

Indoor occupant positioning system using active RFID deployment and particle filters

Kevin Weekly*, Han Zou†, Lihua Xie†, Qing-Shan Jia‡, and Alexandre M. Bayen*

*Department of Electrical Engineering and Computer Sciences
University of California Berkeley

kweekly@eecs.berkeley.edu, bayen@berkeley.edu

†EXQUISITUS, Centre for E-City, School of Electrical and Electronics Engineering
Nanyang Technological University
{zouh0005,elhxie}@ntu.edu.sg

‡Center for Intelligent and Networked Systems, Department of Automation, TNLIST
Tsinghua University
jiaqs@tsinghua.edu.cn

Abstract—This article describes a method for indoor positioning of human-carried active *Radio Frequency Identification (RFID)* tags based on the *Sampling Importance Resampling (SIR)* particle filtering algorithm. To use particle filtering methods, it is necessary to furnish statistical *state transition* and *observation* distributions. The state transition distribution is obstacle-aware and sampled from a precomputed *accessibility map*. The observation distribution is empirically determined by ground truth RSS measurements while moving the RFID tags along a known trajectory. From this data, we generate estimates of the sensor measurement distributions, grouped by distance, between the tag and sensor. A grid of 24 sensors is deployed in an office environment, measuring *Received Signal Strength (RSS)* from the tags, and a multithreaded program is written to implement the method. We discuss the accuracy of the method using a verification data set collected during a field-operational test.

I. INTRODUCTION

The application area for Indoor Positioning Systems (IPS) is rapidly expanding as researchers discover location-aware services such as on-demand lighting or ventilation control. However, due to the unstructured nature of indoor environments, designing an IPS has many different challenges and requirements in comparison to outdoor and large-scale positioning systems [19]. The requirements of an IPS vary in cost, power usage, accuracy, reliability, and computational efficiency, depending on the application. This has led to many IPS concepts which fill particular niches. For example, a robot needs highly accurate positioning to avoid obstacles, whereas a location-aware service for occupants may only need to know which room an occupant is in. For the former case, a highly accurate ultrasound system [18] might be used. However, in the latter case, power consumption is a greater concern than positioning accuracy, where room-level accuracy might be sufficient.

*This research is funded by the Republic of Singapore's National Research Foundation through a grant to the Berkeley Education Alliance for Research in Singapore (BEARS) for the Singapore-Berkeley Building Efficiency and Sustainability in the Tropics (SinBerBEST) Program. BEARS has been established by the University of California, Berkeley as a center for intellectual excellence in research and education in Singapore.

Advances in low-power radios and improved wireless network protocols have enabled Radio Frequency Identification (RFID) as a practical solution to the occupant tracking problem, with many approaches of how to use radio measurements for positioning [15]. A popular avenue of research is to use existing IEEE 802.11 infrastructure by measuring received signal strength (RSS) to the building's access points [13], [3]. However, many buildings do not have the access point density needed for the IPS to function, and more must be installed. Also, IEEE 802.11 devices inherently have higher power requirements than other technologies, but are considered practical since many occupants already carry and use these types of devices, such as smartphones. Alternatively, IEEE 802.15.4 is a low-cost and low-power wireless sensor network technology which can be deployed for positioning and other low data-rate applications [14], [17]. A device based on this technology can be installed in an employee ID badge, or, in our case, a small fob that the occupant carries.

The approach is to process RSS measurements using a sequential Monte-Carlo Bayesian estimation technique [16], [10], also known as a *particle filter*. We implement the most basic form of the algorithm, called Sampling Importance Resampling (SIR). Particle filtering methods are especially applicable to unstructured environments due to their ability to elegantly handle non-continuous systems and integrate heterogeneous measurements. They have been used successfully in robotics [22], and in IPS installations with a combination of sensors such as RFID, inertial, infrared, and laser scanning [20], [9], [4].

Our IPS strictly uses RSS measurements, although we envision adding other types of sensors to improve the tracking performance. This article is an account of how our IPS is built up from the principles of particle filtering, including practical implementation details one might need if installing an RSS-based IPS system of their own. In particular, we provide a unique method of incorporating the RSS sensor measurements in a way that reduces dependence on a physical transmission model by creating empirical estimates of sensor error distributions. This leads our IPS implementation to achieve within a

5 m accuracy for 90% of estimates. Also, we have found, that our method is well-suited to a multithreaded C++ implementation by achieving 18000-times real-time performance, with memory use of less than 10 MB.

The rest of the article proceeds as follows: In Section II, we introduce the SIR algorithm and describe the modelling decisions we made to incorporate RSS measurements and occupant dynamics. This follows with Section III, where we describe the architecture of RSS measurement collection and programming the IPS. In Section IV, we discuss the results of a field operational test using the deployed IPS, and conclude the article in Section V.

II. INDOOR POSITIONING USING SIR

A. SIR Estimation Framework

In this section, we instantiate the SIR algorithm [10], [22] for our system. For this article, we denote x_t as a three-dimensional random variable corresponding to time t . We use a discrete-time algorithm with a time interval of δt , in seconds. The first two dimensions, $x_{t,1:2}$, are the coordinates, in meters, of the occupant-carried RFID tag being positioned, and the last dimension, $x_{t,3}$, represents an attenuation factor, in dBm, of the tag's local environment. This third term attempts to correct for calibration errors or differences in radio propagation between different tags. To simplify notation when discussing the state in a general context, we use $x_{1:2}$ to represent the coordinates of the tag and x_3 to represent the attenuation factor. We define y_t as the RSS observations measured at time t by the RFID system. This observation vector has N_{sensors} dimensions and measured in units of dBm.

