Motion Model for Positioning with Graph-Based Indoor Map

Henri NURMINEN, Mike KOIVISTO, Simo ALI-LÖYTTY, and Robert PICHÉ
Tampere University of Technology, Tampere, Finland
Emails: {henri.nurminen, mike.koivisto, simo.ali-loytty, robert.piche}@tut.fi

***Abstract*—This article presents a training-free probabilistic pedestrian motion model that uses indoor map information represented as a set of links that are connected by nodes. This kind of structure can be modelled as a graph. In the proposed model, as a position estimate reaches a link end, the choice probabilities of the next link are proportional to the total link lengths (TLL), the total lengths of the subgraphs accessible by choosing the considered link alternative. The TLLs can be computed off-line using only the graph, and they can be updated if training data are available. A particle filter in which all the particles move on the links following the TLL-based motion model is formulated. The TLL-based motion model has advantageous theoretical properties compared to the conventional models. Furthermore, the real-data WLAN positioning tests show that the positioning accuracy of the algorithm is similar or in many cases better than that of the conventional algorithms. The TLL-based model is found to be advantageous especially if position measurements are used infrequently, with 10-second or more time intervals.**

***Keywords*—indoor positioning; particle filter; motion model; map-matching; graph**

I. Introduction

This paper is concerned with wireless local area network (WLAN) positioning combined with a graph-based floor plan representation. In the graph-based floor plan, the set of expected user paths is condensed into links (edges). The links are undirected line segments that are connected by nodes (vertices) according to their real-world connectivity. It is assumed that the user can be anywhere on the links.

A conventional scheme would be to combine WLAN positioning, pedestrian dead reckoning (PDR) based on e.g. inertial navigation system (INS), and floor plan information in some form [1–6]. However, many consumer-grade mobile devices that one may want to use as indoor positioning devices are equipped with only low-quality INS or no INS at all, which makes reliable PDR challenging or impossible. Absence of PDR emphasises the need for a realistic statistical model for the users motion, since the motion model combines the measurements of different time instants and can, to some extent, be used to maintain location awareness also between WLAN scans. Floor plans can be used to make the motion model more precise and realistic, but their usage typically results in complex and highly nonlinear motion and/or measurement models.

This research was funded by TUT Graduate School, Finnish Doctoral Programme in Computational Sciences (FICS), and HERE, a Nokia business.

The most straightforward and flexible solution to nonlinear estimation problems is particle filters, which are based on Bayesian statistical theory and Monte Carlo (MC) simulation. Particle filters provide a set of weighted MC samples from the posterior distribution of the hidden state at each time instant. These samples are called particles. A practical limitation of particle filters is that they require a precise enough proposal distribution for propagating the particles. If the dispersion of the particle cloud is high compared to the measurement noise variance, the filter requires a large number particles for accurate modelling of the posterior distribution.

A traditional approach in indoor positioning is to use the motion model as a proposal distribution, and then apply wall constraints as a measurement that reduces the weight of wall-colliding particles. Evennou et al. show, however, that if the motion model is only a random-walk model, the required number of particles is huge [7]. Evennou et al. propose using the allowed user paths instead of the prohibited ones, i.e. propagating the particles on the links of the graph. This reduces the degrees of freedom from the particles' movement, which makes the state space easier to be modelled with small numbers of particles.

Furthermore, a graph-based motion model is potentially more realistic than random-walk-based models. Typical pedestrian movement is in a more determined way directed towards some destination than random-walk based models predict. A well-tuned graph-based motion model assumes more continuity for the user's direction than random-walk models, still allowing sharp turns at corridor junctions. The ideas of "destination" and map-based motion modelling have also been proposed for the wall constraint based map representation by Khider et al., but their method requires storing large tables of pre-computed probabilities [8].

The link transition rule plays an important part in making the graph-based model realistic. This paper proposes a novel model for the link transition probabilities, i.e. a rule for distributing the user position's probability from one link to the others. In the proposed model, the probability to choose a link is proportional to the size of the subgraph that is accessible through each link option. This number is called the total link length (TLL). The more potential destinations a direction allows, the more probably this direction is chosen. Extraction of the proposed model's parameters can be done off-line, and it does not require any training data, but the probabilities can be updated if suitable data are available. This

model can straightforwardly be incorporated in a particle filter, which estimates both position and velocity of the user.

Real-data tests are done in a campus building using the existing WLAN infrastructure. The compared methods are particle filters with three different link transition probability rules, the particle filter with wall-collision model, and the Kalman filter. It is found that 400 particles enables reliable positioning with the proposed TLL-based algorithm. Furthermore, the proposed method is found to outperform the comparison methods especially if WLAN scan rate is low.

In section II the proposed link transition model is explained in detail and compared with the methods in the literature. Section III presents the proposed positioning algorithm, and Section IV presents the real-data tests. Section V presents the conclusions.

Notations: $\mathrm{N}(\mathbf{m}, \mathrm{P})$ refers to the (multivariate) normal distribution with mean $\mathbf{m}$ and covariance matrix P, and $\mathrm{N}(\mathbf{x}|\mathbf{m}, \mathrm{P})$ refers to its probability density function (pdf) evaluated at $\mathbf{x}$. $\mathrm{cat}(p_1, \ldots, p_n)$ is the categorical distribution.

II. Graph-based indoor map

A. Related work

The choice of the indoor map's representation depends on the availability of the map and on the requirements of the positioning method. In outdoor vehicle positioning, the node–link model is a natural choice as map representation because the roads usually strictly restrict the movement of the vehicle and such maps are commonly available [9]. In indoor environments, many maps are in the wall representation format. However, e.g. the topological skeleton of the wall representation, sometimes also called the Voronoi diagram, has been proposed to be used as the graph representation [7], and the skeleton of a monochrome digital image can be created automatically [10]. In this paper it is assumed that a graph is available and it is a reliable representation of the building's floor plan.

Indoors, the graph representation is also a crude approximation of the wall representation because in reality a pedestrian can have arbitrary heading and motion patterns. The density of the graph sets a limit for the positioning accuracy especially in large open areas. However, if the measurement accuracy is low compared to the sizes of the geometrical patterns in the building plan, the graph representation removes unnecessary degrees of freedom; in a corridor, a one-dimensional subspace in the middle of the corridor is an adequate approximation when no better accuracy can be expected from the measurement system. Link density should thus be in accordance with the accuracy of the used measurements.

Link transition is the event of a position estimate reaching the end of one link and moving to another. The link transition is modelled with transition probabilities that are assigned to each link connected to the node. In many cases this probability distribution is not uniform. For instance, the user is in general more likely to continue walking on a corridor than to turn to a small room.

