IAES International Journal of Artificial Intelligence (IJ-AI)
Vol. 12, No. 1, March 2023, pp. 328~346
ISSN: 2252-8938, DOI: 10.11591/ijai.v12.i1.pp328-346

A genetic based indoor positioning algorithm using Wi-Fi received signal strength and motion data

Pham Doan Tinh[1], Bui Huy Hoang[1], Nguyen Duc Cuong[2]
[1]Department of Electronics, School of Electrical and Electronics Engineering, Hanoi University of Science and Technology, Hanoi, Vietnam
[2]Big Data Analytics Center, Viettel Corporation, Hanoi, Vietnam

Article Info

Article history:

Received Dec 21, 2021
Revised Sep 1, 2022
Accepted Sep 30, 2022

Keywords:

Genetic
Machine learning
Wi-Fi received signal strength
Global positioning system

ABSTRACT

The recent trend in location-based services has led to a proliferation of studies in indoor positioning technology. Wi-Fi received signal strength indicator (RSSI) Fingerprinting and pedestrian dead reckoning (PDR) are the two best representatives from both approaches. This research proposed a genetic algorithm to combine Wi-Fi Fingerprinting and PDR. By taking advantage of PDR and genetic algorithm, we only need to collect a limited number of points for the fingerprint dataset with known coordinates, then target trajectories' position can be estimated with high accuracy. Results from our experiments and simulations have shown that even in the scenario of noisy inertial measurement unit (IMU) sensors data, using RSSI measurements and the coordinate of 8 points, our proposed method was able to achieve 1.589 meters of average distance error which is 34.4 percent lower than the conventional Fingerprinting method.

Corresponding Author:

Pham Doan Tinh
School of Electrical and Electronics Engineering, Hanoi University of Science and Technology
No. 1 Dai Co Viet street, Hanoi, Vietnam
Email: tinh.phamdoan@hust.edu.vn

1. INTRODUCTION

Indoor positioning (IP) is receiving much attention due to advancements in sensor technology and the high demand for location-based services. Large indoor environments like shopping malls, train stations, and airports are now trying to deploy internet of things (IoT) systems to improve user experience and quality of service. Thus, it opens the opportunity for indoor localization to be a core technology for user's navigation systems in such large indoor spaces.

The global positioning system (GPS) is a popular method for outdoor navigation. However, its indoor performance drops significantly because of blockage in GPS signals when targets are inside a building or underground. Even outdoor, GPS accuracy could be affected by various factors, for example, atmospheric conditions and multipath [1]. Thus, many new emerging techniques for indoor positioning have been studied and proposed. Depend on different sensors technology and infrastructure, IP can be categorized into many approaches [2]–[4].

One popular approach that take advantage of inertial measuring unit (IMU) is pedestrian dead reckoning (PDR) [5]–[7]. Using signals from IMU sensors like accelerometer, gyroscope and magnetometer, step event [8], step length [9], [10] and heading angle [11] are extracted. Because most smartphones are equipped with IMU, PDR does not require indoor infrastructure to be set up. In addition, data is updated frequently, which enable real-time localization. However, in PDR, target's initial location must be known because current location is estimated based on the previous location. Furthermore, sensors in smartphones and other mobile devices are subjected to noise, interference and tend to produce accumulated errors over time [12]. Thus, complex sensor fusion techniques were utilized to complement between sensors or integrate them with other wireless technologies like Wi-Fi or Bluetooth to improve accuracy [13]–[18], popular techniques namely Kalman Filter [19], Extended Kalman Filter [20], Madgwick Filter [17], and Particle Filter [21].