The goal of the algorithm is to estimate the *belief distribution*

$$\Pr(x_t|y_{1:t}), \quad (1)$$

i.e. the probability at time t of state x_t being the actual state of the system, given all of the past observations from $t = 1$. If we estimate the belief perfectly, then the maximum likelihood estimate,

$$\hat{x}_t = \underset{x}{\operatorname{argmax}} \Pr(x|y_{1:t}),$$

is the ideal estimation of the tag's position given all of the previous data.

If we assume a hidden Markov Model, then the current observations only depend on the current state, i.e.

$$\Pr(y_t|x_{1:t}, y_{1:t}) = \Pr(y_t|x_t),$$

and the next state depends only on the last state, i.e.

$$\Pr(x_{t+1}|x_{1:t}) = \Pr(x_{t+1}|x_t).$$

Applying Bayes rule and the above assumptions gives

$$\Pr(x_t|y_{1:(t-1)}) = \int \Pr(x_t|x_{t-1}) \Pr(x_{t-1}|y_{1:(t-1)}) dx_{t-1}, \quad (2)$$

$$\begin{aligned} \Pr(x_t|y_{1:t}) &= \frac{\Pr(y_t|x_t) \Pr(x_t|y_{1:(t-1)})}{\Pr(y_t|y_{1:(t-1)})} \\ &= \eta \Pr(y_t|x_t) \Pr(x_t|y_{1:(t-1)}), \end{aligned} \quad (3)$$

where η is a normalization constant and essentially accounted for by (4).

Thus, (2) and (3), are a recursive solution to (1), and are termed the *prediction* and *update* steps, respectively.

For many problems, including the indoor positioning task, analytically evaluating (2) and (3) is infeasible, therefore we turn to Monte-Carlo methods where samples are used to estimate the distribution.

We track the belief by a set of n particles, where the i -th particle consists of a *state estimate*, $x^{(i)}$, and *importance factor*, $w^{(i)}$. Although not strictly necessary, when the importance factors are updated, they are renormalized so that

$$\sum_{i=1}^n w^{(i)} = 1. \quad (4)$$

The prediction and update steps are adapted to propagate samples and are as follows:

Prediction: For each particle, sample

$$\bar{x}_t^{(i)} \sim \Pr(x_t^{(i)}|x_{t-1}^{(i)}),$$

so that the particles are now distributed as (2). The *state transition distribution*, $\Pr(x_t^{(i)}|x_{t-1}^{(i)})$ represents the system dynamics and is described in Section II-B.

Update: For each particle, evaluate the non-normalized importance weight

$$\bar{w}_t^{(i)} = \Pr(y_t^{(i)}|x_t^{(i)}),$$

where the *observation distribution*, $\Pr(y_t^{(i)}|x_t^{(i)})$, represents the sensor noise and is described by Section II-C. Then, the importance weights are normalized by

$$w_t^{(i)} = \left(\sum_{j=1}^n \bar{w}_t^{(j)} \right)^{-1} \bar{w}_t^{(i)}.$$

We then take n samples, $\{x_t^{(i)} : i = 1, \dots, n\}$, from the discrete distribution defined over $\{\bar{x}_t^{(i)} : i = 1, \dots, n\}$, where $\Pr(x = \bar{x}_t^{(i)}) = w_t^{(i)}$.

After both steps are run, the particles are will be approximately distributed according to the belief (1). Additionally, the pairs of $\bar{x}_t^{(i)}$ and $w_t^{(i)}$ approximate the probability density function (PDF) of the belief, i.e.

$$w_t^{(i)} \approx \Pr(\bar{x}_t^{(i)}|y_{1:t})$$

Two more implementation details remain. Since this is a recursive algorithm, an initial state must be chosen for each of the particles $\{x_1^{(0)}, \dots, x_1^{(n)}\}$, from a given prior distribution

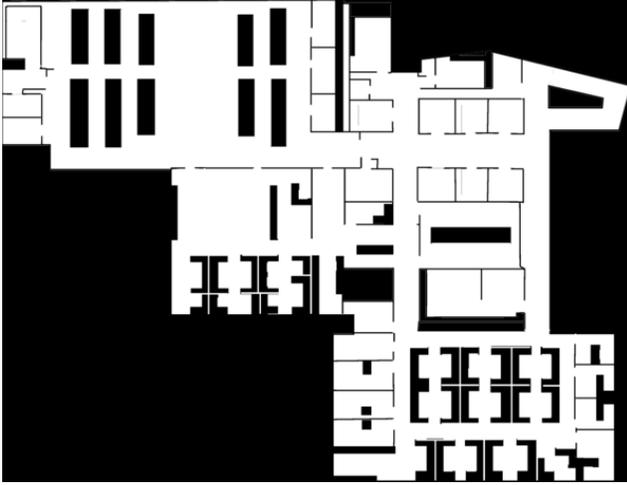


Fig. 1. Obstacle map of the domain. Black areas represent obstacles that occupants cannot enter or pass through.

$f_1(x)$. The initial states are sampled from uniform distributions since little is known about the state of the system before any measurements arrive. For the coordinate states, $x_{1:2}$, the uniform distribution is over the entire floor area of the office space, whereas for the attenuation factor, x_3 , we sample from $\mathcal{U}(-a_{\max}, a_{\max})$, where a_{\max} is a model parameter.