It is possible to define some *a priori* rules for the link transition probabilities even without inertial measurements, training data, or knowledge of the functions of different parts of the building. However, only a little discussion of this topic appears in indoor positioning literature. Liao et al. [11] use a uniform link transition prior distribution, but also propose an algorithm for updating the probabilities using a database of measurement sequences. With uniform link transition probabilities, the probability of reaching a certain destination given a starting point is inversely proportional to the number of link branches that could have been chosen during the transition. For a general case, this is counter-intuitive.

In the approach of Yu et al. [12] the distribution is uniform except that links that result in changing the user's room are given lower weights. This method gives somewhat more weight to smooth motion patterns, but for example when a corridor ends to a room, the user might be more likely to enter the room than to turn back. This method cannot model walking in corridors, either. The articles [13, 14] mention the possibility of having non-uniform link transition distribution, but do not propose any automatic rule for setting the probabilities. Maybe a more realistic automatic approach for setting the link transition probabilities has been proposed by Evennou et al. in [7]. In their model the link transition probabilities are based on the continuity of heading: the probability of moving from link λ_i to link λ_j is defined to be

$$\mathbb{P}(\lambda_j|\lambda_i) \propto 1 + \cos(\Delta\theta_{i,j}), \tag{1}$$

where $\Delta\theta_{i,j}$ is angle between links λ_i and λ_j. In an end node of the graph, the user is assumed to make a U-turn. Although this approach models walking in a corridor, for example, in a more realistic way than the uniform distribution model, it still gives lower probabilities to graph areas behind many junctions even if they have similar heading difference histories. Furthermore, this method will fail if there is a sharp turn in the corridor and walking straight takes you into a small room which is visited relatively rarely.

Graph-based indoor maps have been used in assisting indoor positioning also with inertial navigation systems (INS), for example in [6, 15, 16]. In these articles, probability is first propagated according to the INS output, and then projected to the most probable link. Link transition probability is defined by various criteria related to spatial and heading differences. In principle, these articles use the uniform link transition prior probabilities, but the INS measurements give such a strong update to the probabilities that the prior becomes negligible in most cases.

B. Link transition probabilities based on total link lengths

In this subsection, it is assumed that there is a graph-based indoor map. A node ν_m's basic attributes are three-dimensional position $\textsc{pos}(\nu_m)$ that also contains the altitude. A link λ_i has two end nodes $\textsc{endn}(\lambda_i, 1)$ and $\textsc{endn}(\lambda_i, 2)$, whose numbering is arbitrary, and link length $\textsc{length}(\lambda_i)$, which is the Euclidean distance of the end nodes' x and y

coordinates:

$$\text{LENGTH}(\lambda_i) = \|\text{POS}(\text{ENDN}(\lambda_i, 2)) - \text{POS}(\text{ENDN}(\lambda_i, 1))\| .$$

Assume now that the user is in a node ν_m of the map, coming from the link λ_k. Let us denote the options for the new link with $\lambda_{k_1}, \ldots, \lambda_{k_n}$; the old link λ_k is also an option. The motion model determines probabilities for continuing in the direction of each of the links attached to the current node. Without any prior knowledge, the user's destination of travel can be modelled to follow a uniform distribution over the accessible parts of the building. Assuming that each link corresponds to certain constant width, which is an idealisation, this distribution corresponds to a discrete uniform distribution over all the links. Thus, the probabilities $\mathbb{P}(\lambda_{k_i}|\lambda_k)$ are determined by the following principles:

1) All possible link points are equally probable destinations
2) The user moves from the current location to the destination using the shortest possible path

Furthermore, the following simplifications are made to allow efficient practical algorithms:

3) The destination is within some along-graph distance ℓ_{MAX} of the current position
4) Link history is forgotten except for the latest link λ_k

The assumption 4 means that the link history is not used to update the distribution of the destination; it is uniformly distributed also in the areas that could have been reached with a shorter path from an earlier point of the user trajectory.

Principle 1 implies that the probability of the destination being in an arbitrary link λ_i is

$$\mathbb{P}(\text{IS\_DESTINATION}(\lambda_i)) = \frac{[i \in \mathcal{I}_k] \cdot \text{LENGTH}(\lambda_i)}{\sum_{j \in \mathcal{I}_k} \text{LENGTH}(\lambda_j)}, \quad (2)$$

where $[\cdot]$ is the Iverson bracket and $\mathcal{I}_k$ is the set of possible destination link indices. Since the user uses the shortest path and the latest link is in the memory, the user cannot choose to go back to the latest link. Thus,

$$\mathcal{I}_k = \{j | \text{shortest path from } \nu_m \text{ to } \lambda_j \text{ is shorter than } \ell_{\text{MAX}} \text{ and does not use } \lambda_k\}.$$

Furthermore, the principles 2 and 4 imply

$$\mathbb{P}(\lambda_{k_i}|\lambda_k) = \sum_{j \in \mathcal{J}_{k,i}} \mathbb{P}(\text{IS\_DESTINATION}(\lambda_j)), \quad (3)$$

where

$$\mathcal{J}_{k,i} = \{j | \text{shortest path from } \nu_m \text{ to } \lambda_j \text{ uses } \lambda_{k_i}\}.$$

Combining (2) and (3) gives

$$\begin{aligned} &\mathbb{P}(\lambda_{k_i}|\lambda_k) \\ &= \sum_{j \in \mathcal{I}_k \cap \mathcal{J}_{k,i}} \text{LENGTH}(\lambda_j) / \sum_{j \in \mathcal{I}_k} \text{LENGTH}(\lambda_j) \\ &= \frac{[k_i \neq k] \cdot \text{TLL}(\lambda_{k_i}, 1 + [\text{ENDN}(\lambda_{k_i}, 2) = \nu_m])}{\sum_{j=1}^{n} [k_j \neq k] \cdot \text{TLL}(\lambda_{k_j}, 1 + [\text{ENDN}(\lambda_{k_j}, 2) = \nu_m])}, \end{aligned} \quad (4)$$

where $\text{TLL}(\lambda_i, d)$ is the total link length (TLL) behind the link λ_i starting from the end node $\text{ENDN}(\lambda_i, d)$:

$$\text{TLL}(\lambda_i, d) = \sum_{j \in \mathcal{K}_{i,d}} \text{LENGTH}(\lambda_j), \quad (5)$$

where

$$\mathcal{K}_{i,d} = \{j | \text{shortest path from } \text{ENDN}(\lambda_i, d) \text{ to } \lambda_j \text{ is shorter than } \ell_{\text{MAX}} \text{ and uses } \lambda_i\}.$$

Using this definition, the TLLs can be computed efficiently with the off-line algorithm listed in Algorithm 1. The function GRAPHDIST returns the shortest along-link distances of any two nodes of the distance-weighted graph. They can be computed using Dijkstra's algorithm [17, Ch. 4.8]. A link has two TLLs, one for both end nodes.

