A Particle Filter-based Reinforcement Learning Approach for Reliable Wireless Indoor Positioning

José Luis Carrera V.[1], Zhongliang Zhao[1], Torsten Braun[1], Zan Li[2]

University of Bern, Institute of Computer Science, Switzerland[1],
Jilin University, College of Communication Engineering, China[2]
Email: {carrera, zhao, braun}@inf.unibe.ch, {zanli}@jlu.edu.cn

***Abstract*—Positioning is envisioned as an essential enabler of future fifth generation (5G) mobile networks due to the massive number of use cases that would benefit from knowing users' positions. In this work, we propose a particle filter-based reinforcement learning (PFRL) approach for the robust wireless indoor positioning system. Our algorithm integrates information of indoor zone prediction, inertial measurement units, wireless radio-based ranging, and floor plan into an particle filter. The zone prediction method is designed with an ensemble learning algorithm by integrating individual discriminative learning methods and Hidden Markov Models. Further, we integrate the particle filter approach with a reinforcement learning-based resampling method to provide robustness against localization failure problems such as the kidnapping robot problem. The PFRL approach is validated on a two-tier architecture, in which distributed machine learning tasks are hosted at client and edge layer. Experiment results show that our system outperforms traditional terminal-based approaches in both stability and accuracy.**

Keywords: Indoor Positioning, Particle Filter, Reinforcement Learning, Internet of Things, Ensemble Learning Methods, Hidden Markov Model, Kidnapping Robot Problem.

I. Introduction

5G networks are expected to provide wide bandwidth, which can be used to enable highly accurate positioning. In contrast to existing radio networks, where positioning has been provided only as an add-on feature, positioning will play a key role in future 5G systems. The location-awareness will enable not only a vast amount of location-based services and applications such as intelligent transportation systems (ITS) and autonomous vehicles, but also support valuable location-aware communication enhancements such as proactive radio resource management, proactive handover optimization, etc. Thus, reliable localization methods become the underlying requirement to enable location aware-services. Many wireless indoor positioning approaches have been proposed by exploring radio signals such as fingerprinting, radio-based ranging, etc. Radio-based indoor positioning has the intrinsic problem of signal unreliability due to the multi-path propagation and non-line-of-sight (NLOS) signal conditions in indoor environments. One solution is to apply data-driven methods such as machine learning algorithms on received signal data, where a training dataset including measurements with known locations is used to train a model that estimates the location of a data sample. Wi-Fi received signal strength indicator (RSSI) is a common metric in both range-based and fingerprinting-based approaches. The earth magnetic field, which presents distortions over space due to the presence of ferromagnetic materials, is also used to improve localization accuracy [1].

Corresponding author: Zhongliang Zhao (zhao@inf.unibe.ch).

Range-based approaches convert the received radio signals into range values, which indicate the distance between the target mobile device and radio transceiver. After ranging, multilateration methods can be adopted to derive the absolute position of the target. However, ranging accuracy is detrimentally affected by multi-path effects particularly in Non-Line of Sight (NLOS) conditions. To deal with the negative influence of multi-path effects, NLOS conditions need to be identified, then some methods can be adopted to mitigate its negative effects. Fingerprinting-based systems usually consist of an off-line phase (training phase) and on-line phase (localization phase). The off-line phase is aimed to build the fingerprint database. The fingerprint database is built by collecting several types of radio signals in the area of interest (i.e., target indoor environments). The on-line phase is aimed to perform the localization process, in which a new fingerprint measurement at a random location is compared with the fingerprint database. In the on-line phase, any single learning algorithm can be used. However, ensemble learning methods usually allow better predictive performance compared to single models [23]. Fingerprinting-based system builds the classification model based on fingerprint data collected previously. Therefore, fingerprinting-based systems can be called discriminative learning methods.

In addition to wireless radio signal instability, there are two other common problems in wireless indoor positioning: the global localization problem and the kidnapped-robot problem. The global localization problem happens when the localization system initializes, where the initial position of the target is unknown. The kidnapped-robot problem occurs when a well-located target in operation moves to some arbitrary locations, while the target itself is not aware of this. Therefore, the kidnapped-robot problem normally happens during the regular system operations. Most of the state-of-the-art indoor positioning approaches cannot guarantee avoidance of failure[37]. Therefore, the ability to prevent the system from catastrophic localization failures is essential for truly autonomous and reliable localization systems.

Reinforcement learning is a type of machine learning technique that enables an agent to learn in an interactive environment by trial and error using feedback from its own actions and experiences [36]. The key idea of reinforcement learning is to determine a mapping between actions and states to maximize a numerical reward signal. The learner agent interacts with the environment and receives rewards or penalties for the performed actions. Therefore, reinforcement learning uses rewards and punishments as signals for positive and negative behaviors such that the agent behavior can be optimized continuously.

In this work, we present a particle filter-based reinforcement learning (PFRL) approach for the reliable wireless indoor positioning. The PFRL system includes a particle filter component for accurate indoor positioning and a reinforcement learning-based resampling method to guarantee system robustness against localization failures. Thus, PFRL achieves high localization accuracy and high reliability against localization failures. The particle filter method fuses indoor zone prediction, range radio information, inertial measurement units (IMUs), and floor plan information. The zone prediction method is designed with an ensemble learning algorithm by combining different individual predictors in a Hidden Markov Model (HMM). The zone prediction method provides zone-level localization, which supports to choose the proper ranging models that are specific for each zone. If the mobile client (i.e., object to be located) and a ranging Anchor Node (AN) are at the same zone, the system adopts LOS ranging models with respect to this AN. Furthermore, ranges to ANs located at different zones than the mobile client are calculated by NLOS ranging models. Further, we propose an efficient reinforcement learning-based resampling method to assure the placement of samples (i.e., particles) over areas where the desired distribution is large (i.e., areas with high probability of containing the ground truth position). This scheme reduces convergence time and provides autonomy and robustness to the system. Figure 1 shows the architecture of our proposed system, which is a two-tier architecture supporting distributed machine learning operations. The main contributions of this work are as follows.

- We design a particle filter-based reinforcement learning (PFRL) algorithm for reliable wireless indoor positioning. The particle filter fuses the predicted zone, radio-based ranges, IMUs, and coarse-grained floor plan information to achieve accurate and stable real-time indoor tracking performance. In the particle filter, we provide an reinforcement learning-based resampling method to guarantee system robustness against localization failures.
- We propose a distributed machine learning-based network architecture for indoor positioning, where lightweight ML algorithms (indoor zone prediction) are running on the mobile devices with limit resources and heavy ML calculations (proposed PFRL algorithm) are offloaded to nearby edge servers to support complex and heavy calculations.
- We perform a set of experiments in complex office and classroom-like environments along five different moving paths to validate the accuracy, reliability, and scalability of our system. We compare PFRL with client-based system where all computations are running on end devices. With an average localization error of 0.97 meters and failure recovery time latency of 1.5 seconds, our proposed localization method overcomes traditional solutions in terms of reliability, stability, and accuracy. We also discuss the experimental evidence of the fast convergence of our PFRL approach compared to standard PF-based approaches.

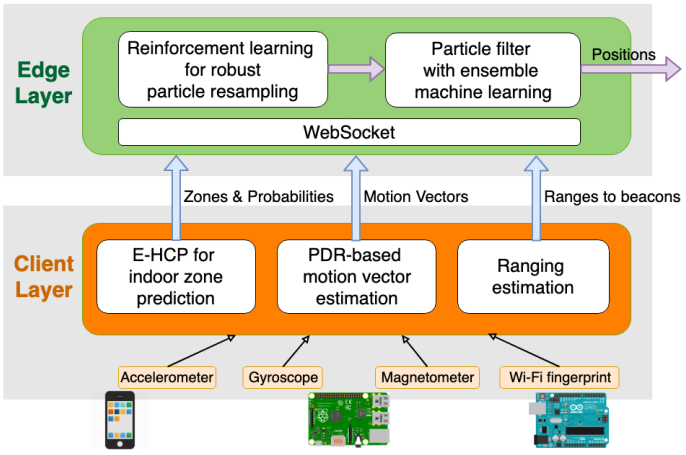


Figure 1: Distributed Machine Learning System Architecture for Reliable Wireless Indoor Positioning.

The rest of this work is organized as follows. Section II presents related work. Section III describes the localization system. Implementation details are presented in Section IV. Section V describes the localization performance evaluation of our approach. Section VI concludes the paper.

II. Related Work

The current development of embedded inertial measurements units (IMUs) in off-the-shelf mobile devices (e.g., smartphones), has increased research interest in Pedestrian Dead Reckoning (PDR) systems. IMUs can be exploited to provide pedestrian movement detection such as step recognition, heading direction, stride length estimation [32]. Therefore, by using IMUs, PDR systems can infer the current location based on the previously defined location. For instance in [16], the stride length is calculated based on accelerometer readings, whereas the heading direction is computed from gyroscope readings. PDR methods measure position changes rather than absolute positions. Thus, PDR methods are prone to accumulate sensor errors over time, which means PDR-based system must apply additional information fusion to revise the localization error periodically.

In indoor environments, radio signals are usually exploited to provide positioning services. In [25] for instance, the authors use radio propagation time information, whereas in [25] Wi-Fi RSSI is used. Radio signal-based indoor localization methods are classified as range-free and range-based methods. Range is the propagation distance from the target device to

Anchor Nodes (AN). The method to compute the propagation distances is called ranging. Ranging is the initial stage in range-based localization methods. After ranging, several localization techniques can be used to estimate the absolute position of the targets, such as multilateration [24]. However, range-based methods are not accurate in indoor environments because of the presence of obstacles (e.g., furniture) and room partitions [13]. Therefore, range-free methods are usually used as an alternative to range-based methods. Fingerprinting [2] is a widely used range-free localization method. It is because of its robustness to multi-path propagation. However, to build up a radio map is very time consuming. Typically, building a radio map database for fingerprint requires much more effort than range-based methods for localization. In order to improve the accuracy of fingerprint-based localization, authors of [14] proposed to fuse step counter measurement with location estimation to reduce the calibration efforts. In [15], authors proposed a graph-based, low-complexity sensor fusion approach for ubiquitous pedestrian indoor positioning using mobile devices. However, the system is not robust again positioning failures.

Hidden Markov Models (HMM) can also be used for indoor positioning. In [39] authors used radio propagation models and HMM to reduce the efforts during the calibration process. The probable positions are inferred by using discrete probability distributions. Afterwards, the position is computed from the set of most probable estimated positions. In [27], the authors propose to fuse wireless signal measurements with IMU readings. Then, the position is determined by computing based on the most probable wireless signals measurement and the pedestrian motion pattern at that position. Although authors claim high accuracy, the transition probability definition method remains unclear. Additionally, the applicability of the approach relies on the fidelity of the PDR method.

Reinforcement learning (RL) has achieved great success recently in different application domains. In [33], authors utilized a deep reinforcement learning algorithm to learn action policies for managing an optimal dose of medicines like heparin for individuals. They used a sample data set of dosage trials and their outcomes from a bunch of electronic medical records. In [44], a convolutional neural network (CNN) model was integrated with a reinforcement learning module to indicate which part of an image should be searched to detect a car from the image. Resource management is another application that can benefit from using RL. In [28], authors applied RL to solve the problem of job scheduling with multiple resource demands. Recently, researchers in computer networks also started to investigate how RL can help them to improve mobile and wireless network performance. In [43], authors made a comprehensive survey of the crossovers between DL and wireless networking.

