and the backtracking algorithm for position estimation. After the calibration, the mean tracking error of the system using WLAN correction was 3.13m, the mean end error 1.67m, and the mean standard deviation 1.48m, which is better than the results on basis of the old database but still worse than the accuracy of a calibrated system. Interestingly enough, variations of $\omega$ in between 0.33 and 0.66 did not result in a much better performance which we assign to the restriction of updating only the nearest fingerprint facing in the same direction. However, the results show that only two calibration tracks with a simple updating mechanism offer an improvement in tracking accuracy and precision. These results let us feel optimistic about using our WLAN calibration scheme in a crowdsourcing solution.

To show the feasibility of our approach, we also created a completely new WLAN fingerprint database using only both calibration tracks and our algorithm for fingerprint creation (see Figure 8). Both tracks were again initialized with a uniform distribution and the particle filter, in absence of any WLAN fingerprint data, only used dead reckoning. Tested with the same six tracks as in the updating case, the system yielded a mean average error of 2.03m, a mean end error of 0.87m, and a mean standard deviation of 1.25m. Especially the small end error shows that WLAN fingerprint data created by our system is sufficiently accurate to serve as calibration means for a particle filter. Table III summarizes the results of WLAN fingerprint calibration. So instead of updating a deprecated database, one should think of automatically generating a completely new one instead, especially if the sole purpose is the stabilization of dead reckoning as in our case.

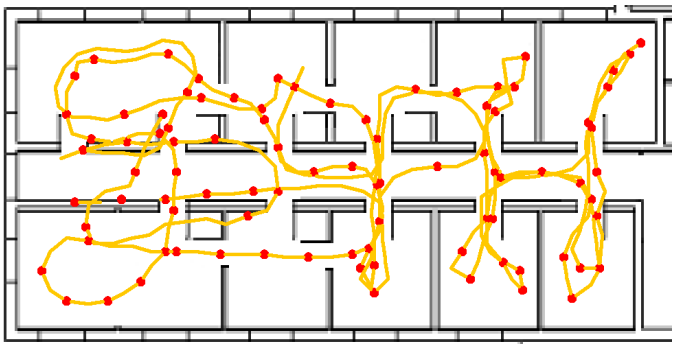

Fig. 8. The result of backtracking both calibration tracks in orange and the created fingerprints in red.

TABLE III
OVERVIEW OF CALIBRATION ACCURACY AND PRECISION

| *Scenario* | Error | Std. dev. | End error |
|---|---|---|---|
| **Deprecated database** | 3.31 m | 1.58 m | 2.05 m |
| **Updated database** | 3.13 m | 1.48 m | 1.67 m |
| **Created database** | 2.03 m | 1.25 m | 0.87 m |

With a little bit more than three minutes work, we automatically created a fingerprint database which offers only slightly worse positioning accuracy as a high quality database created in 2 hours tiresome work. Our approach renders the application of indoor positioning with low cost sensors of a smartphone for large buildings possible.

## VI. DISCUSSION

In the previous section several results concerning the tracking accuracy of the particle filter and the performance of the calibration algorithm were presented. In this section we want to discuss the results, application scenarios, and further considerations.

The main contribution of this paper is the application of a backtracking particle filter for WLAN fingerprint calibration. This is besides non-time-critical tracking scenarios one of the use cases, where no online/real-time location information is needed. Hence, information which is available later on can be used to update the historic state of a system, often leading to better results. In the case of a backtracking particle filter this does not always lead to 100% correct backtracks even in the case of only one surviving particle cluster, because previously separated clusters could have merged while only one of the clusters could have described the real position of the target. In this case additional WLAN correction could add accuracy to the system, but, as in the case of a deprecated database of fingerprints, could also have a negative effect. So one needs to decide whether one includes the WLAN position correction

and the backtracking for WLAN calibration or calculates correction tracks with dead reckoning and backtracking only.

The correction scheme for WLAN fingerprints proposed in this paper depends on frequent calibration measurements. We think of a system, where sensor data is used for both, calculation of a target's position and updating the database at the same time. We plan to evaluate by simulation how many users would be needed to keep such a system up-to-date. Instead of starting with an empty database, one could also think of a modeling approach, where the initial database is calculated from a mathematical model instead of measurements. This model could then be refined by our method.

The results presented in the previous section raise the hope for cheap, easy and accurate pedestrian indoor position. Nevertheless, did the test environment not incorporate large halls, multiple levels, nor many persons causing strong fluctuations in WLAN signal strength. While multiple levels can be handled by including an altimeter to the system, which can already be found in today's smartphones, and/or by applying stair detection in addition to step detection, the feasibility of the approach in other environments as an office building still needs to be examined. Whether WLAN fingerprint calibration in large crowds makes any sense at all should also be considered when thinking of applying our system. Here, switching to AP visibility instead of signal strength could gain a lot.

## VII. Conclusion and Future Work

In this paper we presented a SIR particle filter for fusing accelerometer, magnetometer and WLAN signal strength information on a smartphone for accurate and precise positioning and tracking in indoor environments. Furthermore, we described a backtracking algorithm which is utilized for increasing the accuracy and precision of tracks after recording, thus leading to even higher quality of tracks, especially when no WLAN information or no initial location information is available. The generated high quality tracks are the basis for our proposed automated WLAN fingerprint calibration scheme for keeping existing fingerprint databases up-to-date or creating completely new fingerprint databases. The feasibility of this approach has been demonstrated in a test environment at our site.