The second implementation detail is the method used to select the actual estimate, \hat{x}_t , of the state from the set of particles. We use the estimate using the *nearest weighted mean particle (NWMP)* method:

$$\begin{aligned} \hat{x}_t &= \bar{x}_t^{(i)}, \\ i &= \operatorname{argmin}_j \|\bar{x}_{t,1:2}^{(j)} - m_t\|_2, \\ m_t &= \frac{1}{n} \sum_{k=1}^n w_t^{(k)} \bar{x}_{t,1:2}^{(k)}. \end{aligned}$$

B. State Transition Distribution

During the update step, there is the need to sample from $\Pr(x_t^{(i)} | x_{t-1}^{(i)})$. The state is comprised of the x - y coordinate pair of the tag, $x_{t,1:2}$, and the attenuation factor, $x_{t,3}$, which are sampled from two separate models.

Tag Coordinates: The state transition distribution essentially adds knowledge about the system dynamics into the estimation. For our problem, this means describing where a tracked occupant could move to (i.e. their future position $x_{t+1,1:2}^{(i)}$) between time t and time $t+1$, given their current position is $x_{t,1:2}^{(i)}$.

Common choices for the transition distribution for robot and occupant tracking problems incorporate the velocity or heading in the state and use kinematic models with additive gaussian white noise [4], [22]. Other, more complex models try to add intuition about human motion [6], or learn the model using machine learning techniques [5].

Our initial approach is to use a circular uniform distribution centered on $x_{t,1:2}^{(i)}$ with radius of $v \cdot \delta t$, where v is an average

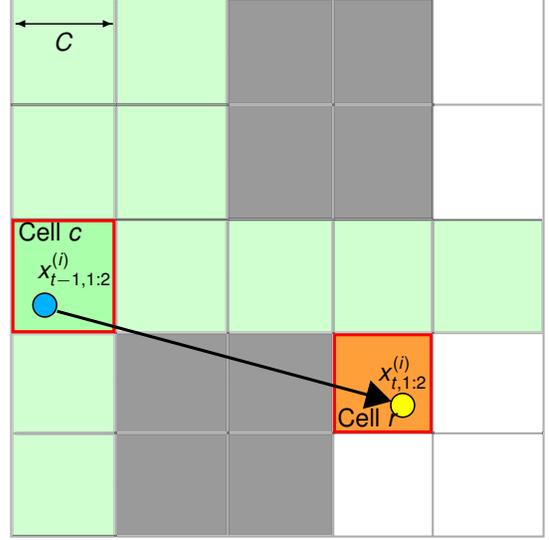


Fig. 2. Illustration of the state transition sampling approach. Grey cells are obstacles, green and orange cells are reachable from cell c within a grid distance of 4. The orange cell r is picked at random from the reachable set. Blue and yellow circles represent the last and current state, respectively.

occupant's walking speed in m/s. This is easy to sample from, but does not incorporate any information about the local environment, such as the fact that occupants will not move into or through obstacles, such as walls and tables.

Therefore, we devise a method which includes reachability information in the transition distribution. We start with an *obstacle map* of the office space, shown in Figure 1, defining the domain of the space and the obstacles, such as walls or desks, which obstruct movement. The domain is discretized into square cells of width C in meters.

Before the IPS program can start, we must precompute a *look-up table (LUT)*. For each cell in the domain, we compute the set of cells reachable within a grid distance of $\lfloor v\delta t C^{-1} \rfloor$. We use depth-limited dynamic programming [8] and a 4-adjacency graph model of the cells to compute these sets.

Although the precomputation step is computationally intensive, it only needs to be performed once if v , C , and δt are constant. The benefit is that we can construct a distribution which is easy to sample from and physically intuitive. Our sampling method is illustrated by Figure 2 for $\lfloor v\delta t C^{-1} \rfloor = 4$. The steps are as follows:

- 1) Find the index of the cell, c , corresponding to the coordinates given by $x_{t-1,2:3}^{(i)}$ (blue circle), shown by the red-bordered and green shaded cell,
- 2) Using the precomputed LUT, find the reachable set, \mathcal{R} , of cells for cell c , shown by the green and orange shaded cells,
- 3) Pick a cell r at random from \mathcal{R} , shown by the orange shaded cell,
- 4) Sample $x_{t,2:3}^{(i)}$ (yellow circle) from a uniform distribution over the points in r .

Other map-matching techniques rely on detecting obstacle crossings during the prediction step and either redrawing sam-

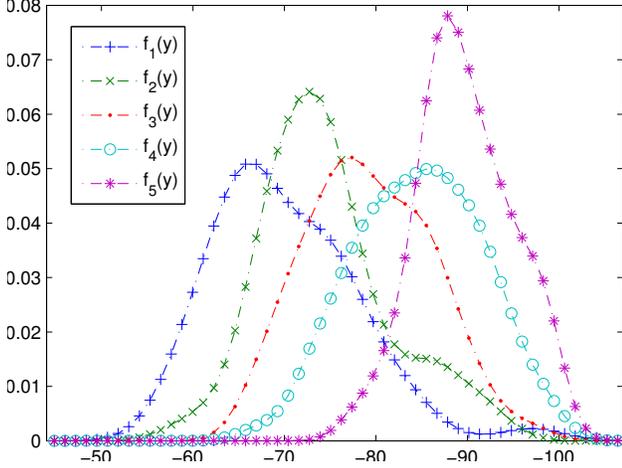


Fig. 3. Example of learned probability distributions for a receiver for $N_{\text{bins}} = 5$ and $d_{\text{max}} = 21$ m.

ples that are within an obstacle or terminating and reinitializing the particle [4]. Our method saves the computational load of detecting and handling obstacle crossings, at the cost of map precision, since it requires discretization. Our method also allows a more accurate representation for large δt , as it allows a particle to travel around an obstacle, provided it is reachable within δt time.