Particle filters are normally known as Sequential Monte Carlo methods [8], which were originally designed to solve statistic problems. Particle filters are able to approximate any probability density function, which can be regarded as a sequential analogue of Markov chain Monte Carlo (MCMC) methods. Although particle filter is a statistic approach, it has been successfully applied in many applications, such as Monte Carlo localization of mobile robots [12], simultaneous localization and mapping (SLAM) [30], etc. However, the benefits of exploring particle filters into the reinforcement learning domain seems to be missing so far. To the best of our knowledge, this work is the first attempt to apply particle filter in reinforcement learning to achieve accurate and reliable wireless indoor positioning.

Despite the localization approaches presented in our previous work [3], [4], [5], [6] achieve high accuracy, the localization and tracking algorithms of these approaches have some drawbacks. Thus, in this work, we focus on improving the algorithm design to overcome these drawbacks. Moreover, the described localization algorithms are completely different in each work. In the following, we present the main differences and improvements of this work compared to our previous work. In [4] we propose an ensemble learning method to localize the target in a zone level accuracy (e.g., room level), whereas in our current work we provide continuous real-time tracking in a sub-zone level accuracy (e.g., sub-room level). Thus, we design and implement a particle filter-based approach to fuse zone detection likelihoods, IMUs, ranging and floor plan information. In [3] we propose an enhanced particle filter with double resampling method to provide indoor tracking. Although the localization approach achieves high accuracy, the localization method relies only on the Wi-Fi ranging method. This could lead to some localization failures such as kidnapping robot problem. In our current work, the localization algorithms do not only rely on Wi-Fi ranging methods but also on machine learning techniques and information about transitions between rooms. Hence, we have modified the particle filter design to provide accurate and more stable indoor tracking performance. In [5], we present a tracking particle filter-based method able to recover the system from localization failures. However, the particle filter and failure recovery methods rely only on Wi-Fi ranging techniques. In [6], we discuss the performance of single well-known machine learning algorithms along Wi-Fi ranging techniques to provide static indoor localization. In this work we incorporate a novel ensemble learning method along zone transition information, IMUs, Wi-Fi ranging and floor plan information to provide continuous tracking. Moreover, in this work we present a novel reinforcement learning-based method to guarantee system reliability against localization failures while keep high tracking accuracy.

III. System Overview

This section presents the design details of the proposed particle filter-based reinforcement learning approach for wireless indoor positioning. Figure 1 summarizes the architecture of our proposed approach, which includes two layers: a Client layer and an Edge layer. The Client layer includes mobile clients to be located, such as smartphones, Raspberry Pi devices, or Arduino devices, etc. The Edge layer includes edge servers, which are responsible for hosting complex positioning

algorithms. Due to limited amount of resources available, mobile clients host components that are able to process low computation overheads, which include: a HMM-based ensemble predictor for indoor zone prediction, an enhanced ranging model, a PDR-based move detection method and a floor plan component that defines a discrete system state, the map likelihood (i.e., allowed areas to move), and the transition model (i.e., physical distribution of zones). Edge servers host the proposed PFRL algorithm, which considers the outputs of mobile clients as inputs to estimate the real-time positions of mobile devices. PFRL includes two parts: a particle filter-based ensemble predictor to provide high positioning accuracy and a reinforcement learning approach, which builds on top of the particle filter to guarantee system robustness against failures. Details of each component and the interconnections between client and edge layers are described in the next subsections.

A. Mobile Client - Ensemble HMM-Conditional Performance Learning (E-HCP) for Indoor Zone Prediction

The key idea of E-HCP is to combine conceptually different individual machine learning models in an HMM. Thus, we combine several individual machine learning algorithms to improve prediction performance compared to individual models. It is worth to notice that the E-HCP method can be applied in any prediction problem that involves HMM. We define a zone as any subarea inside the area of interest. Therefore, hereafter, we refer to a room as a zone. Considering the concept of Markov localization [11], the E-HCP method can be described by estimating the state of the system with controllable state transitions. Therefore, for the localization problem, we define zones as states of E-HCP. Therefore, the E-HCP method is specified by the following components:

- A set of states $Z = \{z_1, ..., z_n\}$, where n is the number of zones, z_l is the identifier value of the zone l. Therefore, the hidden state z_l at time t can be represented by the discrete random variable $z_{l_t} \in Z$.
- A transition probability matrix T,

$$T = \begin{pmatrix} t_{1,1} & t_{1,2} & ... & t_{1,n} \\ t_{2,1} & t_{2,2} & ... & t_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ t_{n,1} & t_{n,2} & ... & t_{n,n} \end{pmatrix},$$

  where $t_{k,l} \in T$ is the likelihood of moving from zone z_k to zone z_l. Thus, T is a square matrix of order n.
- A set of observations C,

$$C = \{(c_1, ..., c_m)_1, ...(c_1, ..., c_m)_{n^m}\}, \tag{1}$$

  where $c_k \in Z$ is the prediction outcome of the k-th constituent individual machine learning algorithm. Thus, C is a set of ${}^nP_m$ permutations with repetition allowed, where n is the number of zones and m is the number of individual machine learning algorithms that constitute the E-HCP method. We defined $(c_1, ..., c_m)_l \in C$ as q_l. The observation q_l at time t can be represented by the random variable $q_{l_t} \in C$.
- A matrix B of observation probabilities. B is named the emission probability matrix.

$$B = \begin{pmatrix} b_{1,1} & b_{1,2} & ... & b_{1,n^m} \\ b_{2,1} & b_{2,2} & ... & b_{2,n^m} \\ \vdots & \vdots & \ddots & \vdots \\ b_{n,1} & b_{n,2} & ... & b_{n,n^m} \end{pmatrix},$$

  where $b_{k,l}$ is the likelihood of observing $q_l \in C$ at zone z_k.
- An initial probability distribution over zones $\pi = \pi_1, ..., \pi_n$, where π_i is the probability of being located in zone i.

In any model with hidden variables such as E-HCP, a decoding task is to determine the sequences of variables that is the underlying source of a observation sequence. The decoding task can be described as the process to solve the following equation:

$$p(z_{l_t} \mid q_{l_t}) = \frac{p(q_{l_t} \mid z_{l_t}) \cdot p(z_{l_t})}{p(q_{l_t})} \tag{2}$$

where $P(z_{l_t} \mid q_{l_t})$ is the probability being located at zone z_{l_t} given the observation q_{l_t} at time t. Therefore, considering a sequence of observations $q_{0_0}, ..., q_{l_t}$, and an HMM model $\lambda = \{\pi, T, B\}$, Equation 2 can be computed by applying the Viterbi algorithm [10].

In the following two subsections we explain how to determine the emission probability matrix B and the transition probability matrix T.

1) Emission Probabilities: The emission probability is the likelihood of observing $q_l \in C$ at zone $z_k \in Z$. Therefore, the probability $b_{k,l}$ (entry of matrix B) of having a particular set of observations q_l at zone z_k can be defined as:

$$b_{k,l} = p(q_l \mid z_k), \forall q_l \in C \wedge z_k \in Z, \tag{3}$$

$p(q_l \mid z_k)$ can be computed assuming conditional independence among the prediction outcomes $c_i \in q_l$ given z_k. Our assumption is that the probability of obtaining the outcome c_i becomes independent if the value of z_k is known. Moreover, the individual constituent learning algorithms of E-HCP are independent and conceptually different of each other. Therefore, it is reasonable to assume that their outcomes are conditionally independent given z_k. Thus, $b_{k,l}$ can be written as follows:

$$b_{k,l} = \prod_{i=1}^{m} p(c_l \mid z_k)_i, \tag{4}$$

where $P(c_l \mid z_k)_i$ is the probability of predicting c_l at zone z_k by the i-th constituent individual learning algorithm of E-HCP. Therefore, $P(c_l \mid z_k)_i$ represents the prediction performance of the i-th individual machine learning algorithm given the knowledge of the ground-truth class label (i.e., zone). $P(c_l \mid z_k)_i$ can be obtained from the confusion matrix of the i-th individual machine learning algorithm part of the E-HCP method.

2) Transition Probabilities: Connection among zones in the coarse-grained floor plan defines the transition probabilities. Thus, the transition probability is the likelihood of moving from one zone to another. Therefore, the transition probability matrix can be expressed as follows:

$$t_{k,l} = p(z_l \mid z_k), \tag{5}$$

where $t_{k,l} \in T$ represents the transition probability between zone z_k to zone z_l. T is a square matrix and $\sum_{l=1}^{n} t_{k,l} = 1$.

B. Mobile Client - PDR-based Motion Vector Estimation

PDR methods estimate the displacement of the mobile client device by detecting changes in a previously estimated position. PDR methods must be adjusted accordingly to consider different movement characteristics of the mobile client such as velocity, acceleration, etc. In this work, the PDR methods estimate pedestrian's displacement. It is estimated by using three device embedded sensors: the accelerometer, the geomagnetic field sensor, and the gyroscope. At time t, the displacement of the pedestrian is defined by the motion vector $M_t = [\ell_t, \theta_t]$, where θ_t is the heading orientation, and ℓ_t is the displacement length. Thus, M_t is passed to the Particle Filter component at instant t when a displacement (e.g., step) of the pedestrian is detected. Additional details about the PDR method can be found in our previous work [3].

C. Mobile Client - Ranging Estimation Process

Ranges can be derived by using signal parameters such as RSSI. In theory, RSSI monotonically decreases with increasing propagation distance [24]. However, in complex indoor environments, WiFi signals suffer from random variations. To reduce ranging errors introduced by NLOS and multi-path propagation, we propose a propagation model by combining Log Distance Path Loss (LDPL) [35] and a Nonlinear Regression Model (NLR) [24] [6]. Our propagation model can be written as follows:

$$r = \begin{cases} 10^{(\frac{Pw(r_0) - Pw(r)}{10 \cdot \gamma})} & \textit{if Tx and Rx are at the same zone} \\ \alpha \cdot e^{(\beta \cdot Pw(r))} & \textit{if Tx and Rx are at different zones} \end{cases} \tag{6}$$

where *Tx* and *Rx* are the transmitter and receiver devices respectively. Variable $P_w(r_0)$ refers to the power loss in a free space, $P_w(r)$ is the received signal power in a propagation distance r. Variable γ is the path loss efficient [35], and both α and β are environmental variables [24].

D. Edge Server - Particle Filter with Data Fusion and Machine Learning Methods

Indoor positioning can be assumed as a filtering problem, in which the system state (e.g., target position) can be inferred from a set of noisy environmental observations. Thus, particle filtering is able to solve estimation problems recursively as observations become available. The objective is to determine the posterior distributions of the system's states given some noisy observations. The posterior probability is expressed as a set of weighted samples (also called particles). Thus, the posterior probability distribution is computed based on some observation O_t at time t [3]. A time t, the particle system state vector X_t can be written as:

Data: M_t, O_t
Result: Mobile client position
Calculate the initial zone probability distribution: $p(z_{l_0} \mid q_{l_0}) =$ *E-HCP()*;
Distribute particles based on $p(z_{l_0} \mid q_{l_0})$;
Initialize particle's weights: $W_0^i = 1/N_p$, $i = 1, 2, ..., N_p$;
while *Localizing* **do**
  Update the particles: $X_t^i = G \cdot X_{t-1}^i + \eta$;
  Calculate the ranging likelihood: $P(\hat{d}_{j,t} \mid X_t^i) = \frac{1}{\sigma_j \sqrt{2\pi}} \exp^{-\frac{[\hat{d}_{j,t} - \sqrt{(x^i - x_j)^2 + (y^i - y_j)^2}]^2}{2\sigma_j^2}}$;
  Calculate the zone probability distribution: $p(z_{l_t} \mid q_{l_t}) =$ *E-HCP()*;
  Calculate the zone likelihood: $P(z_{l_t} \mid X_t^i) = \frac{P(X_t^i \mid \hat{z}_{l_t}) \cdot P(\hat{z}_{l_t})}{P(X_t^i)}$;
  Compute unnormalized weights: $\hat{w}_t^i = P(X_t^i \mid \hat{z}_{l_t}) \cdot \prod_{j=1}^{M} P(\hat{d}_{j,t} \mid X_t^i)$;
  Normalize weights: $w_t^i = \frac{\hat{w}_t^i}{\sum_{n=1}^{N} \hat{w}_n^i}$;
  Resample the particles;
  Calculate the degree of depletion;
  Run RL method for robust tracking resampling;
  Compute the estimated position:$X_t = \sum_{i=1}^{N} w_t^i \cdot x_t^i$;
end

$$X_t = [x_t, y_t, z_{l_t}, \ell_t, \theta_t], \tag{7}$$

where (x_t, y_t) defines the 2-dimensional position of the target object, $z_{l_t} \in Z$ is the zone where the target is located.