Another popular approach is to use Radio frequency, which includes a wide variety of wireless technology like Wi-Fi, Bluetooth [22], [23], Zigbee [24], Ultra-Wideband [25], radio frequency identification (RFID) [26]. Methods that utilize Wi-Fi is popular since most public or private indoor places have Wi-Fi access points (APs) installed and nearly every mobile device is equipped with a Wi-Fi transceiver module. Many researchers have tried to apply outdoor navigation techniques like triangulation, which is divided into two subgroups (lateration and angulation) [26]. The lateration is based on distance and is used in time of arrival (ToA) [27], [28] while angulation is based on direction and is applied in angle of arrival (AoA) [29], [30]. A well-known technique in Wi-Fi indoor positioning is received signal strength indicator (RSSI) fingerprinting [31]–[34]. In this method, the assumption was made that for every location in the environment, its Wi-Fi received signal strength indication (RSSI) measurements are unique and distinctive from others. Then, signals from those positions are recorded and stored in a fingerprint database called the radio map. After fingerprints are collected, researchers can start measuring the user's Wi-Fi RSSI and match it to the similar point in the radio map to acquire user position. The step of constructing the database is also known as the offline phase and the matching step is known as the online phase. In online phase machine learning methods can be used to learn the pattern and produce more accurate results.

In comparison with other indoor positioning techniques, Wi-Fi fingerprinting precision, deployment cost, and availability are superior. It can be applied to nearly every indoor scenario where a Wi-Fi network is installed. However, there are certain drawbacks associated with the use of Wi-Fi fingerprinting. First, Wi-Fi RSSI is subjected to multipath and shadowing interferences in a noisy environment [35]. Thus, the measurements are no longer stable and reliable. Secondly, the offline phase requires intensive fingerprint collecting which consume a lot of time and effort, especially in a large environment. In addition, adjustments in the interior of the environment make the radio map incorrect and need to be recollected.

In this paper, the problem of Wi-Fi Fingerprinting is addressed by proposing a unique method to combine PDR and Wi-Fi Fingerprinting using the genetic algorithm. The proposed method would reduce greatly the amount of data needed to be collected in the offline phase while achieving a high level of accuracy. Using only a small number of labeled RSSI measurements as anchors, user track's coordinates can be estimated by utilizing RSSI and motion from the user's mobile device. Experiment and simulation have shown that the proposed method using the support vector machine model achieved an average distance error of 1.589 meters in the situation of motion error reaching 30%. In addition, only 8 anchor points were used, and the result compared with conventional Wi-Fi fingerprinting is 34.4% lower.

2. LITERATURE REVIEW

This section discussed the existing indoor positioning methods. We mainly focused on methods without offline fingerprinting map and analyze their advantages along with drawbacks. In [36] Jang and Kim classified methods in this approach into three categories, which are simultaneous localization and mapping (SLAM); inter/extrapolation; and crowdsourcing-based technologies.

2.1. Simultaneous localization and mapping (SLAM)

SLAM is a technique that allows researchers to construct the map of the landscape and estimate location of the target simultaneously without prior knowledge of the environment. The principle of SLAM is to use the previous target's locations, control inputs and landmark observation as the condition for the joint probability distribution of target's state and landmark location at current time. Ferris *et al.* in [37] proposed Wi-Fi-SLAM, which use Gaussian process-latent variable model (GP-LVM) to transform the 3 dimensional data to 2 dimensional coordinate. Instead of using fixed measured data from AP, Brian used them as latent variable then probabilistically modeled the relationship between the latent variable and the observed data from target. The relationship can be recovered using the optimization of the marginal likelihood. Due to the high time complexity of GP-LVM, it would be more suitable for a robot equipped with a dedicated computer than a mobile device. In addition, GP-LVM tend to replace curves in the ground truth with straigh lines in the produced map. GraphSLAM in [38] was proposed to overcome the time complexity and the dependence on signal-rich environments of GP-LVM. Authors of GraphSLAM assume that in any small and free space, the intensity of propagated radio waves are almost equal, this allows GraphSLAM to be applied in a variety of environment. In addition, GraphSLAM achieved O(n^2) asymptotic runtime by omitting the whitening transform on the weighted observations and their weights. However, while applying GraphSLAM to large space, it would need to be interpolated into smaller area because of the signal strength constraint mentioned earlier. The interpolating process, if is not done correctly could affect the performance. WiSLAM in [39] is the integration of two method which are PlaceSLAM [40] and FootSLAM [41]. PlaceSLAM performs indoor localization using RSS values and the location of user is associated with the proximity information. In contrast to using proximity information or landmarks, FootSLAM utilizes probabilistic motion model over space to construct the map of the environment. Since FootSLAM uses IMU, authors countered sensor error by Rao-Blackwellized particle filter (RBPF) [42].