A special case in this model is the dead end nodes. Because there is in fact no direction choice possibility, the probability of moving beyond a dead end node is reflected backwards in the same link. This U-turn approach is also used in [7].

Note that this scheme does not model static periods or destination shifts. For those, there have to be separate models, especially for modelling stopping and U-turns.

The principle 2 enables fast and straightforward computation of the link transition probabilities. In principle, detours and loops could be taken into account in TLLs with smaller weights, but this is not considered in this paper. Because the history information does not affect the link transition probabilities, detours and loops do in fact have a non-zero probability. The assumption 3 is made to limit the domination of the major passageways so that less probable options can also be modelled by Monte Carlo approximations. In the tests, $\ell_{\text{MAX}} = 40\,\text{m}$ was used. The history forgetting principle 4 is used to allow off-line computation of the probabilities and recursive positioning algorithms.

The presented model generalises straightforwardly to interfloor links, i.e. vertical links connecting different floors at staircases or elevators etc.. In TLL modelling, a vertical link should be given a length higher than the true length to avoid staircases absorbing too much probability.

One possible drawback of the presented model is the requirement that the level of detail is uniform everywhere in the graph-based floor plan. If an area has exceptionally high link density without denser activity in the area, this area gets unjustifiably high probabilities. For instance, if in two similar office rooms one has a link to each desk in the room but the other has only one link leading to the room, the TLL model is not viable with this graph-based floor plan. This is a challenge especially in large open areas. Furthermore, the model may sometimes generate so low or high transition probabilities that their modelling with Monte Carlo algorithms is inefficient. Therefore, some maximum and minimum values should be used for the used transition probabilities.

C. Parameter learning

Liao et al. present an EM (Expectation–Maximisation) algorithm for determining the link transition probabilities using

Algorithm 1 Total link length computation

input: links $\lambda_{1:n_\lambda}$, destination distance ℓ_{MAX}
output: matrix of total link lengths $\mathrm{L} \in \mathbb{R}^{n_\lambda \times 2}$ for each link for both directions

1) Set node index $k := 1$.
2) Compute the vector of the shortest node distances $\delta^{\text{min}} :=$ GRAPHDIST$(\lambda_{1:n_\lambda}, k)$.
3) For each link index $I \in \{i | \lambda_i \text{ connected to } \nu_k\}$, choose direction $d = 1 + [\text{ENDN}(\lambda_I, 2) = \nu_k]$ and compute $\mathrm{L}_{I,d} :=$ TLL_RECURSIVE$(I, d, \lambda_{1:n_\lambda}, \delta^{\text{min}}, \ell_{\text{MAX}}, 0)$.
4) If all nodes handled, stop. Otherwise, set $k := k + 1$ and go to step 2.

function $\ell =$ TLL_RECURSIVE$(I, d, \lambda_{1:n_\lambda}, \delta^{\text{min}}, \ell_{\text{MAX}}, \ell_{\text{used}})$
$\ell :=$ LENGTH(λ_I);
$d' :=$ mod $(d, 2) + 1$;
if $\ell_{\text{used}} + \ell > \delta^{\text{min}}_{\text{ENDN}(\lambda_I, d')}$ OR $\ell_{\text{used}} + \ell > \ell_{\text{MAX}}$ **then**
  $\ell := \min\left\{\frac{1}{2}(\ell + \delta^{\text{min}}_{\text{ENDN}(\lambda_I, d')} - \ell_{\text{used}}), \ell_{\text{MAX}} - \ell_{\text{used}}\right\}$;
  return;
end if
$\ell_{\text{used}} := \ell_{\text{used}} + \ell$;
for all $J \in \{j | j \neq I, \text{ENDN}(\lambda_j, 1) = \text{ENDN}(\lambda_I, d')\}$ **do**
  $\ell := \ell +$ TLL_RECURSIVE$(J, 1, \lambda_{1:n_\lambda}, \delta^{\text{min}}, \ell_{\text{MAX}}, \ell_{\text{used}})$;
end for
for all $J \in \{j | j \neq I, \text{ENDN}(\lambda_j, 2) = \text{ENDN}(\lambda_I, d')\}$ **do**
  $\ell := \ell +$ TLL_RECURSIVE$(J, 2, \lambda_{1:n_\lambda}, \delta^{\text{min}}, \ell_{\text{MAX}}, \ell_{\text{used}})$;
end for
end function

training data [11]. Their method requires iteration of fixed-interval particle smoothing varying the value of the highly multidimensional parameter vector, which is not practical in large-scale systems. It is also not possible to learn and store specific parameters for each user.

In this article it is assumed that the training data contain the information on which direction the users chose after being close to a certain node. Let the link options from the node be $\lambda_1, \lambda_2, \ldots, \lambda_{n_\lambda}$. The number of users that chose the link λ_i is now denoted with y_i, the link transition probability random variable with p_i, and the normalised TLL with π_i. Assuming that the choices of different users are independent, the vector $\mathbf{y} = \begin{bmatrix} y_1 & y_2 & \cdots & y_{n_\lambda} \end{bmatrix}^{\mathrm{T}}$ follows the multinomial distribution

$$p(\mathbf{y}|\mathbf{p}) = \mathrm{lhd}(\mathbf{p}) \propto \prod_{i=1}^{n_\lambda} p_i^{y_i}. \tag{6}$$

The TLL probabilities can be interpreted as a Dirichlet prior

$$p(\mathbf{p}) = \text{Dirichlet}(\mathbf{p}|\boldsymbol{\pi}, a) \propto \prod_{i=1}^{n_\lambda} p_i^{a \cdot \pi_i - 1}, \tag{7}$$

where $a \in \mathbb{R}^+$ is a parameter that determines how many observations the prior corresponds to. This gives the Dirichlet posterior

$$p(\mathbf{p}|\mathbf{y}) = \text{Dirichlet}\left(\mathbf{p}|\begin{bmatrix} \frac{a\pi_1 + y_1}{a+n} & \cdots & \frac{a\pi_{n_\lambda} + y_{n_\lambda}}{a+n} \end{bmatrix}, a + n\right), \tag{8}$$

where $n = \sum_{i=1}^{n_\lambda} y_i$ is the total number of observations. Thus, the update formula for the link transition probabilities is

$$p_i' = \frac{a \cdot p_i + y_i}{a + n}. \tag{9}$$

Further updates can be made using the same formula as more data become available. The U-turn probability can be set to zero in the positioning phase.