At time t, the set of particles can be expressed as:

$$P_t = [X_t^i, W_t^i], i = 1, ..., N_p, \tag{8}$$

where N_p is the number of particles, X_t^i is the state vector, and W_t^i is the associated weight of the *i*-th particle at time *t*. The posterior probability given a sequence of observations $p(X_t \mid O_{1:t})$ can be defined as:

$$p(X_t \mid O_{1:t}) \approx \sum_{i=1}^{N_p} w_t^i \delta(X_t - X_t^i), \tag{9}$$

where X_t^i is the i-th particle, and w_t^i is the associated weight at time t. The associated weights w_t^i can be computed as follows:

$$w_t^i \propto w_{t-1}^i * p(O_t \mid X_t^i), \tag{10}$$

where $p(O_t \mid X_t^i)$ is the *likelihood* function calculated from the observation vector O_t at time t. The RSSI in indoor environments tends to be unstable due to multi-path effects. To alleviate these problems, we propose to fuse multiple information in an enhanced particle filter such as wireless radio-based ranging, inertial measurement units, floor plan

information, and indoor zone prediction results. Algorithm 15 indicates the procedures of the proposed particle filter.

1) Prediction Model: Since each location belongs to a zone, the current zone z_{l_t} depends on the current Cartesian coordinates (x_t, y_t). Coordinates x_t and y_t are elements of the system state vector X_t. Therefore, it is possible to define a function $getZone(X_t)$ to derive the current zone z_{l_t} from the system state vector X_t. Thus, z_{l_t} can be obtained as follows:

$$z_{l_t} = getZone(X_t) \tag{11}$$

Therefore, the particle filter prediction function can be written as:

$$X_t = G \cdot X_{t-1} + \eta, \tag{12}$$

where

$$G = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}, \eta = \begin{pmatrix} \ell_t \cdot cos(\theta_t) \\ \ell_t \cdot sin(\theta_t) \\ z_{l_t} \\ \ell_t \\ \theta_t \end{pmatrix}$$

As reported in previous studies [20], [22] , heading orientation θ_t and stride length ℓ_t are interfered by zero-mean Gaussian random noises $\varepsilon^{'}$ and $\varepsilon^{''}$, respectively. Variables θ_t and ℓ_t are estimated by PDR methods. The variable z_{l_t} identifies the zone where the particle is located at time t.

State vector X_t^i of each particle is updated from the particles at the previous time interval X_{t-1}^i based on Equation (12). Thus, the new set of particles P_t is generated from P_{t-1}. Particles are not allowed to move through restricted areas, (e.g., movement through walls is not allowed).

2) Observation Model for Data Fusion: Particles are propagated based on Equation (12). Afterwards, the associated weight w_t^i of the propagated particles must be calculated. At time t, the associated weight $p(O_t \mid X_t^i)$ is calculated based on the likelihood of the observations conditioned on current particle state X_t^i. The observation vector O_t contains the ranging information to different ANs and the predicted zone information. Thus, at time t, the observation vector can be expressed as $O_t = [d_t, q_{l_t}]$, where d_t contains ranges to different ANs and $q_{l_t} \in C$ contains the predicted zone information. Ranges are computed by the Ranging Estimation process presented in section III-C. The zone prediction information is provided by the E-HCP method presented in section III-A.

Since the ranging method (i.e., the method to estimate ranges) and the E-HCP method for zone prediction are completely different, we can assume that $p(q_{l_t} \mid X_t^i)$ and $p(d_t \mid X_t^i)$ are independent of each other. Therefore, the probability $p(O_t \mid X_t^i)$ can be expressed as follows:

$$p(O_t \mid X_t^i) = p(d_t \mid X_t^i) \cdot p(q_{l_t} \mid X_t^i) \tag{13}$$

We refer to $p(d_t \mid X_t^i)$ as the ranging likelihood, and $p(q_{l_t} \mid X_t^i)$ as the zone likelihood. Therefore, the associated weight $w_t^i = p(O_t \mid X_t^i)$ of each particle is given by the ranging and zone prediction information. The particles at the absolute position (x_t, y_t) with low probability to observe d_t^j in their position will be assigned a small ranging likelihood. Particles positioned in zones with low probability of observing q_{l_t} will be assigned small zone likelihood values.

3) Ranging likelihood: The ranging likelihood $p(d_t \mid X_t^i)$ is given by the ranging information. Since ANs operate independently, it is possible to assume that the ranges to different ANs are conditionally independent of each other. Thus, the ranging likelihood can be written as follows:

$$p(d_t \mid X_t^i) = \prod_{j=1}^{N_a} p(\hat{d}_{j,t} \mid X_t^i), \tag{14}$$

where N_a is the number of ANs, $\hat{d}_{j,t}$ is the estimated distance to the j-th AN at time t. Hereafter, $p(\hat{d}_{j,t} \mid X_t^i)$ will be referred as the individual ranging likelihood. Ranging errors can be modeled as Gaussian distributed values [41], [24]. Thus, the individual ranging likelihood can be further written as:

$$p(\hat{d}_{j,t} \mid X_t^i) = \frac{1}{\sigma_j \sqrt{2\pi}} \exp^{-\frac{[\hat{d}_{j,t} - \sqrt{(x^i - x_j)^2 + (y^i - y_j)^2}]^2}{2\sigma_j^2}}, \tag{15}$$

where (x_j, y_j) are the coordinates of the j-th ranging AN.

4) Zone Likelihood: Zone likelihood refers to the zone prediction information. Therefore, zone likelihood is the probability of observing q_{l_t} in the current particle state X_t^i. The value of the variable q_{l_t} is computed by the E-HCP method. Thus, $p(q_{l_t} \mid X_t^i)$ can be written as:

$$p(q_{l_t} \mid X_t^i) = \frac{p(X_t^i \mid q_{l_t}) \cdot p(q_{l_t})}{p(X_t^i)}, \tag{16}$$

where q_{l_t} is the zone related set of observations at time t. Since the $p(X_t^i)$ and $p(q_{l_t})$ are constant, $p(q_{l_t} \mid X_t^i)$ depends only on $p(X_t^i \mid q_{l_t})$. Therefore, $p(q_{l_t} \mid X_t^i) \propto p(X_t^i \mid q_{l_t})$. $p(X_t^i \mid q_{l_t})$ can be written as follows:

$$p(X_t^i \mid q_{l_t}) = \frac{p(q_{l_t} \mid X_t^i) \cdot p(X_t^i)}{p(q_{l_t})} \tag{17}$$

From function $getZone$ defined in Equation 11, $z_{l_t}^i$ can be obtained from X_t^i. Therefore, considering the respective zone $z_{l_t}^i$ of X_t^i, Equation 17 can be written as follows:

$$p(z_{l_t}^i \mid q_{l_t}) = \frac{p(q_{l_t} \mid z_{l_t}^i) \cdot p(z_{l_t}^i)}{p(q_{l_t})} \tag{18}$$

Since C is the set of observations related to the zone prediction (see Equation 1) and q_{l_t} is a vector that contains zone related observations, we can consider q_{l_t} as an element of C ($q_{l_t} \in C$). Thus, $p(z_{l_t}^i \mid q_{l_t})$ can be solved by the E-HCP method described in section III-A.

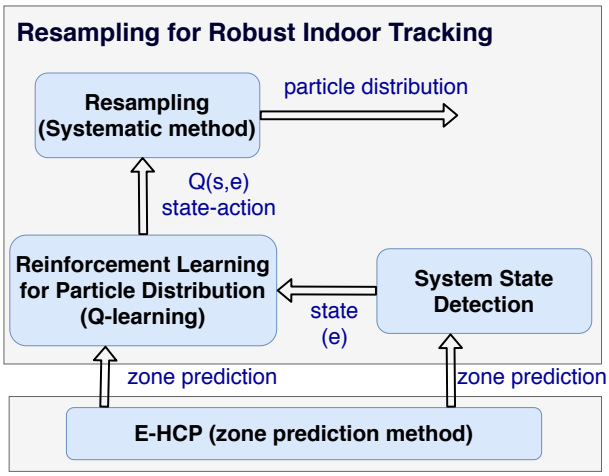


Figure 2: Reinforcement Learning Method for Robust Indoor Tracking Resampling.

E. Edge Server - Reinforcement Learning for Robust Indoor Tracking Resampling

In particle filter localization approaches, the localization process performs poorly if the proposal particle distribution (i.e., distribution used to generated samples) places too few samples in areas where the desired posterior distribution is large. Such behaviour leads to increase convergence time of the particle filter. Moreover, an unsuitable proposal distribution could trigger localization failures such as the kidnapping robot problem. To mitigate these problems, we propose to use adaptive proposal distributions in addition to the resampling method. The proposal particle distributions are built based on a reinforcement learning method which relies on the E-HCP outcomes and the current state of the system. Figure 2 depicts the architecture of the proposed reinforcement learning method for robust indoor tracking. The proposal particle distribution assures the placement of samples over the areas where the desired distribution is large. The tracking algorithm learns by itself which proposal distribution suits better at each state of the system. This scheme reduces convergence time and provides autonomy and robustness to the system.

1) Resampling Method: Resampling is a fundamental process for particle filters. Without resampling, particle filters will produce a degenerate set of propagated particles (i.e., most of the particles with negligible weight). The resampling process modifies the weighted approximate density p to an unweighted density $\hat{p}$ by eliminating particles with low importance weights (i.e., small associated weight) by multiplying particles having high importance weights (i.e., high associated weight). The new density $\hat{p}$ is called the proposal particle distribution. Therefore, $p(X_t \mid q_{1:t}) = \sum_{i=0}^{N_s} w_t^i \delta(X_t - X_t^i)$ is replaced by $p(\hat{X}_t \mid q_{1:t}) = \sum_{i=0}^{N_s} \frac{n_i}{N_s} \delta(\hat{X}_t - \hat{X}_t^i)$, where n_i is the number of copies of particle X_t^i in the new set of particles $\hat{P}_t$. There are many methods to generate $\hat{P}_t$ [7]. We perform the resampling process by using the systematic method. The systematic resampling method aims to prevent the degeneracy of the propagated particles by modifying the set P_t to $\hat{P}_t$.

Figure 3: PFRL transition model learner agent.