Attenuation Factor: In addition to the position components of the state, $x_{1:2}$, we also describe the state transition distribution of the attenuation factor x_3 by

$$\Pr\left(x_{t,3}^{(i)} \mid x_{t-1,3}^{(i)}\right) = \mathcal{U}\left(x_{t-1,3}^{(i)} - \delta a, x_{t-1,3}^{(i)} + \delta a\right),$$

where δa is a model parameter.

Repositioning: One problem of particle filters is that the particle state estimates converge as more information becomes available and the variance of the belief is reduced. While this seems to be ideal, this means that the particle states cover less and less of the domain, and we lose information about that part of the distribution. Thus, we augment the sampling algorithm with a *repositioning* step:

$$x_t^{(i)} \sim \begin{cases} f_1(x) & z < \Gamma \\ \Pr\left(x_t^{(i)} \mid x_{t-1}^{(i)}\right) & \text{otherwise} \end{cases}$$

$$z \sim \mathcal{U}(0, 1)$$

where $\Gamma \in [0, 1]$ is the *repositioning ratio*, and an algorithm parameter. A $\Gamma > 0$ ensures that some fraction of particles are artificially moved to “explore” parts of the domain that might have been missed.

C. Observation Distribution

It is well known that modelling propagation of radio signals in an unstructured indoor environment is highly complex [11] due to varied reflection and attenuation properties of materials in the space, as well as multi-path effects, which are especially present with narrow-band and high-frequency channels. Therefore, there are a variety of models used for radio positioning

systems such as ours. Some directly derive the model from physical phenomena [3], and others are data-driven models where many measurements are taken at fixed points to create an RSS *fingerprint* [17], [23], [13].

Our method is a data-driven model, but motivated by the physical property that the RSS will, in general, be lower as the sensor is farther from the transmitter. A broad description of the method is that, for each sensor, r , we radially partition the domain into N_{bins} evenly-spaced bins. For each bin, b , there is a corresponding empirically-derived PDF, $f_b^r(\cdot)$, of the signal strength measurement if the sensor is inside the corresponding partition.

To derive the method, we start by assuming the sensor outputs are independent from each other, given the state. Thus, we can decompose the observation distribution by

$$\Pr(y|x) = \prod_{r=1}^{N_{\text{sensors}}} \Pr(y_r|x),$$

where y_r is the RSS reading from the r th receiver. We model the individual sensor distributions as

$$\Pr(y_r|x) = \begin{cases} f_b^r(y_r) + x_3 & b \leq N_{\text{bins}} \\ 0 & \text{otherwise} \end{cases}$$

$$b = \left\lfloor \frac{N_{\text{bins}}}{d_{\text{max}}} \|p_r - x_{1:2}\|_2 \right\rfloor + 1,$$

where p_r is the coordinates of the r th sensor and d_{max} is an algorithm parameter and represents the maximum distance between any sensor and a possible tag’s position.

The distribution $f_b^r(y_r)$ is empirically derived from a ground truth training data set

$$\mathbf{GT} = \{(y_1, g_1), \dots, (y_T, g_T)\},$$

where y_i is an RSS measurement, g_i is the coordinates where the transmitter is located when y_i is taken, and T is the total number of training data points.

The RSS measurements are partitioned into N_{bins} sets

$$\{\mathbf{BIN}_i^r : i \in 1 \dots N_{\text{bins}}\},$$

where

$$\mathbf{BIN}_i^r \triangleq \{y_j : (i-1) \cdot s \leq \|g_j - p_r\|_2 \leq i \cdot s\},$$

and $s = d_{\text{max}}/N_{\text{bins}}$.

Finally, we make a gaussian kernel estimate [21] using the `scipy.stats.gaussian_kde` function of the Scipy [12] distribution. The estimate is sampled at 100 points from -125 dBm to -10 dBm and used as the numerical estimate for $f_b^r(\cdot)$. Figure 3 shows the family of PDFs, $f_b^r(\cdot)$ for $b = \{1, \dots, 5\}$, learned from a 40 min ground truth experiment, for a single sensor.

III. SYSTEM ARCHITECTURE

A. Sensor Placement

A grid of 24 sensors are installed in the false ceiling of a $16.5 \text{ m} \times 13.2 \text{ m}$ (217.8 m^2) office area, above the ceiling tiles, to be hidden from view. The grid configuration is 6 by 4 sensors and the spacing between sensors is 3.3 m and

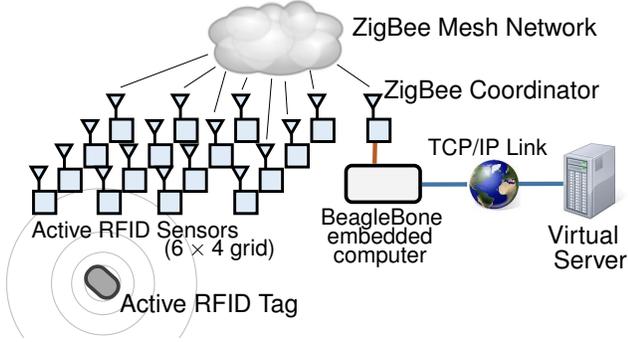


Fig. 4. Network diagram showing data flow from beacons emitted by occupant-carried RFID tags to an Internet server which stores the data and runs the positioning program.