III. Positioning algorithm

A. Speed model

Indoor positioning Kalman filters conventionally assume a Gaussian random-walk model for either velocity or position. Either of these is problematic: Velocity is seldom random-walk due to relatively frequent sharp turns and halts. Position is not likely to be random-walk either, since people tend to have transient tendencies in their movement and a position-random-walk does not model these tendencies. However, the graph-based motion models have more potential to take both abrupt changes in direction and tendencies in velocity into account: sharp turns are given probability only at corridor junctions and bends, and when the direction is known, the user's speed can be modelled as random-walk, except for halts.

The motion model described here is similar to the motion model presented by Liao et al. [11], except for the link transition probabilities. A two-mode motion model is used, the modes m_k being random-walk speed combined with the link transition probabilities ('motion'), and static ('static'). The state vector $\mathbf{x}_k$ at time instant t_k is four-dimensional, containing the link index I_k, position on the link $p_k \in [0, 1]$, direction $d_k \in \{1, 2\}$, and speed v_k. Additionally, the state contains the mode m_k. A state vector thus contains all the information needed to calculate the position in Cartesian coordinates as well as the floor. The model for position propagation expressed in probabilistic notation is then

$$p(s_k, v_k | v_{k-1}, m_k) = \begin{cases} \mathrm{N}\left(\begin{bmatrix} s_k \\ v_k \end{bmatrix} | \begin{bmatrix} (\Delta t)_k v_{k-1} \\ v_{k-1} \end{bmatrix}, \mathrm{Q}_k\right) & \text{, if } m_k = \text{'motion'} \\ \text{DIRAC}(s_k) \cdot \text{DIRAC}(v_k) & \text{, if } m_k = \text{'static'} \end{cases}, \tag{10}$$

where s_k is the translated distance within time interval $[t_{k-1}, t_k]$, DIRAC is Dirac delta function, $(\Delta t)_k$ is the length of the discretisation, interval, and

$$\mathrm{Q}_k = \sigma_v^2 \begin{bmatrix} \frac{1}{3}(\Delta t)_k^3 & \frac{1}{2}(\Delta t)_k^2 \\ \frac{1}{2}(\Delta t)_k^2 & (\Delta t)_k \end{bmatrix} \tag{11}$$

is the process noise covariance matrix. The normal distribution should be truncated so that only feasible pedestrian speeds are possible. The distance s_k is not in the state because it only depends on its previous value through the state variable v_k. Together with the link transition probabilities, the distribution of s_k describes how probability propagates on the links. The evolution model of motion mode m_k has two parameters that describe the mode transition probabilities.

B. WLAN positioning

WLAN is the only measurement source used in this paper, in addition to the map. Since the motion model is based on the map, the map cannot give feedback information on the state's distribution, unlike WLAN. WLAN is also the only information channel that can produce a static snapshot estimate of the position; the information contained by the indoor map is mainly useful in a filtering (time-series) context.

WLAN positioning is based on training data ("fingerprints") that are collected from each floor of the building beforehand. It is assumed that the fingerprints contain accurate position. Furthermore, it is assumed that the received signal strength indicator (RSSI) reported by the device can be mapped to the actual received signal strength (RSS) in dBm units so that the measurements of each data collection device are comparable. A method for this is presented in [18], for example.

The WLAN positioning method used in this article's tests is based only on RSS measurements because their usage does not require any hardware modifications and they can be measured by the receiving mobile terminal alone. To average out noise and to keep the number of stored parameters small, the standard logarithmic path loss model is used to model the dependency of the RSS from user position:

$$P = A - 10n \log_{10}(\|\mathbf{r} - \mathbf{r}_{\mathrm{BS}}\|) + w_{\mathrm{PL}}, \tag{12}$$

where P is the measured RSS level, $\mathbf{r}$ is the user's position, $\mathbf{r}_{\mathrm{BS}}$ is the access point's position, and $w_{\mathrm{PL}} \sim \mathrm{N}\left(0, \sigma^2_{\mathrm{PL}}\right)$ is a normally distributed shadowing noise component. The model parameters A and n, and the access point positions are estimated by the Gauss–Newton method as in [19].

The positioning algorithm is another Gauss–Newton method described in [19]. These methods estimate and take into account variances of the path loss parameters and access point positions. Article [19] shows that this approach gives more realistic variance information for the position measurement. Consistent uncertainty estimation is important when combining measurements from different sources: if the used variances are too small, the system might rely too much on possibly erroneous measurements, and if too large, the information is not used efficiently [20].

The true likelihood of one WLAN measurement is not Gaussian, but it is approximated with a Gaussian in the Gauss–Newton method as described in [19]. Thus, the used WLAN measurement model is

$$p(\mathbf{y}_k | I_k, p_k, v_k, m_k) \propto \mathrm{N}\left(\mathbf{r}_k(I_k, p_k) | \mu_k(\mathbf{y}_k), \Sigma_k(\mathbf{y}_k)\right). \tag{13}$$

As Liao et al. point out in [11], the exact likelihood for a link point is the integral of the two-dimensional likelihood over the space that is mapped to the considered link point. However, the implementation of this idea might be computationally infeasible, and it would require using the wall representation of the floor plan also. This is to be avoided at least for mobile applications involving map data transfer via wireless networks.

C. Particle filter

A particle filter is a Monte Carlo algorithm that approximates the posterior distributions $p(\mathbf{x}_k|\mathbf{y}_{1:k})$ provided that certain Markov assumptions hold and the probability distributions $p(\mathbf{x}_0)$, $p(\mathbf{x}_{k+1}|\mathbf{x}_k)$ and $p(\mathbf{y}_k|\mathbf{x}_k)$ are known and their density values are computable for each time index k. The components of the vector $\mathbf{x}$ are the state variables, and the vector $\mathbf{y}$ contains the measurements. No assumptions of linearity or Gaussianity have to be made.

A particle filter approximates the presented model's posterior distributions with a set of weighted particles $\{(\mathbf{x}^i_k, W^i_k) | i \in \{1, \ldots, N\}\}$ that are random realisations of the statistical model [21]. The random samples $\mathbf{x}^i_k$ are generated using the pre-specified proposal distributions, also known as the importance distributions, and the weights W^i_k, also known as importance weights, make the sample set approximate the true posterior in the sense that the moments of the distribution are approximated by the weighted sum over the particle set:

$$\mathrm{E}(\mathbf{g}(\mathbf{x}_k)|\mathbf{y}_{1:k}) \approx \sum_{i=1}^{N} W^i_k \cdot \mathbf{g}(\mathbf{x}^i_k) \tag{14}$$

where $\mathbf{g}$ is an almost arbitrary Borel measurable function.