Particles from P_t with higher weights are more likely to be included in the new set of particles $\hat{P}_t$. Thus, in the next iteration, more particles will be propagated in zones with large probability masses [21]. Before resampling, the weights W_t^k are normalized, i.e., $\sum_{k=1}^{N_p} W_t^k = 1$. Then, a set of N_p numbers u_t^n is generated from an uniform distribution. This set of numbers is used to select N_p particles from P_t. Thus, the particle x_t^n is selected in the n-th iteration if the following condition is satisfied:

$$S_t^{m-1} < u_t^n \leq S_t^m, m = 1, ..., N_p, \tag{19}$$

where

$$S_t^m = \sum_{k=1}^{m} W_t^k, \tag{20}$$

The interval $(0, 1]$ is divided into N_p disjoint sub-intervals $(0, 1/N_p] \cup ... \cup (1 - 1/N_p, 1]$. Then, u_t^1 is generated as a random number from the uniform distribution on $(0, 1/N_p]$. The remaining u_t^n numbers are obtained from u_t^1 as follows:

$$\begin{aligned} u_t^1 &\sim U(0, 1/N_p], \\ u_t^n &= u_t^1 + \frac{n-1}{N_p}, \quad n = 2, 3, ..., N_p, \end{aligned} \tag{21}$$

After generating the set of u_t^n numbers, the new set of particles $\hat{P}_t$ is generated by selecting N_p particles from P_t based on the condition presented in Equation 19. Although the systematic resampling method can achieve high performance, this method alone does not guarantee the system to avoid and recover from localization failures.

2) Reinforcement Learning Method for Particle Distribution: Considering a reinforcement learning context, our tracking algorithm is modeled as the learner agent (LA). The LA provides localization and at the same time learns the optimal behaviour to prevent and recover the system from localization failures. The Q-learning [42] approach is adopted as a reinforcement learning method. Q-learning provides agents with the ability to learn how to proceed optimally by experiencing the consequences of actions [42]. In Q-learning, the LA evaluates the consequences of an action at a particular state. This evaluation is performed in terms of an immediate penalty or reward. Thus, by trying all actions in all states repeatedly, LA learns the optimal behaviour at each state. Figure 3 shows the states and transition model of our proposed reinforcement learning model. Since the purpose of the LA is to provide autonomy and robustness to the system, we defined three states: Starting state, which is achieved when the system is started, Localizing state, which is achieved when the system is providing localization service, and Failing state, which is achieved when some localization failure is detected. Elements of the set of actions are as follows.

- Action e_1 defines a uniform particle distribution across the whole target area. This action can be performed in the Starting and Failing states.
- Action e_2 defines a particle distribution across the predicted zone. Zone information is computed by the E-HCP method. This action can be performed in the Starting and Failing states.
- Action e_3 defines a particle distribution based on the predicted zone probability distribution. Zone probability distribution is computed by the E-HCP method. This action can be performed in the Starting and Failing states.
- Action e_4 defines a particle distribution of the $g\%$ worst evaluated particles (i.e., particles with the lowest weight) across the predicted zone. This action can be performed in the Localizing and Failing states.
- Action e_5 defines a particle distribution of the $g\%$ worst evaluated particles based on the predicted zone probability distribution. This action can be performed in the Localizing and Failing states.

The Q-learning algorithm performs the learning process based on the Bellman equation as follows:

$$Q(s,e) \leftarrow (1-\alpha)\cdot Q(s,e) + \alpha \cdot [R(s,e) + \gamma \cdot \max(Q(s',e'))], \tag{22}$$

where $Q(s,e)$ determines the quality of a state-action combination. Thus, when action e is performed in state s, $Q(s,e)$ is updated based on Equation 22. The learning rate is defined by α, which determines how valuable recent information is for the learning process. Thus, if $\alpha = 0$, the LA exploits only previous learned knowledge, while $\alpha = 1$ makes the LA to consider only the most recent information. The parameter γ defines the discount rate, which determines what percentage of a future reward must be considered in the training process. The variable $\max(Q(s',e'))$ is the maximum achievable $Q(s',e')$ value, which is possible to obtain in the next state s' by performing e'. The function $R(s,e)$ computes the reward of performing action e at state s. Further details about the Q-learning algorithm can be found in [42], [29].

We define $R(s,e)$ as a function to compute the degree of depletion in the particle filter method. The degree of depletion describes the rate of particles having a negligible weight. Some particles can be located away from the ground truth location. Therefore, these particles are evaluated with nearly negligible weights. The density of particles should be high in high-probability zones, and low in low-probability zones. The effective number of samples (N_{eff}) is an indicator of the degree of depletion [26]. Therefore, N_{eff} measures how efficiently the particle distribution is representing the ground truth location. Since the degree of depletion indicates the quality of particle distributions, and the effective number of samples N_{eff} is an indicator of the degree of depletion, we define $R(s,e) = N_{\text{eff}}$. Thus, the value of N_{eff} for N_p number of particles can be calculated as follows:

$$R(s,e) = N_{\text{eff}} = \frac{1}{\sum_{i=1}^{N_p}(w^i)^2} \tag{23}$$

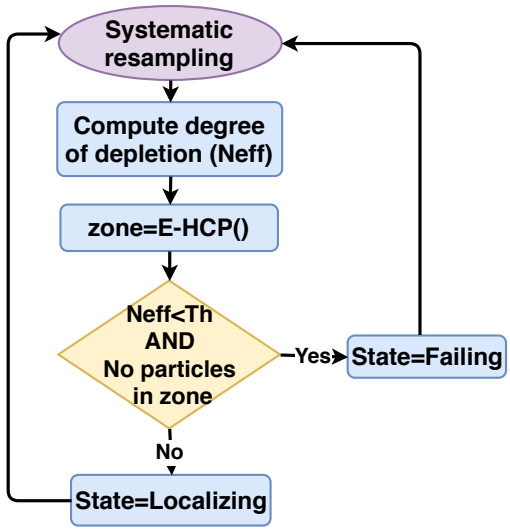


Figure 4: PFRL system state detection method flowchart.

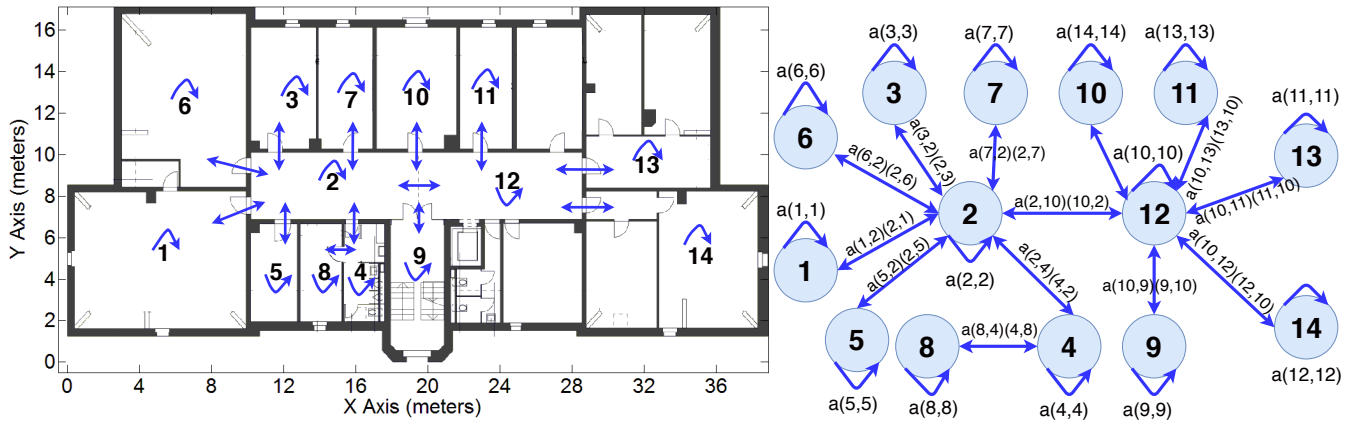
where w^i is the particle's associated weight.

3) System State Detection Method : Q-learning is a method that evaluates which action to perform based on the state of system. Therefore, detecting the current system state is an essential requirement. The system moves from the Starting state to the Localizing state when the starting position is established (i.e., when particles converge to the initial starting position). Thus, the LA must perform the optimal action for fast and accurate convergence. The Starting state is clearly identifiable. When a localization failure is detected, the LA must perform the optimal action for a quick and accurate recovery.

To detect the current System state, we propose a novel and effective method, which is based on the zone prediction information provided by the E-HCP method, the current particle distribution and the current degree of depletion in the particle filter method. Therefore, after performing the systematic resampling process, the system state detection method is executed. Thus, if any particle is placed in the predicted zone, and the effective number of samples is lower than a predefined threshold T_h, the algorithm assumes a localization failure. Figure 4 shows the flowchart of the System state detection method.

F. Data flow between Mobile Client and Edge Server

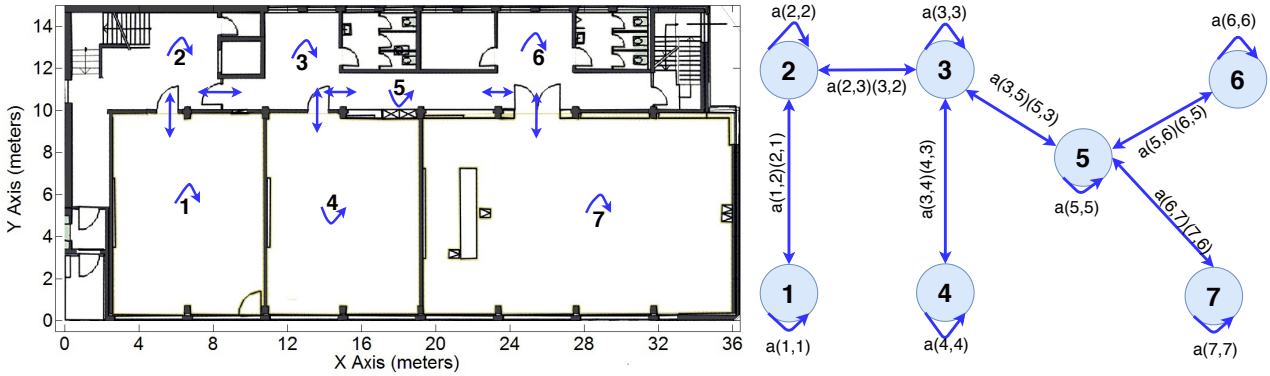
Mobile clients collect raw data using on-board sensors, such as accelerometer, gyroscope, magnetometer, Wi-Fi signals, etc. Such data are processed on mobile devices using lightweight machine learning algorithms, such as HMM-based zone prediction (Section III.A). Afterwards, mobile clients could derive the zone information with probabilities, motion vectors, and ranges to relevant beacons. These information are then transmitted to the edge server using the WebSocket for further processing. At the edge server, heavy machine learning approaches, including ensemble learning (Section III.D) and reinforcement learning (Section III.E), will be applied on the received information to estimate the accurate indoor positions.



(a) Physical layout, zone definition and graphical transition model in scenario 1

(b) Transition model for E-HCP in scenario 1

Figure 5: Scenario 1: Transition information among zones are used to define the transition model for E-HCP method.



(a) Physical layout, zone definition and graphical transition model in scenario 2

(b) Transition model for E-HCP in scenario 2

Figure 6: Scenario 2: Transition information among zones are used to define the transition model for E-HCP method.

As shown in Figure 1, communication between the client and the edge layer is implemented by using the WebSocket technology. WebSocket is a computer communication protocol, which allows two or more connected devices to communicate with one another in both directions through a single TCP connection. It is supported by many platforms. WebSocket technology uses the HTTP upgrade header to change from the HTTP to the WebSocket protocol [9]. Thus, Tornado [38] was used to provide web server and WebSocket server in the cloud layer.