4.4 m, respectively. Power to the sensors is supplied by 5 V low-voltage wires run through the false ceiling, with 4 cable runs powering 6 sensors each. We measured the voltage drop at the end of each cable run to be less than 1 V, even when the system is powered on. Though this means there is only 4 V available to the last sensor, this is not a cause for concern since the sensors internally regulate the input power to 3.3 V.

B. Network Setup

A diagram of the network configuration of our RFID-based IPS system is shown in Figure 4. Our sensor network is constructed of a low-cost network of ZigBee devices based on the TI CC2530 System-on-Chip [23]. There are three types of devices in the network:

- The *tags* (see Figure 5a) are small key-fob devices which are programmed with a unique 16-bit ID and broadcast this ID every second. They are designed to operate continuously for 1 month on a small lithium battery.
- The *sensor* (see Figure 5b) devices are low-cost (approx. 15 USD) and also programmed with a unique 16-bit ID. They form a ZigBee mesh network with the other sensors and coordinator. Upon receiving a beacon from any tag, they note that tag's ID and the RSS value of the beacon, and send it along the ZigBee network to the coordinator. The radios of the sensors must always be listening in order for the system to operate, thus these sensors cannot be battery-powered.
- There is one *coordinator* which initializes the ZigBee network and collects data packets from the sensor nodes. These packets are output in a custom format out of an external serial connection.

Not every beacon from the tags actually arrives at the coordinator due to network collisions or interference. For example, we experienced only a 8-20% successful transmission rate. We believe a higher transmission rate would greatly improve the performance of the IPS method.

A BeagleBone [1] embedded computer is connected to the coordinator and parses the custom format. The BeagleBone connects over the Internet via a TCP/IP socket to a Virtual Private Server (VPS) instance which handles sensor data

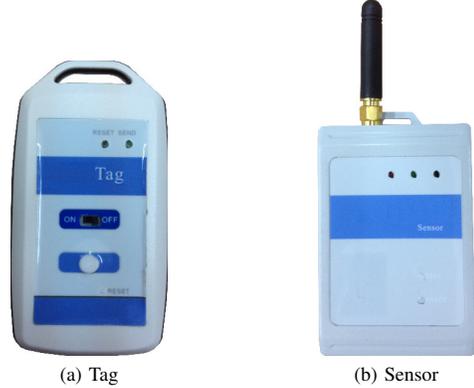


Fig. 5. Active RFID devices used by the IPS sensor network.

storage and processing. Every second the software on-board the BeagleBone reports the RSS measurements received from the sensor network in the last second to the VPS.

The VPS runs an instance of the sMAP [7] server, which efficiently stores the sensor data. The VPS also runs custom software to aid in translating information to and from the sMAP format.

C. Software Setup

Precomputation: The precomputation steps are to compute the reachability LUT for the state transition distribution described in Section II-B, and also to compute the numerical probability distributions from the ground truth data, referred to by Section II-C.

We relied heavily on the SciPy [12] distribution to implement loading and manipulating ground truth data for generating $f_b^T(\cdot)$. Fortunately, despite using a non-compiled language such as Python, an input of 14760 data points is processed in less than 1.7 s.

Although computing the reachability LUT is computationally arduous, a small cell size, C , is desirable to reduce discretization error. Therefore, this part of the algorithm is incorporated into our C++ framework and compiled with a high optimization level. Computing the LUT for 49152 grid cells at $C = 25$ cm, takes less than 30 s, and for the default parameter value of $C = 75$ cm, computing the map takes less than 500 ms.

SIR Algorithm Program: We implement the SIR algorithm as a multithreaded application in C++ and emphasize speed of execution. Our development machine has 4 cores, so we use a pool of 5 worker threads and one coordinator thread, in an attempt to saturate the CPU with workload. We are able to use multiprocessing techniques whenever each particle needs to be independently processed without interaction from others. In the SIR algorithm, this is during the prediction step and while evaluating $\Pr(y_t^{(i)} | x_t^{(i)})$ during the update step. For these tasks, we divide the set of particles into 5 equal subsets and each worker thread is responsible for predicting and updating their respective subset. There is also the need to sort an n -sized array during the update step, when the particles

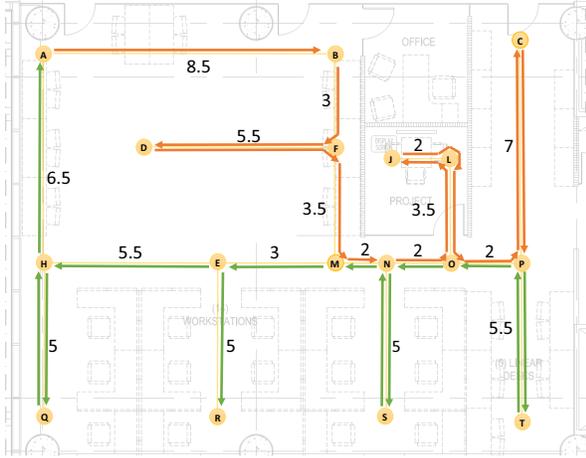


Fig. 6. Ground truth trajectory diagram plotted over the floor plan of the office space. Trajectory starts at anchor point A and all distances are in meters.

TABLE I. LIST OF PARAMETERS TUNED AND THEIR DEFAULTS.