The SIR particle filter with a high quality WLAN fingerprint database could achieve a mean positioning error of 1.10m, a end error of 0.71m, and a standard deviation of 0.86m. With the backtracking algorithm, the mean positioning error could even be reduced to 0.66m with a standard deviation of 0.38m. For the WLAN calibration, two tracks without initial position information covering every room in the environment were recorded. Based on the gathered data, a coarse WLAN fingerprint database was automatically created which offered a mean positioning error of 2.03m with a standard deviation of 1.25m and an end error of 0.87m. Considering the time for creation, which was a little bit more than 3 minutes in the case of automated calibration and approximately 2 hours for the high quality database, the results are quite promising.

## References


[1] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, 2002.

[2] P. Bahl and V. N. Padmanabhan. RADAR: An in-building RF-based user location and tracking system. In *Proceedings of the Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'00)*, volume 2, pages 775–784, 2000.

[3] K. Chintalapudi, A. Padmanabha Iyer, and V. N. Padmanabhan. Indoor localization without the pain. In *Proceedings of the 16th International Conference on Mobile Computing and Networking (MobiCom'10)*, pages 173–184, 2010.

[4] F. Evennou and F. Marx. Advanced integration of WIFI and inertial navigation systems for indoor mobile positioning. *EURASIP Journal on Advances in Signal Processing*, 2006:1–11, 2006.

[5] M. Kessel and M. Werner. SMARTPOS: Accurate and precise indoor positioning on mobile phones. In *Proceedings of the International Conference on Mobile Services, Resources, and Users (MOBILITY'11)*, pages 158–163, 2011.

[6] T. King, S. Kopf, T. Haenselmann, C. Lubberger, and W. Effelsberg. COMPASS: A probabilistic indoor positioning system based on 802.11 and digital compasses. In *Proceedings of the International Workshop on Wireless Network Testbeds, Experimental Evaluation and Characterization (WiNTECH'06)*, pages 34–40, 2006.

[7] J. Ledlie, J.-g. Park, D. Curtis, A. Cavalcante, L. Camara, A. Costa, and R. Vieira. Molé: A scalable, user-gnerated WiFi positioning engine. In *Proceedings of the 2nd International Conference on Indoor Positioning and Indoor Navigation (IPIN'11)*, 2011.

[8] J. A. B. Link, P. Smith, N. Viol, and K. Wehrle. FootPath: Accurate map-based indoor navigation using smartphones. In *Proceedings of the 2011 International Conference on Indoor Positioning and Indoor Navigation (IPIN'11)*, 2011.

[9] H. Liu, H. Darabi, P. Banerjee, and J. Liu. Survey of wireless indoor positioning techniques and systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 37(6):1067–1080, 2007.

[10] E. Martin, O. Vinyals, G. Friedland, and R. Bajcsy. Precise indoor localization using smart phones. In *Proceedings of the International Conference on Multimedia (MM'10)*, pages 787–790, 2010.

[11] A. Rai, K. K. Chintalapudi, V. N. Padmanabhan, and R. Sen. Zee: Zero-effort crowdsourcing for indoor localization. In *Proceedings of the 18th International Conference on Mobile Computing and Networking (MobiCom'12)*, pages 293–304, 2012.

[12] P. Robertson, M. Angermann, and B. Krach. Simultaneous localization and mapping for pedestrians using only foot-mounted inertial sensors. In *Proceedings of the 11th International Conference on Ubiquitous Computing (Ubicomp'09)*, pages 93–96, 2009.

[13] M. Schäfer, C. Knapp, and S. Chakraborty. Automatic generation of topological indoor maps for real-time map-based localization and tracking. In *Proceedings of the 2nd International Conference on Indoor Positioning and Indoor Navigation (IPIN'11)*, 2011.

[14] Widyawan, M. Klepal, and S. Beauregard. A backtracking particle filter for fusing building plans with pdr displacement estimates. In *Proceedings of the 5th Workshop on Positioning, Navigation and Communication (WPNC'08)*, pages 207–212, 2008.

[15] Widyawan, M. Klepal, and D. Pesch. A bayesian approach for RF-based indoor localisation. In *Proceedings of the 4th International Symposium on Wireless Communication Systems (ISWCS'07)*, pages 133–137, 2007.

[16] O. Woodman and R. Harle. Pedestrian localisation for indoor environments. In *Proceedings of the International Conference on Ubiquitous Computing (UbiComp'08)*, pages 114–123, 2008.

[17] O. Woodman and R. Harle. RF-based initialisation for inertial pedestrian tracking. In *Proceedings of the 7th International Conference on Pervasive Computing (Pervasive'09)*, pages 238–255, 2009.

[18] M. Youssef and A. Agrawala. The horus WLAN location determination system. In *Proceedings of the International Conference on Mobile Systems, Applications, and Services (MobiSys'05)*, pages 205–218, 2005.