IV. Implementation

We implemented the proposed system on edge servers and smartphones. It comprises three main components: a mobile target (MT), a set of Wi-Fi Anchor Nodes (ANs), and an edge server. ANs are off-the-shelf Wi-Fi access points, which are placed at certain locations to guarantee the maximum coverage for the indoor areas. We used a Motorola Nexus 6 smartphone with 3 GB RAM and Quad-core 2.7 GHz CPU as experimental device. A HP EliteBook with 8 GB RAM and 2.30 GHz Intel Core i5-5300U processor is used as edge server.

Communication between the edge layer and the client layer (i.e., MT) is implemented by using WebSocket technology [9]. WebSocket is a communication protocol to allow connected devices to communicate in both directions by using a single TCP connection. The positioning algorithm (i.e. Particle filter) and the reinforcement learning-based method (i.e., Q-learning) are implemented at the edge server by using Python 2.7.

In addition, the coarse-grained information about the area of interest (i.e., indoor floor plan) is of great importance to guarantee system performance. The system requires information related to physical connections among zones, i.e., connectivity among zones. We define 14 zones for scenario 1 and 7 zones for scenario 2 in our areas of interest (details of the scenarios can be found in Section V). Each zone is a wall separated subarea (e.g., corridor, rooms). The transition model is defined based on the zone distribution. Figures 5 and 6 show the physical layout of the indoor environment, zone distribution and the transition model built on top of the indoor layout. The probability transition matrix T is set based on the assumption

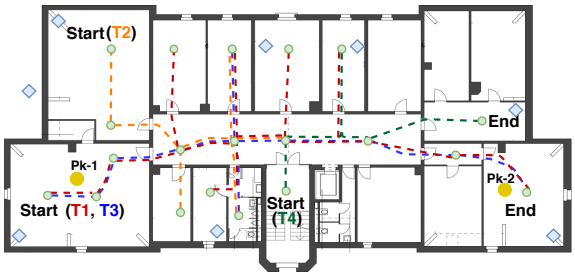


Figure 7: Trajectories definition in Scenario 1, circle yellow points are the kidnapped robot check points.

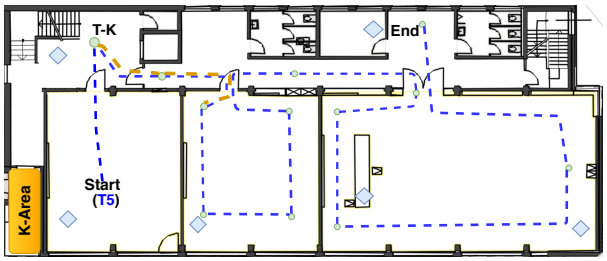


Figure 8: Trajectories definition in Scenario 2, yellow region is the area where particles are kidnapped.

that the probability of staying in the same zone is higher than transferring to another one.

To ensure independence between the individual learning methods in E-HCP, we set up three conceptually different machine learning algorithms (KStar, Multilayer Perceptron (MLP) and J48). Weka for Android library [18] was used to implement the individual machine learning algorithms at the Client layer. To build the zone fingerprinting database, we collected 11200 fingerprint instances, approximately 800 in each zone. The structure of a fingerprint instance consists of Wi-Fi RSS and MF readings. We ask a person to walk randomly through each zone holding the phone in her hand. Zone fingerprint database entries were collected equally distributed over the whole area in each zone. The data collection rate is only constrained by the computational capabilities of the Wi-Fi sensor of the MT. Thus, in our experiments every fingerprinting entry was collected at a rate of 3 entries/second. Since our approach does not need to predefine any survey point, the time needed to build the fingerprinting database is proportional to the number of collected instances multiplied by the instance collection rate. Machine learning algorithms have internal parameters that are optimized during the training process. Nevertheless, some algorithms have internal parameters that are not optimized during the training process. These parameters are named hyperparameters, which have a significant impact on the machine learning algorithms' performance. Therefore, we use a nested cross validation approach to choose the optimized hyperparameter values [31]. To reduce the negative impact of environmental changes and different hardware, we use differential Wi-Fi RSS instead of absolute raw values [40].

V. Performance Evaluation

We made intensive experiments two buildings of the Institute of Computer Science at the University of Bern. The first scenario is an office-like environment with an area of $702m^2$ ($39m$ x $18m$). The second scenario is a classroom-alike scenario with an area of $524m^2$ ($36.2m$ x $14.5m$). The MT is held by a person moving along five different trajectories. Every time when a new fingerprint measurement is available, the zone detection method is launched. Figures 7 and 8 show the physical layout of experiment areas, where trajectories are dotted lines, circle green points are the position checking points, diamond blue points are the anchor nodes.

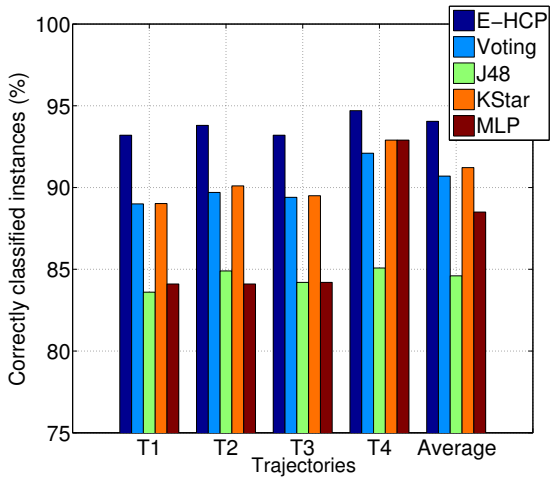


Figure 9: Zone Prediction Performance, Accuracy.

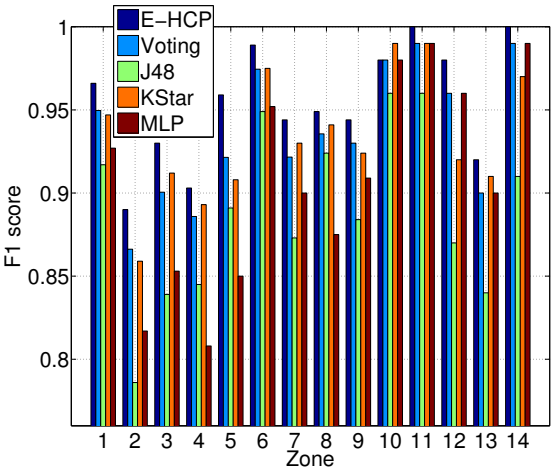


Figure 10: Zone Prediction Performance, F1 score.

A. Indoor Zone Prediction Results

In this section, we present the indoor zone prediction results of the E-HCP method. This experiment was performed in scenario 1. We consider prediction accuracy, sensitivity, precision, and F1 score [19]. Accuracy is defined as the ratio of correctly predicted observations to the total observations. Sensitivity is defined as the number of true positives (TP) divided by TP and the number of false negatives (FN): TP/(TP+FN). Precision is defined as TP divided by TP and the number of false positives

(FP): TP/(TP+FP). F1 is the harmonic mean of sensitivity and precision. Therefore, F1 considers both performance measures, sensitivity, and precision. F1 can be expressed as follows:

$$F1 = 2 \cdot \frac{sensitivity \cdot precision}{sensitivity + precision} \quad (24)$$

We compare the E-HCP method to another ensemble learning algorithm. Thus, we implemented a soft voting machine learning algorithm [34], which is referred to as Voting method hereafter. The Voting method computes the average predicted probabilities of KStar, J48, and MLP. Figures 9 and 10 present the model prediction accuracy and F1 score in scenario 1.

Figure 9 depicts the indoor zone prediction accuracy of the five predictors following four different trajectories in scenario 1. Performance accuracy of the individual machine learning algorithms (J48, KStar, and MLP) is higher than 81% in the four trajectories. However, results show a clear improvement of E-HCP compared to the individual learning and the Voting algorithms. The accuracy of E-HCP is improved by 9.3%, 9.2%, 4.1%, and 9.2% compared to Voting, J48, KStar, and the MLP method respectively.

Since both FN and FP predictions lead to increase the localization error, it is necessary to evaluate the prediction performance by considering these two metrics together. Therefore, the reliability of the zone prediction method is determined by the F1 score. Considering F1, E-HCP outperforms others in all tested zones but zone 10 (see Figure 10 for scenario 1). It means that the predictive reliability of E-HCP is higher than the other tested learning methods.

In indoor environments, measured RSSI values vary according to locations. However, these variations are expected to remain small at nearby positions. For example, at locations close to zone borders, high similarities will be observed on Wi-Fi RSSI values. These similarities could lead to misclassification problems. The E-HCP algorithm outperforms KStar, J48, MLP, and Voting in terms of accuracy and F1 score. However, we can observe in Figure 10 that the KStar algorithm shows high prediction performance too. This is because as an instance-based learner algorithm, KStar uses entropy as a distance measure to describe the similarities of two instances. The distance between two instances can be defined as the complexity of transforming one instance into another. Thus, this method is very sensitive to slight instance variations. In contrast to KStar, J48 uses the entropy of information at the attribute level to build the classification model. This means that J48 calculates entropy using attribute domain information to decide which attribute should be considered in a decision node. Therefore, J48's classification model is prone to misclassification in this specific zone prediction problem. It explains the low performance of J48 compared to the other tested predictors.

Although, MLP prediction performance is lower than E-HCP, KStar, and Voting, we observe in Figure 10 that in some zones MLP achieves higher sensitivity and precision than the other tested algorithms. This is because MLP is able to extract patterns and detect trends that are too complex to be noticed by either humans or other machine learning algorithms.

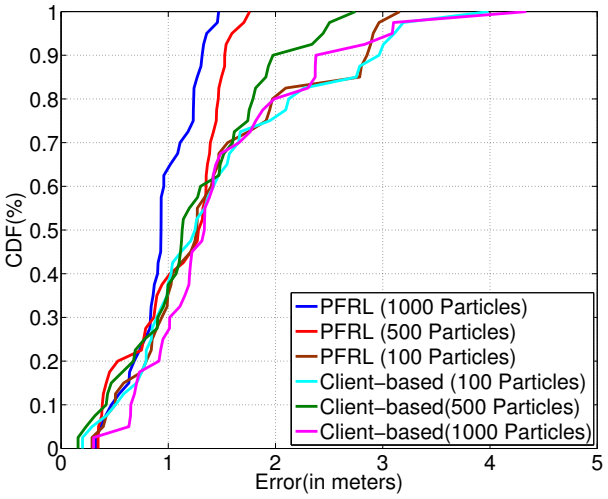


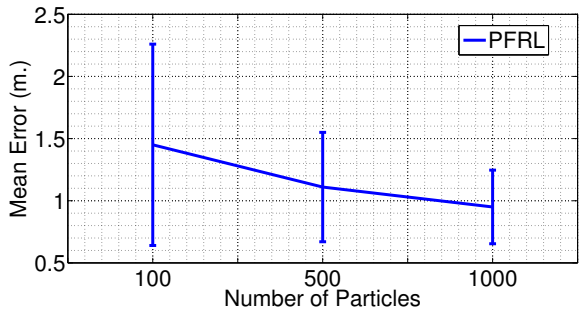
Figure 11: Impacts of particle numbers on performance of PFRL and client-based PFRL in scenario 1.

E-HCP balances out strengths and weaknesses of its constituent algorithms (KStar, J48, MLP). Moreover, E-HCP includes transition information among zones in the predicting process. However, when transition information is ambiguous (i.e., same probability of moving to multiple zones), E-HCP relies only on the prediction performance of its constituent algorithms. This explains why in zone 10 we observed that E-HCP is outperformed by one of its constituent algorithm considering the F1 score.