Parameter	Description	Default value
δt	Time interval	5 s
n	Number of particles	100
Γ	Repositioning ratio	0%
a_{\max}	Maximum initial attenuation	1.5 dBm
δa	Maximum change of attenuation	0.75 dBm
C	Reachability LUT cell width	0.75 m
v	Occupant move speed	0.5 m/s
N_{bins}	Number of partitions for the calibration in Section II-C	5

are resampled. At this time, the worker threads are used to sort 5 subarrays and the coordinator thread merges the subarrays into one sorted array. This leads to very fast execution: about 5 hours is simulated per real-time second. Also, because efficient data storage is used, the average memory use is 7.8 MB. Thus, in terms of computational resource requirements, there is much room for more complexity to be added and still maintain real-time performance.

The C++ program also uses Simple Directmedia Layer (SDL) [2] to animate the progress of the particle filter. The visualizer shows the obstacle map, the particle state coordinates, sensor positions, IPS output estimate, and the actual position of the tag, if given. These visualizations assist in debugging the implementation and tuning parameters. An example of 4 steps of the SIR algorithm in progress is shown by Figure 7.

IV. FIELD OPERATIONAL TEST

To evaluate the method, we ran three experiments and collected three corresponding sets of RSS measurements. All of the described experiments consisted of walking once along the trajectory shown in Figure 6.

GT The *Ground Truth* experiment was to collect the learning data needed for the calibration described in Section II-C. Average speed was 5 cm/s, enforced by pausing for 10 s every 50 cm. Three tags were carried for this test, to increase the number of data points collected. The experiment lasted

approximately 40 minutes and collected 14760 (8.5% of sent) RSS measurements.

TUN The *Tuning* experiment was to tune the model parameters to a quicker moving particle than the **GT** data set. The average speed was 50 cm/s and 1106 (19% of sent) RSS measurements were taken.

VER The *Verification* experiment was reserved to evaluate the performance of the method and we do not use the results to tune or adjust the algorithm. The procedure was the same as the **TUN** experiment and 721 (12.5% of sent) RSS measurements were taken.

Using the default parameter values and running the IPS program 1000 times on the **VER** data set, we achieve an accuracy of 50% estimates within 3 m and 90% estimates within 5 m. The mean error is 2.9 m and RMS error is 3.6 m. A histogram of the 144 thousand error samples is plotted in Figure 8. From the histogram we can conclude that, while the mean error is within room-level accuracy, there is still a significant probability of large errors, e.g. 1% probability of error being greater than 9 m.

From [15], we conclude that our system has one or two more meters of error than the best radio-based RSS positioning demonstrations. For instance, LANDMARC [17] achieves an accuracy of 50% estimates within 1 m using a dense deployment of RFID tags, which include static tags. Thus, the direction of future work will focus on improving the reliability of our IPS by adding sensors, such as occupancy sensors, which can constrain particle estimates closer to reality. The particle filtering framework allows for easily including these extra pieces of information into the estimation.

An example of the errors seen during two tests is shown by Figure 9, where the timeseries of coordinates and the error term is plotted. In Figure 10 we plot the true and estimated locations on a map of the space.

Examining these results, our IPS is able to achieve substantially better tracking along the x dimension than the y dimension. This could be a result of there being 6 sensors along the x dimension and spaced 3.3 m apart, rather than 4 sensors spaced 4.4 m apart in the y dimension.

Another contributing factor is that the particle estimates seem to lag behind the true value, especially during fast moves, such as those along the y dimension. The difference is made more clear in Figure 11, where we plot the estimation results using the **GT** dataset, where the occupant was moving much slower and the estimate follows the true position much more closely.

Our motion model does not incorporate velocity or direction of movement, thus, the cloud of particles tends to expand radially outwards, instead of following the true position. An improvement to the algorithm would track the occupant's intended direction, in addition to their position, and favor estimates in that direction. This would likely correct for the lag, but would require a finer time resolution, to avoid overshooting if the occupant suddenly changes direction.