In the initial phase the particle values are generated from the initial prior $p(\mathbf{x}_0)$ with equal weights. A filter update has two stages: the prediction stage, in which the particles of the current time instant are generated from the proposal distributions $q(\mathbf{x}_k|\mathbf{x}^i_{1:k-1}, \mathbf{y}_{1:k})$, and the update stage, in which the measurements are used to update the weights of the particles. Given the previous time instant's weights W^i_{k-1}, the current time instant's unnormalized weights $\tilde{W}^i_k$ are obtained using the formula

$$\tilde{W}^i_k = \frac{p(\mathbf{y}_k|\mathbf{x}^i_k) p(\mathbf{x}^i_k|\mathbf{x}^i_{k-1})}{q(\mathbf{x}^i_k|\mathbf{x}^i_{1:k-1}, \mathbf{y}_{1:k})} \cdot W^i_{k-1}. \tag{15}$$

The normalization factors of the pdf's $p(\mathbf{x}^i_k|\mathbf{x}^i_{k-1})$ and $q(\mathbf{x}^i_k|\mathbf{x}^i_{1:k-1}, \mathbf{y}_{1:k})$ are not required, since the normalisation constants are approximated rigorously by the ordinary normalisation $W^i_k = \tilde{W}^i_k / \sum_{j=1}^{N} \tilde{W}^j_k$ [22, Ch. 7.2].

The choice of the proposal distribution is a crucial ingredient of any particle filter. A rule of thumb is that the proposal distribution should be as close to the final posterior as possible. In this paper the state transition distribution $p(\mathbf{x}_{k+1}|\mathbf{x}_k)$ is used as a proposal distribution.

A third stage is required by any particle filter to avoid all the weight concentrating to one particle: resampling. At the resampling stage the particles' weights are equalised by sampling with replacement N new particles from the old particle set with the old particle weights as probabilities. One drawback of this resampling method is the danger of sample impoverishment, which means that most of the particles are resampled to one or few locations. This can be avoided by resampling only occasionally, e.g. only when the effective number of particles, $N_{\mathrm{eff},k} = 1/\sum_{i=1}^{N} (W^i_k)^2$, goes below a threshold. Since resampling increases the Monte Carlo

variance of the estimate, the estimate is reported before the resampling. [21, Ch. 3.3]

Particle filters have the drawback that they are unable to detect increase of probability in areas where there are no particles. The proposal distribution should be such that this is improbable. However, getting stuck in a local maximum is possible, and this risk is exacerbated by the floor plan model, which may make points that are close in Euclidean sense distant. Map errors such as missing links may have similar effect, since the graph-based particle filter cannot model map errors, unlike the wall-collision particle filter presented in [5]. To recover from filter divergences, the divergence monitoring and re-initialisation procedures proposed in [5] are used. After the re-initialised particles have been generated from the Gaussian distribution, they are moved to the closest link points.

With the model of this article, the prediction stage updates the particle values with the graph-based motion model and randomly generated noise modifying the weights of wall-crossing particles, and the update stage corrects the particle weights based on the WLAN measurement. In case a WLAN scan is not made at some time instant, the update stage is simply omitted. The algorithm description is in Algorithm 2.

D. Point estimator

It would be convenient for the end user that the reported point estimate of the position would be in a feasible pedestrian area. The links of the graph-based floor plan provide a definition for the feasible area because they should contain the most likely locations. However, the weighted mean of the particles, which minimises the weighted sum of squares, can even be outside the building if the distribution is multimodal or L-shaped, for example.

The Constrained Mean algorithm proposed in [23] can be used to enforce the reported position to be in one of the links. The used estimator is the minimiser of the weighted sum of squares in the set of the link positions:

$$\hat{\mu}_k = \arg\min_{\boldsymbol{\xi} \in C} \sum_{i=1}^{N} W_k^i \left\| \mathbf{r}_k^i - \boldsymbol{\xi} \right\|^2, \tag{16}$$

where C is the set of all link positions. A straightforward derivation shows that the weighted sum of squares function is a strictly increasing function of the distance to the ordinary weighted mean. Thus, the Constrained Mean estimator for the graph-based model equals choosing the link point that is closest to the weighted mean. The detailed description of the used point estimator is in Algorithm 3.

Algorithm 2 Particle filter for 2D indoor positioning with graph-based indoor map

1) For each $i \in \{1, \ldots, N\}$ set $W_0^i := \frac{1}{N}$, and generate $\mathbf{x}_0^i \leftarrow p(\mathbf{x}_0)$. Set the time index $k := 1$.
2) Generate motion states m_k^i based on the previous motion states m_{k-1}^i and the transition probabilities. For the particles $i \in \{i | m_{k-1}^i = \text{'static'} \text{ and } m_k^i = \text{'motion'}\}$ generate $v_{k-1}^i \leftarrow p(v_o)$ and $d_{k-1}^i \leftarrow \mathrm{cat}(0.5, 0.5)$. For each $i \in \{i | m_k^i = \text{'motion'}\}$ generate
$$\begin{bmatrix} s_k^i \\ v_k^i \end{bmatrix} \leftarrow \mathrm{N}\left(\begin{bmatrix} (\Delta t)_k v_{k-1}^i \\ v_{k-1}^i \end{bmatrix}, \mathrm{Q}_{k-1} \right),$$
set $v_k^i := \min(\max(v_k^i, v_{\min}), v_{\max})$ and set $d_k^i := d_{k-1}^i$. The matrix Q_{k-1} is the one in Eq. (11). For each $i \in \{i | m_k^i = \text{'static'}\}$ set $s_k^i := 0$ and $v_k^i := 0$.
3) For each $i \in \{1, \ldots, N\}$ move the particle to the direction d_k^i at most to the end of the link $\lambda_{I_{k-1}^i}$ to distance s' no more than s_k^i. Set $\tilde{s} := s_k^i - s'$ and $I_k^i = I_{k-1}^i$, and compute
   while $\tilde{s} > 0$ **do**
      Generate the new link index I_k^i from the categorical distribution $\mathrm{cat}(\mathbb{P}(\lambda_{k_1} | \lambda_{I_k^i}), \ldots, \mathbb{P}(\lambda_{k_{n_\lambda}} | \lambda_{I_k^i}))$
      Update the direction d_k^i
      Move particle i to the direction d_k^i at most to the end of the current link to distance s' no more than $\tilde{s}$
      Set $\tilde{s} := \tilde{s} - s'$
   end while
   Compute the particle's position $\mathbf{r}_k^i$ in the Cartesian coord..
4) Set $\tilde{W}_k^i := W_{k-1}^i$ for each $i \in \{1, \ldots, N\}$. Perform divergence monitoring. If re-initialised, go to phase 6.
5) If no WLAN measurement is obtained at time index k, go to Phase 6. Otherwise, run the Gauss–Newton algorithm to obtain the WLAN estimate's mean $\boldsymbol{\mu}_k$ and covariance matrix Σ_k . Set
$$\tilde{W}_k^i := \mathrm{N}\left(\mathbf{r}_k^i | \boldsymbol{\mu}_k, \Sigma_k\right) \cdot \tilde{W}_k^i,$$
and normalize by $W_k^i := \tilde{W}_k^i / \sum_{j=1}^N \tilde{W}_k^j$.
6) Compute the reported estimate $\hat{\boldsymbol{\mu}}_k$ using Algorithm 3.
7) If $1/\sum_{i=1}^N (W_k^i)^2 < N_{\text{eff,lim}}$, perform resampling and equalise the weights.