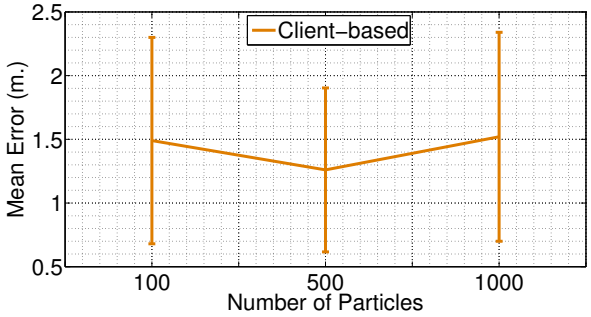
Unlike Voting, E-HCP combines zone transition information with individual machine learning algorithms to improve prediction accuracy. Therefore, by considering the transition probabilities among zones and using ensemble learning techniques, we achieve high zone prediction performance. Moreover, experiment results show that the E-HCP method is able to provide more reliable zone prediction information than the other tested machine learning algorithms. Therefore, our ensemble prediction model allows the production of better prediction performance compared to ensemble voting and individual models.

B. Indoor Tracking Accuracy

In this section, we discuss the tracking performance of the PFRL algorithm. Additionally, to show the benefits of our distributed localization approach using a two-layer architecture, we compare the performance between our two layer-based PFRL with a client-based version of PFRL, where all the computations are hosted on mobile devices. Hereafter, we will refer to the two layer-based PFRL system as PFRL. The Client-based PFRL will be referred to as the client-based PFRL (Client-based). To evaluate the system performance, we consider the metrics of Cumulative Distribution Function (CDF) of localization errors, mean tracking error, the standard deviation of localization errors, and average processing time to get the position.



(a) PFRL (distributed) Confidence Interval.



(b) PFRL (client-based) Confidence Interval.

Figure 12: PFRL Confidence Intervals.

1) Localization Accuracy vs Number of Particles: Localization accuracy can be theoretically boosted by using more particles [17]. However, increasing the number of particles leads to an increased computational complexity of the application too. A large number of particles produces computational inefficiency and high memory request. Figure 11 shows the CDF of localization errors for PFRL and Client-based with different particle numbers in scenario 1.

PFRL achieves better performance when using 1000 particles compared to when using 100 and 500 particles. PFRL is able to reduce the mean localization error in a 24% by increasing the number of particles from 100 to 500, whereas by increasing the number of particles from 500 to 1000, PFRL reduces the mean localization error in 14%. As shown in Figure 12a, PFRL also reduces the confidence interval by increasing the number of particles. The standard deviation is reduced by 64% and 45% when increasing the number of particles from 100 to 500 and from 100 to 1000 respectively. Although PFRL decreases 34.5% the mean localization error when increasing from 100 to 1000 particles, the mean localization error improvement is only about 14% when particles are increased from 500 to 1000. Therefore, it is expected that after a certain number of particles, the mean localization error is not further improved.

Regarding the client-based PFRL approach, the best performance is achieved when 500 particles are used, which is better than using 100 and 1000 particles (also presented in Table I). As shown in Figure 12b, client-based PFRL reduces the confidence interval by increasing the number of particles from 100 to 500. However, the confidence interval is increased when 1000 particles are used. To explain this behaviour, we look at the negative influence produced by increasing processing time in real-time systems. The efficiency of real-time systems depends not only on the precise results but also on the time to get these results. In real-time localization, high processing time could lead the system to stay processing a position while the ground truth position is changing. Therefore, clearly in real-time localization applications, processing time influences the accuracy performance of the system. In the client-based method, we noticed that the average processing time seems to grow exponentially with the number of particles (see Figure 13). Therefore, there is no more performance improvement when a certain particle number is used, due to the negative influence of the exponential growth of the processing time.

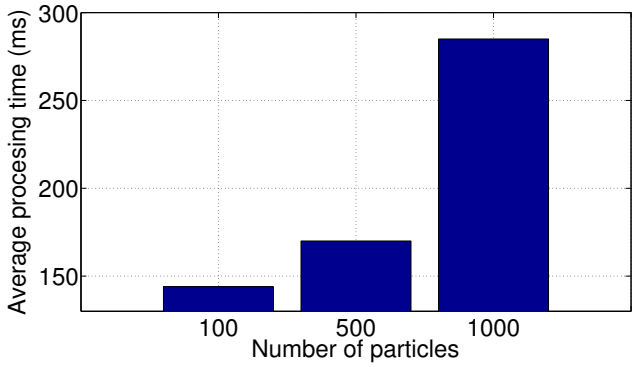


Figure 13: Processing time of client-based PFRL solution vs number of particles

Figure 13 shows the average processing time for the client-based PFRL with different particle numbers. As we can see, when using 1000 particles, the average processing time is 290 ms, which is much bigger than the average processing time of 170 ms when using 500 particles. Therefore, due to the limited computation resources available on mobile devices, increasing the number of particles exponentially increase processing time, which leads to lower localization performance. This explains why 500 particles lead to a better accuracy than 1000 particles when using a client-based PFRL solution.

2) Failure Avoidance and Recovery Performance: The global localization and the kidnapped robot problem are used to evaluate the ability of the system to avoid, detect, and recover itself from localization failures. Moreover, this evaluation measures the ability of self-localizing the target when the system is started (i.e., global localization problem). To test the recovery performance of PFRL, we conducted two experiments in this section. The first experiment tests the self-localizing ability of the system when it is started. We refer to this experiment as the Global localization experiment. The second experiment tests the system ability to recover itself from failures when the system is in normal operation (Localizing). We refer to this as the Kidnapping robot experiment.

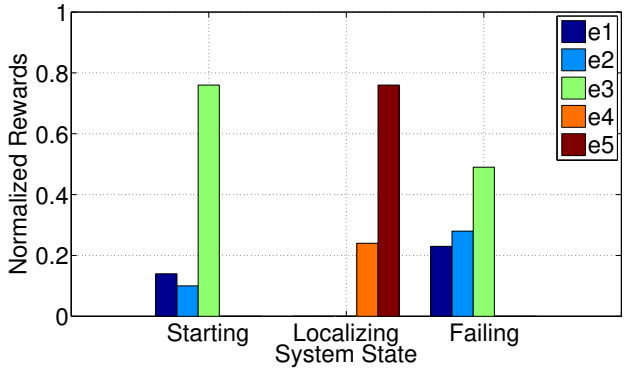


Figure 14: System State vs Normalized Rewards (Normalized Q-table)

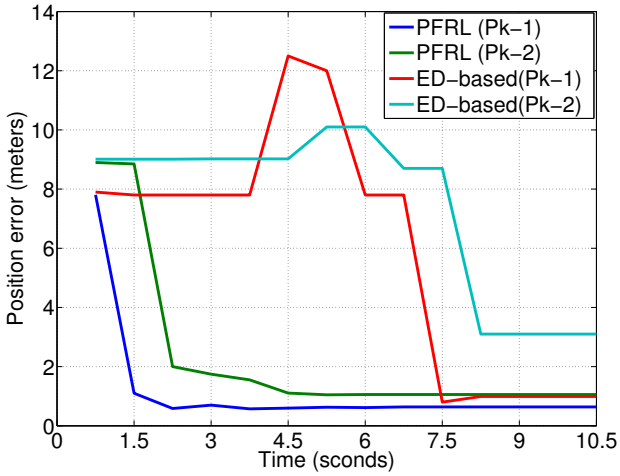


Figure 15: Scenario 1: Global Localization Failure recovery. Particles convergence time after localization failure

The **Global localization experiment** was performed in scenario 1. In this experiment, we simulate localization failures by setting up the initial position of the set of particles at an arbitrary position outside the area of interest. Afterwards, the pedestrian started the system standing at a known position (these positions are shown in Figure 7). We repeated the experiment in three different zones. The derived position was registered in each iteration. Figure 14 shows the normalized rewards (normalized rewards from Q-table) learned in each state. It can be seen that the reinforcement learning method defines *e3* as the best action to perform at states Starting and Failing, and action *e5* is defined as the best action to be executed when the system is at Localizing state. Therefore, to recover the system from localization failures, particles are spread based on the zone probability distribution predicted by the E-HCP method. To avoid localization failures, the 10% worst evaluated particles are spread based on the zone probability distribution predicted by the E-HCP method.

As it can be seen in Figure 15, the average number of iterations to recover the system from localization failures is 4. Each iteration is processed when new Wi-Fi information is available. Since our MT has a Wi-Fi sampling rate up to 3Hz, the latency to recover the system is approximately $1.5s$. This means that the initial position is determined approximately $1.5s$ after the system is started. Moreover, if a localization failure occurs during the tracking process, the system can be automatically recovered with an acceptable time latency of $1.5s$. Regarding the number of iterations, this current approach outperforms by 70% to the equally distributed (ED) localization recovery method presented in [5].

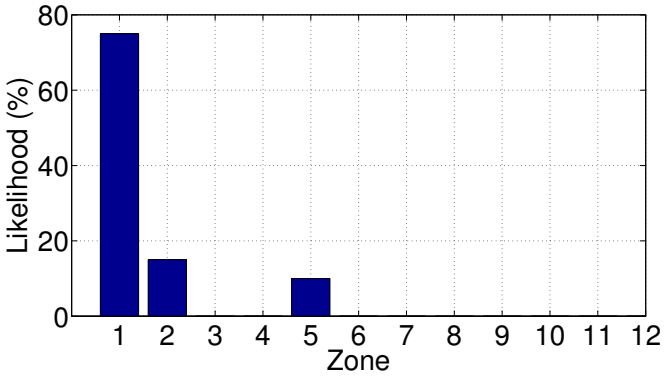
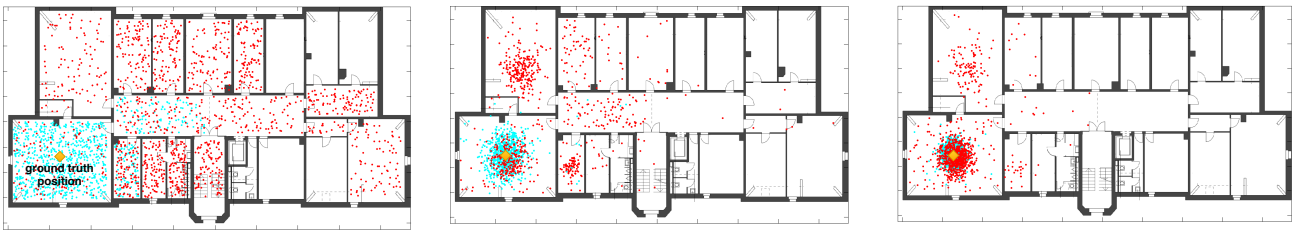


Figure 16: E-HCP Zone probability distribution during the Global localization experiment in scenario 1.

To present PFRL's capability to generate fast particle convergence, we show the physical locations of particles during a localization procedure. First, we present the zone probability distribution results of E-HCP, which has a consequence on the particle convergence speed. Figure 16 shows the zone distribution when the system was started at Pk-1 position during the Global localization experiment. According to the normalized rewards table (Figure 14) and the zone probability distribution (Figure 16), 75% of the particles are distributed over zone 1, whereas 15% and 10% of the particles are distributed over zone 2 and 5 respectively. Thus, more particles are generated in the zones of large likelihood of containing the ground truth location. Unlike ED-based, PFRL focuses on the exploration to high-probability zones. Figure 17 shows the physical distributions of the particles in the Starting state after a localization failure recovery procedure is triggered. The ground truth position is located at Pk-1 (see Figure 7). The cyan points represent the particles of the PFRL method, whereas the red points depict particles of the ED-based method. As mentioned, in PFRL particles are distributed based on the normalized rewards table (Figure 14). In the Starting state, the particles are distributed based on action $e3$. Therefore, to recover the system from the global localization problem, particles are distributed according to the zone probability distribution given by the zone prediction method E-HCP. Consequently, particles in PFRL converge faster than ED-based method, which leads to faster failure recovery in PFRL.