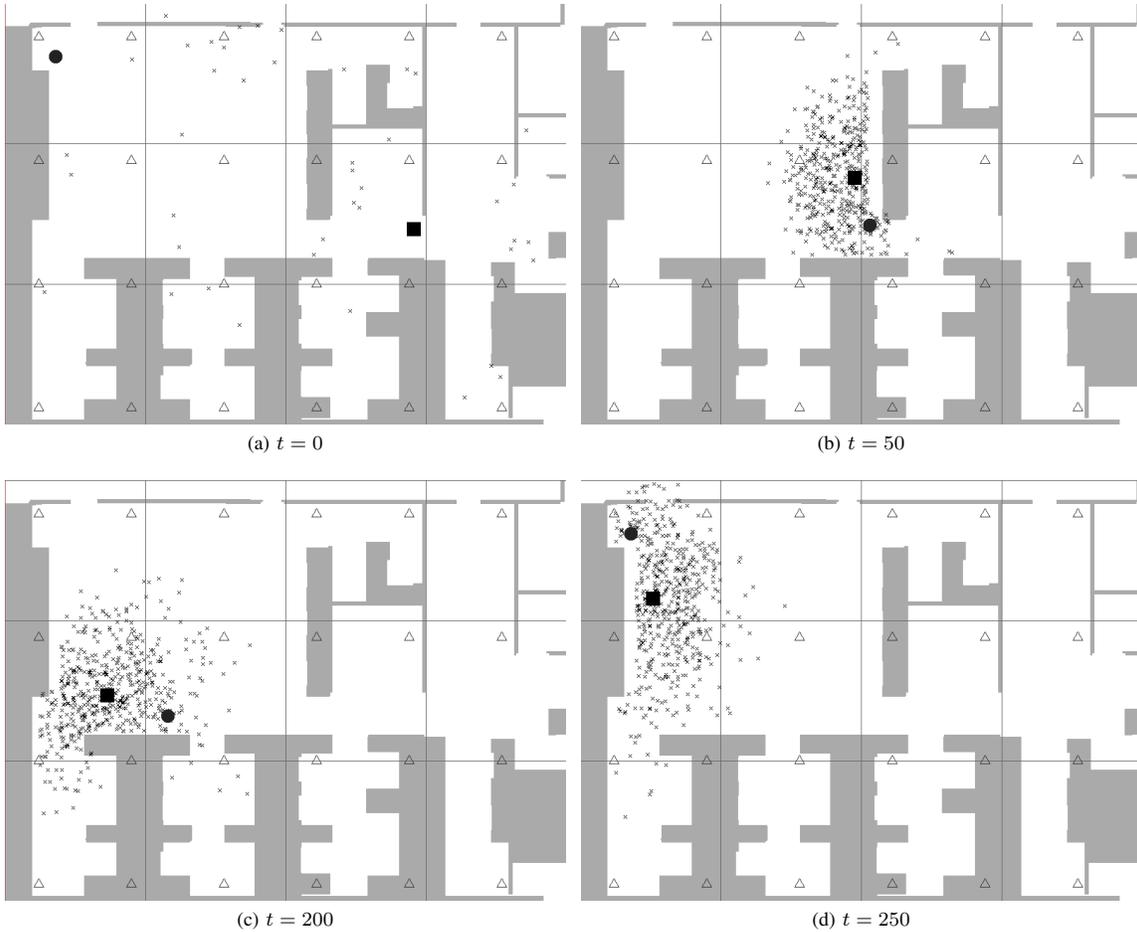


Fig. 7. Intermediate steps of the SIR program visualized. Particle state estimates are small crosses, the NWMP estimate is a square, the actual tag's position is a circle, and triangles give the sensor positions. Grid lines are spaced every 5 m. Parameters for the simulation were set to default values given by Table I, except $n = 600$, for illustrative purposes.

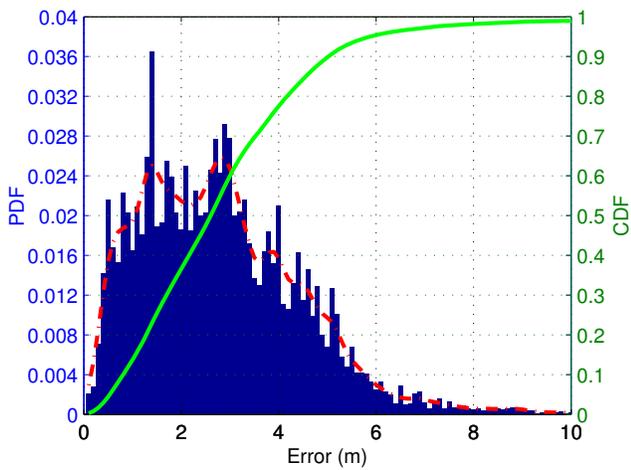


Fig. 8. Histogram of error magnitudes collected over 1000 tests using the **VER** data set. Dotted line shows PDF estimate and solid line shows cumulative distribution function estimate.

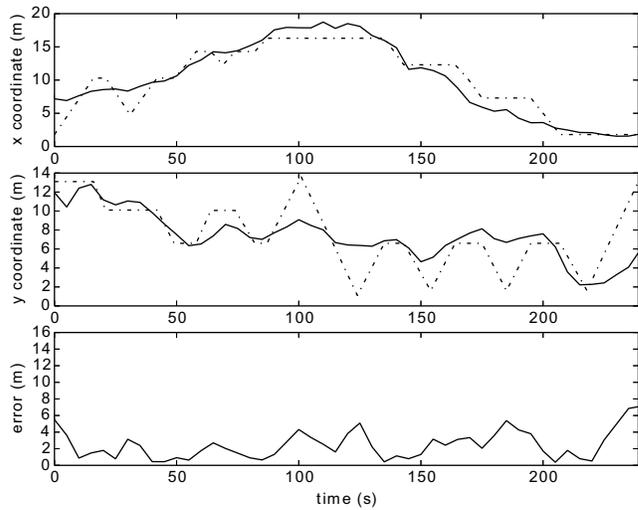


Fig. 9. Example trajectory using default parameter values and **VER** data set. Dotted line is actual trajectory of occupant.

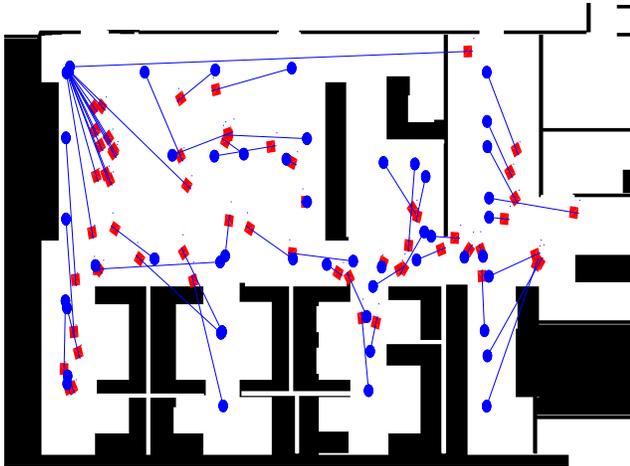


Fig. 10. Plot of actual coordinates (blue circles) and estimated coordinates (red squares) during VER experiment.