IV. Testing

A. Test setup

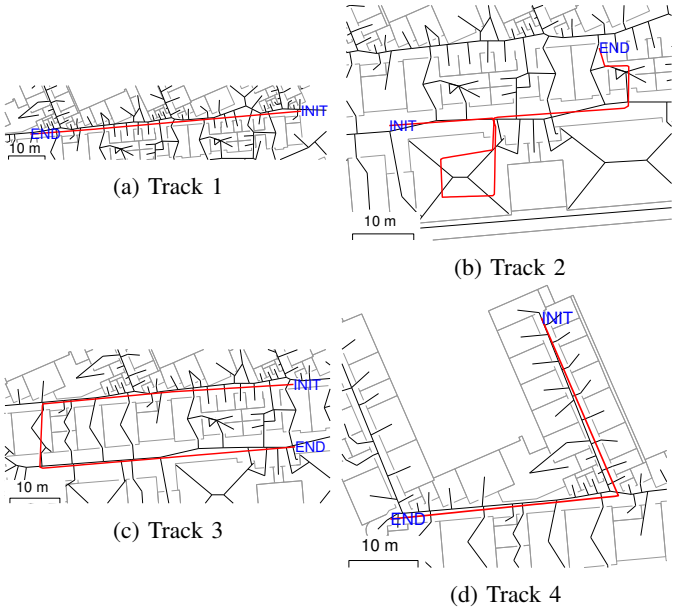
The experimental tests are carried out in the building Tietotalo of Tampere University of Technology. Only the existing communication WLAN infrastructure is used. A Nexus 7 tablet with Android 4.4.2 OS is used to log the WLAN measurements. All training and positioning data are collected with the same device. The reference locations are set manually by the user by tapping a floor plan figure on the tablet's screen. Positioning algorithms are computed with MATLAB. The filters are updated and errors computed with 0.5 s intervals.

The test tracks are shown in Fig. 1. Track 1 in Fig. 1a is a straight corridor that has offices and classrooms around. There are also some rooms that have access to another long corridor, as well as some branching corridors. Track 2 in Fig. 1b contains two visits to smaller rooms around the corridor. Track 3 in Fig. 1c contains two long corridors and a room that has access to both corridors. Track 4 in Fig. 1d uses only main corridors but makes a 90-degree turn. Additionally, a longer test track that combines all the shorter tracks is collected for evaluating the required number of particles.

Algorithm 3 Point estimate computation
(Link-constrained weighted least squares)

Input: particle positions $\mathbf{r}^i$ with weights W^i, $i \in \{1, \ldots, N\}$
Output: point estimate $\hat{\boldsymbol{\mu}}$.

1) Compute the weighted mean $\hat{\boldsymbol{\mu}}' := \sum_{i=1}^{N} W^i \cdot \mathbf{r}^i$.
2) For each interesting link j (e.g. all links of the building part), find the link point $\mathbf{z}_j$ that is closest to $\hat{\boldsymbol{\mu}}'$. If the orthogonal projection of $\hat{\boldsymbol{\mu}}'$ to the line corresponding to the link is between the end points of the link, choose this point. Otherwise, choose the closest end point.
3) Set $s_j := \|\mathbf{z}_j - \hat{\boldsymbol{\mu}}'\|$ for each interesting link j.
4) Set $j^* := \arg\min_j s_j$ and $\hat{\boldsymbol{\mu}} := \mathbf{z}_{j^*}$.



(a) Track 1 (b) Track 2 (c) Track 3 (d) Track 4

Figure 1. The test tracks. Dim colour represents the walls, black colour the links, and red colour the ground truth. Initial points and end points are labelled.

The compared methods are three graph-based particle filters, the wall-collision particle filter (WCPF), and the Kalman filter (KF). The particle filters use the uniform link transition model, link angle difference based model (Evennou et al.), and the proposed total link length (TLL) model for determining the link transition probabilities. The WCPF and the KF use the random-walk position model. In the WCPF the particles that collide with walls are given zero weight, and the quality monitoring as well as re-initialisation are done similarly as in the other particle filters.

B. Results and discussion

To evaluate Monte Carlo errors and to find a suitable number of particles, the filter is run repeatedly for the long test track. The box plot of mean errors of 100 replications are presented in Fig. 2. The different levels of the boxes represent 5 %, 25 %, 50 %, 75 %, and 95 % empirical quantiles. The figure shows that the proposed filter performs reasonably well already with small numbers of particles, such as 20 or 50 with a 5-second measurement interval. After 400 particles the performance

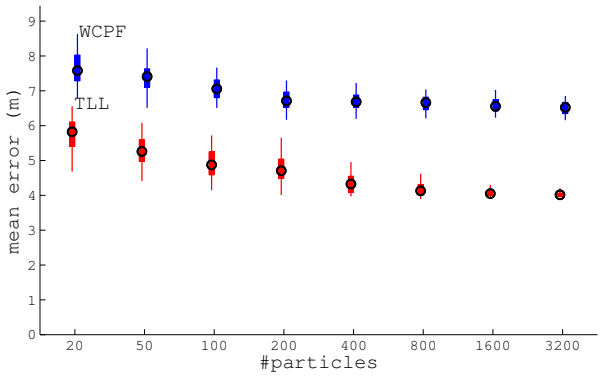


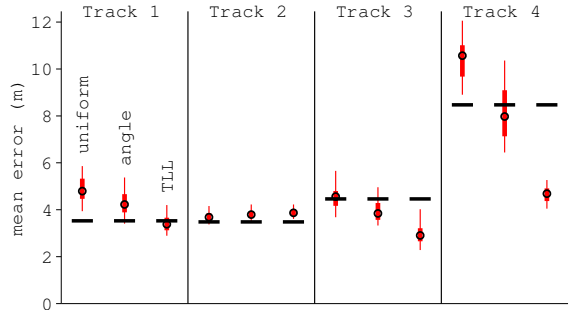
Figure 2. Mean error distribution as a function of particle number for the graph-based total link length particle filter (TLL) and for the wall collision particle filter (WCPF). WLAN scan interval is 5 s.

does not improve significantly. The WCPF also achieves stable performance with 400 particles, but it is clearly worse than that of the TLL-based method.