(a) Particles distributions after 0.5 s when a localization failure recovery happens

(b) Particle distributions after 1.2 s when a localization failure recovery happens

(c) Particles distributions after 1.6 s when a localization failure recovery happens

Figure 17: Particle distributions in PFRL and ED-based approaches. The cyan points represent PFRL particles; the red points represent ED-based particles; the diamond yellow point represent the ground truth position.

The **Kidnapping robot experiment** was performed in scenario 2. The goal of this experiment is to show the failure recovery capability of our algorithm. In this experiment, we simulate localization failures by kidnapping all the particles to a predefined area in the environment. We refer this area as the K-area (yellow region in the left bottom corner in Figure 8). After 25 seconds in normal localization operation along trajectory T-K (trajectory T-K and K-area are depicted in yellow color in Figure 8), the system is kidnapped to the K-area. We registered the mean localization errors and time to recover the system from the localization failure. We compare the recovery performance of PFRL to ED-based [5] and PF-based [3]. Figure 18 depicts mean localization error over time for the three tested localization methods in scenario 2. PFRL recovers around 5 seconds after the failure. ED-based approach recovers around 11 seconds after the failure. PF-based (without recovery) solution never recover from the failure. It is worth to mention that the main difference between the Global localization and Kidnapping robot experiments is that in the latter the system must detect the failure to execute the recovery method. In the Global localization experiment, the system is aware that it is starting. Thus, the recovery method is launched immediately after the system starts. Therefore, in the Kidnapping robot experiment, we test the performance of the System State Detection Method presented in section III-E3 and the Reinforcement Learning method for robust tracking resampling presented in section III-E. As it can be seen in Figure 18, PFRL overcomes by around 50% to the equally distributed method presented in [5]. Unlike ED-based, PFRL implements an effective method to detect localization failures. This method is based on the E-HCP method for zone prediction. Thus, if any particle is placed in the current predicted zone, the system assumes a failure. Thus, a kidnapping robot problem is detected immediately when it occurs. Moreover, PFRL implements an effective reinforcement learning-based method for recovering the system from failures. Therefore, after detecting a failure, the system is recovered by sampling particles based on the normalized rewards (see Figure 14) that are learned by the reinforcement learning method. This allows to detect and recover the system in around 5 seconds.

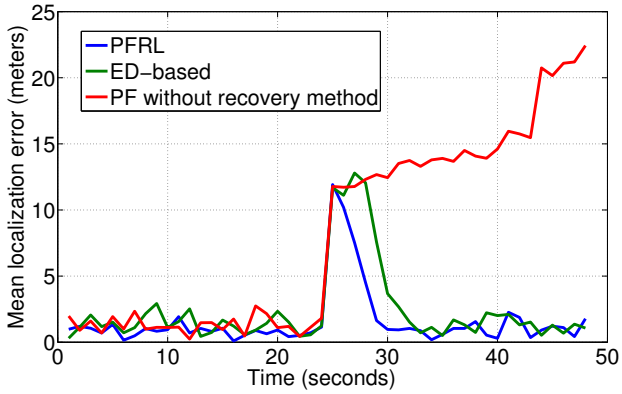


Figure 18: Scenario 2: Kinapping Robot Problem. Particles convergence time after a localization failure is detected.

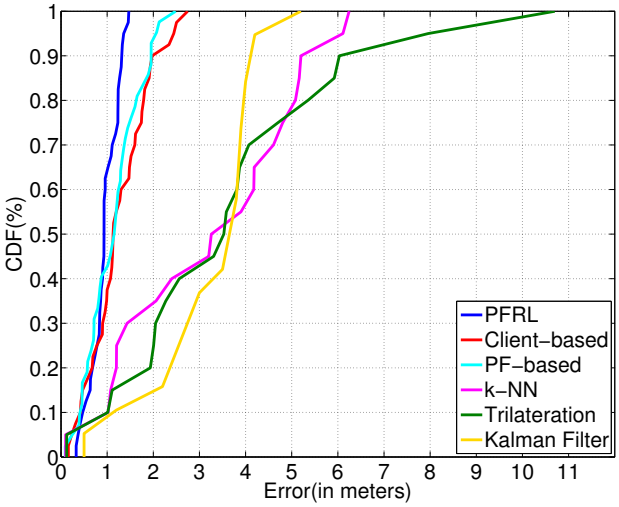


Figure 19: Scenario 1: Localization error CDF of PFRL (1000 particles), Client-based (500 particles) NLS, k-NN (k=3) and KF.

Table I: Scenario 1: Localization methods performance

Configuration	Mean error	S.D	90% Acc.
PFRL (100 Ptc.)	1.45m	0.81m	2.9m
PFRL (500 Ptc.)	1.11m	0.45m	1.5m
PFRL (1000 Ptc.)	0.97m	0.3m	1.3m
Client-based (100 Ptc.)	1.493m	0.907m	3.1m
Client-based (500 Ptc.)	1.267m	0.645m	2.0m
Client-based (1000 Ptc.)	1.515m	0.8188m	2.9m
PF-based (1000 Ptc.)	1.15m	0.61m	2.1m
NLST	3.79m	2.52m	8.0m
k-NN (k=3)	3.32m	1.89m	6.1m
Kalman Filter	3.36m	1.11m	4.1m

3) Best Localization Performance comparison with other systems: It is difficult to fairly compare our approach with other state-of-the-art localization approaches (e.g., fingerprinting-based, landmark-based, range-based). This is because indoor localization system performance are environmental-dependent (i.e., they rely on the presence of numerous landmarks in the environment), and it is impossible to duplicate the exact indoor environments in another indoor areas. Moreover, it is rather hard to implement all the specific details of an existing solution and repeat the identical experiment to get the same results that were collected in another physical indoor environment. Similar to other localization systems, we compare the performance of our localization approach with the k-Nearest Neighbors (k-NN), Nonlinear Least Squares Trilateration (NLST), and Kalman Filter-based (KF) localization methods. Morevover, we compare PFRL to another basic particle filter-based (PF-based) approach which is presented in [3]

Figure 19 shows the CDF of localization errors for the best performance of PFRL, Client-based, PF-based, k-NN (k=3), NLST, and KF methods. Table I summarizes the results, which show that NLST achieves the worst localization performance with around $8.0m$ for 90% accuracy. PFRL overcomes NLST by approximately 83.7% and 74.4% regarding 90% accuracy and mean error respectively. Moreover, the standard deviation of PFRL is 88.09% smaller than NLST. k-NN achieves around $6.1m$ for 90% accuracy with the mean error of $3.32m$ and the standard deviation of $1.89m$. The KF approach achieves a 90% accuracy of $4.1m$, the mean error is $3.36m$ and the standard deviation is $1.11m$. Therefore, PFRL overcomes k-NN by around 78.68%, and KF by around 68.29% considering 90% accuracy. The mean error of the PFRL approach is 71.1% and 70.78% better than for KF, and k-NN respectively. Experiment results show that PFRL outperforms Client-based, KF, NLST, and k-NN for accuracy and stability. Although PF-based achieves high localization performance, PFRL outperforms PF-based by around 15.7% and 24.6% considering mean localization error and standard deviation respectively. This is because PF-based does not have any method to identify if the AN and the mobile client are located at the same zone. Thus, the same ranging method is adopted for all the ANs in PF-based method. Unlike PF-based, PFRL includes a zone prediction method, which supports to choose the proper ranging model for each zone. Therefore, PFRL outperforms traditional particle filter and fingerprinting-based localization methods (e.g., k-NN) by combining range-based localization methods and fingerprinting models.

Table II: Scenario 2: Localization methods performance

Configuration	Mean error	S.D	90% Acc.
PFRL (1000 Ptc.)	0.98m	0.49m	1.8m
Client-based (1000 Ptc.)	1.3m	0.65m	2.1m
PF-based (1000 Ptc.)	1.34m	0.7m	2.1m
NLST	4.57m	1.9m	7.0m
k-NN (k=3)	3.83m	1.71m	6.1m
Kalman Filter	3.29m	1.24m	4.9m

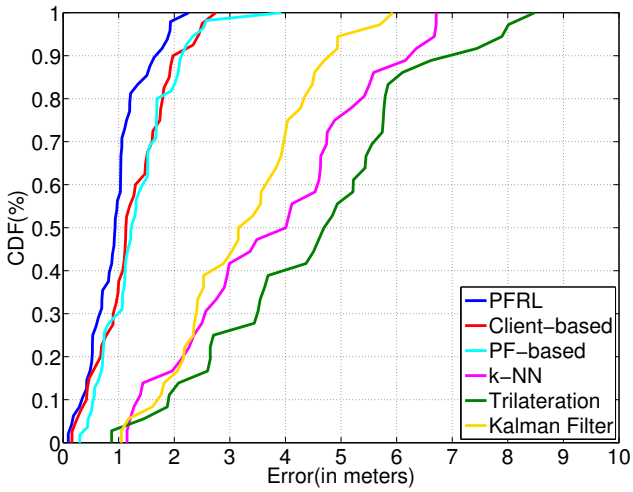


Figure 20: Scenario2: Localization error CDF of PFRL (1000 particles), Client-based (500 particles) NLS, k-NN and KF.

4) Performance vs Area of Interest: To validate the environment independence of PFRL, we chose a second scenario to deploy the localization system. As mentioned in previous sections (Section V), scenario 2 is a classroom-alike indoor scenario at the University of Bern with an area size of $524m^2$. We set up PFRL with the configuration that achieved the best performance on experiments executed in scenario 1. Figure 8 depicts scenario 2 and trajectory 5, which was used to test the localization approaches. Table II summarizes the mean tracking error, standard deviation and 90% accuracy. Figure 20 shows the CDF of localization errors for the best performance of PFRL, Client-based, PF-based, k-NN (k=3), NLST and KF methods. PFRL achieves around $0.98m$ for mean localization error, which outperforms Client-based, PF-based, NLST, K-NN and KF by around 24.6%, 24.8%, 78.5%, 74.1% and 70.2% respectively. Although the high localization performance of PFRL in scenario 2, the mean error, standard deviation and 90% accuracy were slightly increased compared to experiments in scenario 1. This reflects that the density of ANs along the area of interest influence the localization performance. In scenario 1 we deployed 8 ANs, whereas in scenario 2 we deployed 7 ANs. As a conclusion, the proposed PFRL could guarantee best performance in multiple indoor areas.

VI. Conclusions

This work proposed a particle filter-based reinforcement learning (PFRL) approach for the autonomous robust wireless indoor positioning system. The system is validated on a distributed machine learning-based network architecture, which includes a client layer and an edge layer. The client layer includes mobile devices that host lightweight ML algorithms (supervised ML algorithms) to recognize zones, while the edge layer includes edge server that hosts heavy machine learning operations to run complex particle filter and reinforcement learning calculations. The PFRL algorithm includes several components. An efficient ensemble predictor that could achieve high zone prediction performance by integrating HMM with discriminative learning techniques. Thanks to the fusion of fingerprinting and zone transition information, our zone prediction method outperforms traditional fingerprinting approaches. Additionally, we proposed an efficient probabilistic model to fuse zone detection, radio-based ranging, IMU, and floor plan information to provide stable and accurate indoor localization. A reinforcement learning approach is applied on top of the proposed particle filter to improve the system robustness against positioning failures. We evaluated our localization system in two complex real-world indoor environments. Evaluation results show that our proposed method can deliver more accurate localization results and is more robust to localization failures than traditional indoor localization methods. Thanks to the reinforcement learning approach, the proposed PFRL solution could make the localization system converge much faster than other systems without failure recovery mechanism.