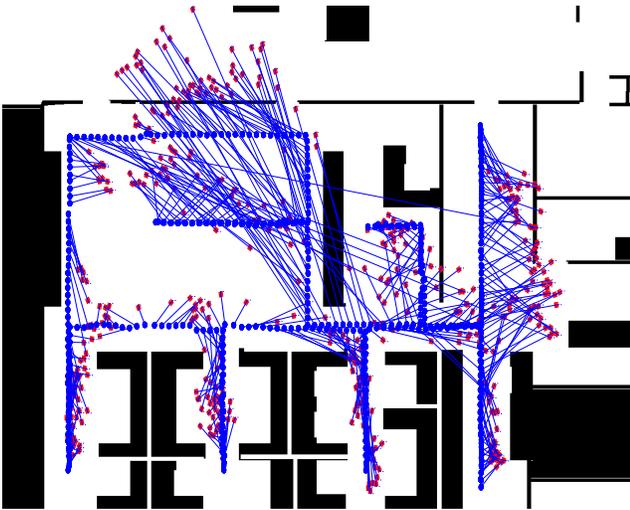


Fig. 11. Plot of actual coordinates (blue circles) and estimated coordinates (red squares) during GT experiment.

V. CONCLUSION

We demonstrate that the proposed IPS system using particle filters achieves an average of room-level accuracy when deployed in a real office environment. Our eventual goal is to track all occupants in the office space (on the order of hundreds), so the low computational requirements of our algorithm is particularly encouraging. Additionally, there are clear avenues for improvement, such as using more complex motion models and incorporating other types of sensors into the system.

REFERENCES

- [1] BeagleBone. <http://beagleboard.org/Products/BeagleBone>, 2013.
- [2] Simple DirectMedia Layer. <http://www.libsdl.org/>, 2013.
- [3] P. Bahl and V. N. Padmanabhan. RADAR: An in-building RF-based user location and tracking system. In *Proceedings of the 19th Joint*

- Conference of the IEEE Computer and Communications Societies*, volume 2, pages 775–784. IEEE, 2000.
- [4] S. Beauregard, Widyawan, and M. Klepal. Indoor PDR performance enhancement using minimal map information and particle filters. In *IEEE/ION Position, Location and Navigation Symposium (PLANS)*, pages 141–147. IEEE, 2008.
- [5] M. Bennewitz, W. Burgard, and S. Thrun. Using EM to learn motion behaviors of persons with mobile robots. In *Proceedings of the Conference on Intelligent Robots and Systems (IROS)*, Lausanne, Switzerland, 2002.
- [6] A. Bruce and G. Gordon. Better motion prediction for people-tracking. In *Proceedings of the IEEE international conference on robotics and automation (ICRA)*, 2004.
- [7] S. Dawson-Haggerty, X. Jiang, G. Tolle, J. Ortiz, and D. Culler. sMAP: a simple measurement and actuation profile for physical information. In *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*, pages 197–210. ACM, 2010.
- [8] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271, 1959.
- [9] F. Evennou and F. Marx. Advanced integration of WiFi and inertial navigation systems for indoor mobile positioning. *EURASIP journal on applied signal processing*, 2006:164–164, 2006.
- [10] N. J. Gordon, D. J. Salmond, and A. F. Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. In *IEE Proceedings F (Radar and Signal Processing)*, volume 140, pages 107–113. IET, 1993.
- [11] H. Hashemi. The indoor radio propagation channel. *Proceedings of the IEEE*, 81(7):943–968, 1993.
- [12] E. Jones, T. Oliphant, P. Peterson, et al. SciPy: Open source scientific tools for Python. <http://www.scipy.org/>, 2001–.
- [13] K. Kaemarungsi and P. Krishnamurthy. Modeling of indoor positioning systems based on location fingerprinting. In *Proceedings of the 23rd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, volume 2, pages 1012–1022. IEEE, 2004.
- [14] S. Lanzisera, D. T. Lin, and K. S. Pister. RF time of flight ranging for wireless sensor network localization. In *International Workshop on Intelligent Solutions in Embedded Systems*, pages 1–12. IEEE, 2006.
- [15] 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.
- [16] S. Maskell and N. Gordon. A tutorial on particle filters for on-line nonlinear/non-gaussian bayesian tracking. In *Target Tracking: Algorithms and Applications (Ref. No. 2001/174)*, IEE, pages 2–1. IET, 2001.
- [17] L. M. Ni, Y. Liu, Y. C. Lau, and A. P. Patil. LANDMARC: indoor location sensing using active RFID. *Wireless networks*, 10(6):701–710, 2004.
- [18] N. B. Priyantha. *The cricket indoor location system*. PhD thesis, Massachusetts Institute of Technology, 2005.
- [19] A. H. Sayed, A. Tarighat, and N. Khajehnouri. Network-based wireless location: challenges faced in developing techniques for accurate wireless location information. *Signal Processing Magazine, IEEE*, 22(4):24–40, 2005.
- [20] D. Schulz, D. Fox, and J. Hightower. People tracking with anonymous and id-sensors using rao-blackwellised particle filters. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 921–928, 2003.
- [21] D. Scott. *Multivariate density estimation*. *Multivariate Density Estimation*, Wiley, New York, 1992, 1, 1992.
- [22] S. Thrun, D. Fox, W. Burgard, and F. Dellaert. Robust monte carlo localization for mobile robots. *Artificial intelligence*, 128(1):99–141, 2001.
- [23] H. Zou, H. Wang, L. Xie, and Q.-S. Jia. An RFID indoor positioning system by using weighted path loss and extreme learning machine. In *Proceedings of the IEEE 1st International Conference on Cyber-Physical Systems, Networks, and Applications (CPSNA)*, pages 66–71, 2013.