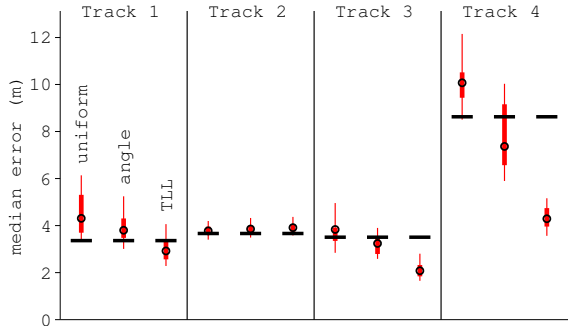
In the actual positioning test, each particle filter is run 100 times with 400 particles for each of the four test tracks. Figure 3 compares the obtained error distributions. The black horizontal lines show the KF's errors. The different subfigures represent different error statistics: mean errors, median errors, 95 % quantiles of errors, and room detection rates.

In track 1, the straight corridor, the proposed TLL method performs slightly better in positioning errors than the other particle filters. This can be explained by the fact that in the TLL method most particles typically continue on the corridor links, which have high TLLs. Thus, the TLL-based filter infers the motion pattern that happens to be correct for this case. The other particle filters, especially the uniform distribution method, distribute large proportions of particles to branching links, and thus their motion models are closer to that of the KF. In fact, their performances are slightly worse than the KF's, which might be explained by the Monte Carlo errors, i.e. low particle densities or poor coverage in interesting state space areas. Track 2 is not expected to favour the TLL method, since it contains two visits to low-TLL branches and lacks stationary motion patterns. However, in the test results the TLL method is still only slightly worse than the others for this track. In track 3, the TLL method outperforms especially the KF. This might reflect the fact that the motion pattern inferred by the TLL method, the speed estimate, is still usable after the 90-degree turns. The KF, in contrast, cannot infer any velocity information. The performance of the TLL method is superior also in track 4. This shows that the TLL methods performs well in corridors even if the corridor is branched or has turns.

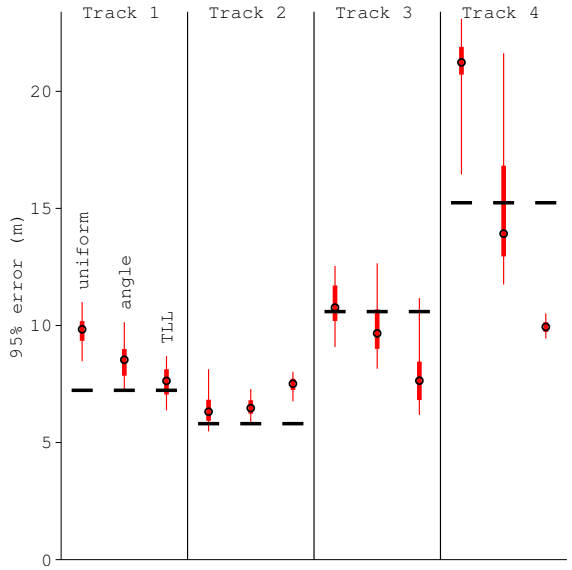
In some applications, it is enough to know the room of the user correctly. Therefore, the box plot in Fig. 3d shows the percentages of how often the position estimates are inside the correct room. The corridor system is counted as one room. According to the figure, the graph-based particle filters perform significantly better than the KF. The difference is clearer in room detection rate than in positioning accuracy. Furthermore, the TLL-based filter has slightly better room detection rates than the comparison methods.



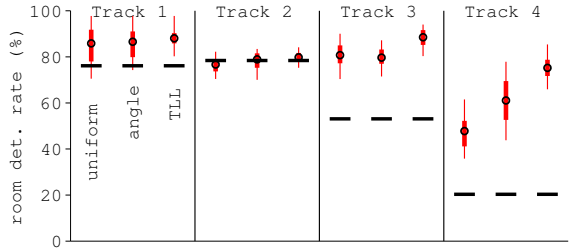
(a) Mean errors



(b) Median errors

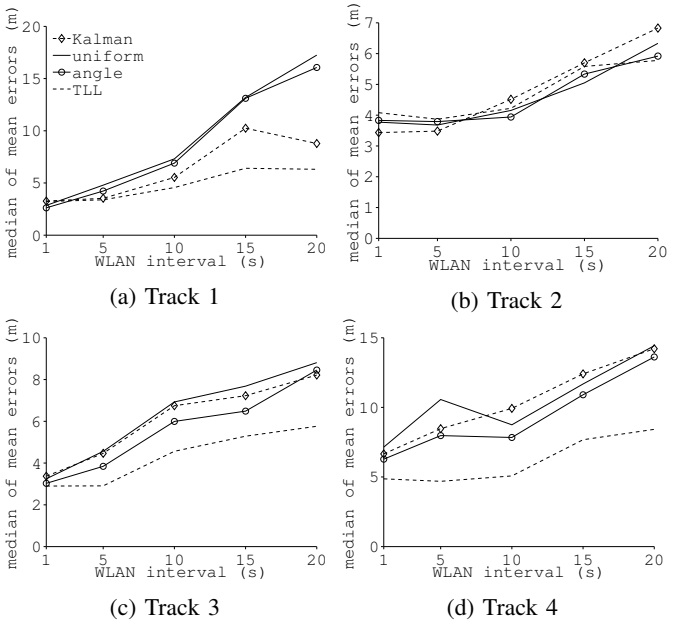


(c) 95 % quantiles of errors



(d) Room detection rates

Figure 3. Error statistics and room detection rates of the four test tracks as a function of the link transition probability rule. The black horizontal line segments indicate the Kalman filter results. The WLAN scan interval is 5 s.



(a) Track 1 (b) Track 2

(c) Track 3 (d) Track 4

Figure 4. Errors as a function of the WLAN scan interval with different link transition probability rules and the Kalman filter

Fig. 4 shows the medians of the mean errors obtained in 100 Monte Carlo replications as a function of the WLAN measurement time interval. An interesting result is that the TLL-based particle filter outperforms the other methods more and more clearly when WLAN interval is increased. The only exception is track 2, where the unpredictable turns to small rooms make the performance deteriorate with large WLAN intervals regardless of the algorithm.

V. Conclusions

A novel statistical training-free motion model for indoor positioning with graph-based indoor map was presented. The proposed model gives more probability to the directions that allow access to larger or more branched areas in the graph. The probability parameters can be updated if training data are available. A particle filter using the proposed model and WLAN-based position measurements was also presented and compared with the conventional methods. Moreover, an optimal method for enforcing the particle cloud based position estimate to be located on a link of the graph was presented.