Acknowledgment

This work was partly supported by the Swiss National Science Foundation via the Intelligent Mobility Services project (200021_184690).

References

[1] H. Abdelnasser, R. Mohamed, A. Elgohary, M. Alzantot, H. Wang, S. Sen, R. Choudhury, and M. Youssef, "Semanticslam: Using environment landmarks for unsupervised indoor localization." *IEEE Transactions on Mobile Computing*, vol. 15, pp. 1770–1782, 2016.

[2] P. Bahl and V. Padmanabhan, "Radar: an in-building rf-based user location and tracking system." *Joint Conference of the IEEE Computer and Communications Societies. Proceedings*, vol. 19, 2000.

[3] J. Carrera, Z. Li, Z. Zhao, and T. Braun, "A real-time indoor tracking system in smartphones." *Proceedings of the 19th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM)*, November 2016.

[4] J. Carrera, Z. Zhao, and T. Braun. (2018) Room recognition using discriminative ensemble learning with hidden markov models (submitted). [Online]. Available: http://arxiv.org/abs/1804.09005

[5] J. Carrera, Z. Zhao, T. Braun, Z. Li, and A. Neto, "A real-time robust indoor tracking system in smartphones," *Elsevier Computer Communications*, vol. 117, pp. 104–115, February 2018.

[6] J. Carrera, Z. Zhao, T. Braun, H. Luo, and F. Zhao. (2018) Discriminative learning-based smartphone indoor localization (submitted). [Online]. Available: http://arxiv.org/abs/1804.03961

[7] F. G. D. Hol, B. Schon. (2018) On resampling algorithms for particle filters. [Online]. Available: http://people.isy.liu.se/rt/schon/Publications/HolSG2006.pdf

[8] A. Doucet, S. Godsill, and C. Andrieu, "On sequential monte carlo sampling methods for bayesian filtering," *Statistics and computing*, vol. 10, no. 3, pp. 197–208, 2000.

[9] I. Fette and A. Melnikov, "The websocket protocol," *Internet Engineering Task Force (IETF)*, 2011.

[10] G. Forney, "The viterbi algorithm," *Proceedings of the IEEE*, vol. 61, pp. 268–278, March 1973.

[11] D. Fox, W. Burgard, and S. Thrun, "Markov localization for mobile robots in dynamic environments," *J. Artif. Int. Res.*, vol. 11, no. 1, pp. 391–427, Jul. 1999.

[12] D. Fox, W. Burgard, F. Dellaert, and S. Thrun, "Monte carlo localization: Efficient position estimation for mobile robots," *AAAI/IAAI*, vol. 1999, no. 343-349, pp. 2–2, 1999.

[13] S. He and G. Chan, "Wi-fi fingerprint-based indoor positioning:recent advances and comparisons," *IEEE Communications Survey Tutorials*, 2016.

[14] S. He, S. G. Chan, L. Yu, and N. Liu, "Calibration-free fusion of step counter and wireless fingerprints for indoor localization," *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp 2015)*, 2014.

[15] S. Hilsenbeck, D. Bobkov, G. Schroth, R. Huitl, and E. Steinbach, "Graph-based data fusion of pedometer and wifi measurements for mobile indoor positioning," *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp 2014)*, 2014.

[16] F. Hong, H. Chu, L. Wang, Y. Feng, and Z. Guo, "Pocket mattering: Indoor pedestrian tracking with commercial smartphone," *International Conference on Indoor Positioning and Indoor Navigation*, 2012.

[17] F. Hong, Y. Zhang, M. Wei, Y. Feng, and Z. Guo, "Wap: Indoor localization and tracking using wifi-assited particle filter," *39th IEEE Conference on Local Computer Networks*, pp. 210–217, 2014.

[18] j. Marsan. (2018) Weka-for-android. [Online]. Available: https://github.com/rjmarsan/Weka-for-Android

[19] R. Joshi. (2016, Mar.) Accuracy, precision, recall and f1 score: Interpretation of performance measures.

[20] D. Kamisaka, T. Watanabe, S. Muramatsu, A. Kobayashi, and H. Yokoyama, "Estimating position relation between two pedestrians using mobile phones," in *Pervasive Computing*, J. Kay, P. Lukowicz, H. Tokuda, P. Olivier, and A. Krüger, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 307–324.

[21] G. Kitagawa, "Monte carlo filter and smoother and non-gaussian nonlinear state space models," *Journal of Computational and Graphical Statistics*, vol. 5, pp. 1–25, 1996.

[22] M. Kourogi, N. Sakata, T. Okuma, and T. Kurata, "Indoor/outdoor pedestrian navigation with an embedded gps/rfid/self-contained sensor system," *Advances in Artificial Reality and Tele-Existence, 16th International Conference on Artificial Reality and Telexistence (ICAT 2006)*, 2006.

[23] Z. Li, T. Braun, X. Zhao, and Z. Zhao, "A narrow-band indoor positioning system by fusing time and received signal strength via ensemble learning," *IEEE Access 2018*, pp. 9936–9950, 2018.

[24] Z. Li, D. Dimitrova, and T. Braun, "A passive wifi source localization system based on fine-grined power-based trilateration." *IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, 2015.

[25] ——, "A time-based passive source localization system for narrow-band signal," *The IEEE International Conference on Communications (ICC)*, vol. 39, pp. 4599–4605, June 2015.

[26] J. Liu and R. Chen, "Sequential monte carlo methods for dynamic systems," *Journal of the American Statistical Association*, vol. 93, 1998.

[27] J. Liu, L. P. R. Chen, R. Guinness, and H. Kuusniemi, "A hybrid smartphone indoor positioning solution for mobile lbs," *Sensors*, 2012.

[28] H. Mao, M. Alizadeh, I. Menache, and S. Kandula, "Resource management with deep reinforcement learning," in *Proceedings of the 15th ACM Workshop on Hot Topics in Networks*. ACM, 2016, pp. 50–56.

[29] F. Melo. (2018) Convergence of q-learning: a simple proof. [Online]. Available: http://users.isr.ist.utl.pt/ mtjspaan/readingGroup/ProofQlearning.pdf

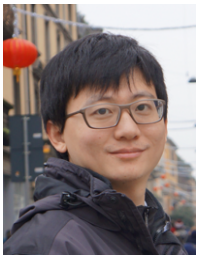
[30] M. Michael, T. Sebastian, K. Daphne, and W. Ben, "Fastslam 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges," in *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI)*, vol. 2, 2003.

[31] K.-R. Müller, M. Krauledat, G. Dornhege, G. Curio, and B. Blankertz, "Machine learning techniques for brain-computer interfaces." *BIOMEDICAL ENGINEERING*, pp. 11–22, 2004.
[32] K. N. and S. Kamijo, "Pedestrian dead reckoning for mobile phones through walking and running mode recognition," *in The 16th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pp. 261–267, Oct 2013.
[33] S. Nemati, M. M. Ghassemi, and G. D. Clifford, "Optimal medication dosing from suboptimal clinical examples: A deep reinforcement learning approach," in *Engineering in Medicine and Biology Society (EMBC), 2016 IEEE 38th Annual International Conference of the*. IEEE, 2016, pp. 2978–2981.
[34] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.
[35] S. Seidel and T. Rappaport, "914 mhz path loss predictions models for indoor wireless communications in multifloored buildings." *IEEE Transactions on Antennas and Propagation*, 1992.
[36] R. S. Sutton and A. G. Barto, *Introduction to reinforcement learning*. MIT press Cambridge, 1998, vol. 135.
[37] S. Thrun, W. Burgard, and D. Fox, "Probabilistic robotics," https://docs.ufpr.br/ danielsantos/ProbabilisticRobotics.pdf, 2000.
[38] T. Tornado-Authors. (2018) Tornado: Running and deploying. [Online]. Available: http://www.tornadoweb.org/ en/stable/guide/running.html
[39] M. Wallbaum and O. Spaniol, "Indoor positioning using wireless local area networks," *in Proceedings of the IEEE John Vincent Atanasoff International Symposium on Modern Computing.*, pp. 17–26, Oct 2006.
[40] J. Wang, Q. Gao, H. Wang, H. Chen, and M. Jin, "Differential radio map-based robust indoor localization," *EURASIP Journal on Wireless Communications and Networking*, vol. 2011, p. 12, 2011.
[41] P. Wang and Y. Luo, "Research on wifi indoor location algorithm based on rssi ranging." *17 4th International Conference on Information Science and Control Engineering (ICISCE)*, 2017.
[42] C. Watkins and P. Dayan, "Technical note q-learning," *Machine Learning Technical Note*, vol. 8, pp. 279–292, 1992.
[43] C. Zhang, P. Patras, and H. Haddadi, "Deep learning in mobile and wireless networking: A survey," *arXiv preprint arXiv:1803.04311*, 2018.
[44] D. Zhao, Y. Chen, and L. Lv, "Deep reinforcement learning with visual attention for vehicle classification," *IEEE Transactions on Cognitive and Developmental Systems*, vol. 9, no. 4, pp. 356–367, 2017.

José Luis Carrera Villacrés received the B.S.E. degree from the National Polytechnic School from Equateur and the MSc. degree in computer sciences from the Swiss Joint Master of Science in Computer Science program of the universities of Neuchatel, Fribourg and Bern in 2015. He is currently working toward the Ph.D. degree in the Institute of Computer Sciences of the University of Bern. His research interests include Artificial Intelligence, Machine Learning, indoor localization and distributed systems.

Zhongliang Zhao received the Ph.D. degree from University of Bern, Switzerland in 2014. Since then, he worked as a senior researcher in the same institute. He has been active in the EU FP7 project Mobile Cloud Networking, co-PI of the Sino-Swiss Science and Technology Cooperation project M3WSN, technical coordinator of the SNF project SwissSenseSynergy. Currently he is the co-PI of the Orange-funded industry project Context Awareness Engine. His current research interests include machine learning for wireless communication networks, urban computing, UAV ad-hoc networks, etc.

Torsten Braun got his Ph.D. degree from University of Karlsruhe (Germany) in 1993. From 1994 to 1995, he was a guest scientist at INRIA Sophia-Antipolis (France). From 1995 to 1997, he worked at the IBM European Networking Centre Heidelberg (Germany) as a project leader and senior consultant. Since 1998, he is a full professor of Computer Science at University of Bern. Currently, he has been a vice president of the SWITCH (Swiss Research and Education Network Provider) Foundation from 2011 to 2019. He was a Director of the Institute of Computer Science and Applied Mathematics at University of Bern between 2007 and 2011. He has been serving as Deputy Dean of the Faculty of Science, University of Bern from 2017 to 2019. He received best paper awards from LCN 2001, WWIC 2007, EE-LSDS 2013, WMNC 2014, and the ARMS-CC-2014 Workshop as well as the GI-KuVS Communications Software Award in 2009.



Zan Li received the Ph.D. degree from the University of Bern, Switzerland, in 2016. Afterwards, he was with the Alibaba Group, China, as a Senior Algorithm Engineer from 2016 to 2017. Since 2017, he has been with the College of Communication Engineering, Jilin University, as a Lecturer. He was a recipient of the Best Paper Award from the WMNC 2014. His Ph.D. dissertation won the Fritz-Kutter Award 2016 (the best Ph.D. dissertation in Swiss Universities).