Compared to the other models, the proposed model is intuitive and is the only one allowing most of the position's probability mass to proceed along a corridor instead of dispersing it to side corridors and rooms. With the other models, the probabilities that continue on the corridor are so low that the particle filters were not able to distribute particles in the true position's area. Still, the method is also able to handle occasional visits in small rooms. With the proposed model, the longer side corridors also get significant weight, as is natural. The proposed model assigns more particles to the graph's most branched areas whose modelling with small numbers of particles is most difficult.

The presented real-data tests showed that the link transition probability rule has significant influence on the filter performance and the particle filter relying on the proposed model

outperforms the comparison methods in office environments. The difference in positioning accuracy is significant especially if most of the test track is in straight or intersecting corridors or if the WLAN measurement time interval is large. Room detection rate, which may be important for the user experience, is also the highest for the proposed method.

In future, methods for ensuring the floor plan graph's uniform level of detail, which is an assumption in the proposed model, will be studied. Moreover, suitability of the graph models to different environments, such as shopping centres, will be tested. Particle smoother using the proposed model will be an interesting topic since the incorporation of future measurements is also based on realistic state modelling. Furthermore, using the graph structure for routing and navigation and incorporating the destination information in the link transition model can be studied. Testing the models with large authentic measurement data sets would be important for making final conclusions of the statistical behaviour of each method and the presented parameter learning algorithm.

Acknowledgment

We thank the HERE team and our colleagues (especially Matti Raitoharju) for their valuable comments.

References

[1] O. Woodman and R. Harle, "Pedestrian localisation for indoor environments," in *10th international conference on Ubiquitous computing (UbiComp'08)*, September 2008, pp. 114–123.

[2] S. Beauregard, Widyawan, and M. Klepal, "Indoor PDR performance enhancement using minimal map information and particle filters," in *2008 IEEE/ION Position, Location and Navigation Symposium (PLANS 2008)*, May 2008, pp. 141–147.

[3] P. Blanchart, L. He, and F. Le Gland, "Information fusion for indoor localization," in *12th International Conference on Information Fusion, 2009. (FUSION'09)*, July 2009, pp. 2083–2090.

[4] H. Leppäkoski, J. Collin, and J. Takala, "Pedestrian navigation based on inertial sensors, indoor map, and WLAN signals," *Journal of Signal Processing Systems*, vol. 71, no. 3, pp. 287–296, June 2013.

[5] H. Nurminen, A. Ristimäki, S. Ali-Löytty, and R. Piché, "Particle filter and smoother for indoor localization," in *2013 International Conference on Indoor Positioning and Indoor Navigation (IPIN2013)*, October 2013, pp. 137–146.

[6] M. I. Khan and J. Syrjärinne, "Investigating effective methods for integration of building's map with low cost inertial sensors and wifi-based positioning," in *2013 International Conference on Indoor Positioning and Indoor Navigation (IPIN2013)*, October 2013, pp. 884–891.

[7] F. Evennou, M. François, and E. Novakov, "Map-aided indoor mobile positioning system using particle filter," in *2005 IEEE Wireless Communications and Networking Conference (WCNC 2005)*, vol. 4, March 2005, pp. 2490–2494.

[8] M. Khider, S. Kaiser, and P. Robertson, "A novel three dimensional movement model for pedestrian navigation," *Journal of Navigation*, vol. 65, pp. 245–264, April 2012.

[9] P. Davidson, J. Collin, and J. Takala, "Application of particle filters to a map-matching algorithm," *Gyroscopy and Navigation*, vol. 2, no. 4, pp. 285–292, October 2011.

[10] L. Lam, S.-W. Lee, and C. Y. Suen, "Thinning methodologies–A comprehensive survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 9, pp. 869–885, September 1992.

[11] L. Liao, D. Fox, J. Hightower, H. Kautz, and D. Schulz, "Voronoi tracking: location estimation using sparse and noisy sensor data," in *2003 IEEE/RSJ International Conference on Intelligent Robots and Systems. (IROS 2003)*, October 2003, pp. 723–728.

[12] J. Yu, W.-S. Ku, M.-T. Sun, and H. Lu, "An RFID and particle filter-based indoor spatial query evaluation system," in *16th International Conference on Extending Database Technology (EDBT'13)*, March 2013, pp. 263–274.

[13] B. Ferris, D. Hähnel, and D. Fox, "Gaussian processes for signal strength-based location estimation," in *2006 Robotics: Sciences and Systems Conference (RSS 2006)*, August 2006.

[14] I. Kim, E. Choi, and H. Oh, "Indoor user tracking with particle filter," in *The Fourth International Conference on Advanced Cognitive Technologies and Applications (COGNITIVE 2012)*, July 2012, pp. 59–62.

[15] P.-Y. Gilliéron, I. Spassov, and B. Merminod, "Indoor navigation enhanced by map-matching," *European Journal of Navigation*, vol. 3, no. 3, 2005.

[16] S. Hilsenbeck, D. Bobkov, G. Schroth, R. Huitl, and E. Steinbach, "Graph-based data fusion of pedometer and WiFi measurements for mobile indoor positioning," in *2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp 2014)*, September 2014, pp. 147–158.

[17] S. S. Skiena, *The Algorithm Design Manual*. New York, NY, USA: Springer-Verlag, 1998.

[18] C. Laoudias, R. Piché, and C. G. Panayiotou, "Device signal strength self-calibration using histograms," in *2012 International Conference on Indoor Positioning and Indoor Navigation (IPIN2012)*, November 2012, pp. 272–279.

[19] H. Nurminen, J. Talvitie, S. Ali-Löytty, P. Müller, E.-S. Lohan, R. Piché, and M. Renfors, "Statistical path loss parameter estimation and positioning using RSS measurements in indoor wireless networks," in *2012 International Conference on Indoor Positioning and Indoor Navigation (IPIN2012)*, November 2012, pp. 461–469.

[20] P. Ivanov, S. Ali-Löytty, and R. Piché, "On consistency of the estimation," in *International Conference on Localization and GNSS (ICL-GNSS)*, June 2014.

[21] B. Ristic, S. Arulampalam, and N. Gordon, *Beyond the Kalman Filter, Particle Filters for Tracking Applications*. Boston, London: Artech House, 2004.

[22] S. Särkkä, *Bayesian Filtering and Smoothing*. Cambridge, UK: Cambridge University Press, 2013.

[23] R. Piché and M. Koivisto, "A method to enforce map constraints in a particle filter's position estimate," in *11th IEEE Workshop on Positioning, Navigation and Communication (WPNC'14)*, March 2014.