2.2. Extrapolation and interpolation

Triangle interpolation and extrapolation (TIX) [43] creates mapping curves by using at least 3 access points (APs) to measure receive signal strength (RSS) between themselves. Then, server can estimate the distance between target and each AP using the mapping curve and RSS measurement from target's device. A technique called proximity in signal space (PSS) is used to choose the suitable mapping curve. Major drawbacks of this method are the reliance on APs as their location need to be known beforehand and TIX also need at least 3 APs to perform. Furthermore, when the number of APs increases, the computational cost also grows. Signal distance map (SDM) in [44] is quite similar to TIX because they both use RSS to locate client position and attain the mapping function using inter-AP measurements. The main difference comes from a more complex algorithmic computation that author used in SDM. Although the result of SDM is better than TIX, the trade off is the increase in calculation time of SDM.

2.3. Crowdsourcing and crowdsensing

While other techniques focus on exploiting the wireless network infrastructure and sensors, crowdsourcing and crowdsensing takes an approach which take advantage of the crowd wisdom to make decision. User contribution is the most important factor as it affects the performance of a crowdsourcing and crowdsensing system. LocateMe in [45] utilizes and captures the magnetic signatures in hallways while person is walking in the environment. Those signatures are stored in a database similar to fingerprinting database and when a new magnetic signature is collected, it will be classified to one of the fingerprints in the database. The main disadvantage of this method is that magnetic measurements is often inconsistent because of the existence of magnetic material, metal and also electronics devices in the environment. QrLoc in [46] is a crowdsourcing method which substitute the data collection of fingerprinting method by having user scan QR codes attached to the landmarks to mark his position and record the Wi-Fi measurements. Walkie-Markie [47] is a method which based on crowdsourcing to create the map of an indoor environment by using user's trajectory acquired from IMU sensors and Wi-Fi landmarks. Wi-Fi landmark is refered as RSS trend tipping point (RTTP), where signal strength shift from rising to falling. When users move around the environment, mobile device would calculate stride length and direction using motion data while simultaneously scans Wi-Fi signal strength to detect Wi-Fi landmarks. The Wi-Fi landmarks can be created by different tracks from different users and the server would cluster those landmarks into one landmark using clustering algorithm.

Based on the literature review of several indoor positioning method without offline survey fingerprinting, it reveals the strengths and limitations exist in each one of them. Although SLAM doesn't need information about the environment, it requires observable landmark to locate itself. In addition, the reliance on probabilistic model make SLAM perform heavy computation which hinders its ability to run on mobile devices. Extrapolation and interpolation may solve the problem of time complexity of SLAM, but this method require several condition and restriction like knowing the location of APs or the minimum number of APs for the method to perform. With crowdsourcing, the idea to use the participation of user is a great solution to eliminate labor intensive data collecting procedure and this idea was utilized in our proposed method. In addition, the proposed method aim is not to build the map of the environment and it relies on genetic algorithm instead of probabilistic model to determine user position. As the result, our method is simpler to be implemented and doesn't require specific knowledge of APs to perform.

3. RESEARCH METHOD

3.1. Overall system architecture

The proposed method architecture is divided into two main phases, which is Data preparation and Coordinate optimization as illustrated in Figure 1. At first, on the leftmost, the Wi-Fi access points network are set up in the environment and transmit signals to devices denoted as mobile phone. In this step, the collected data is separated into two groups, RSSI measurement points with coordinate denoted as anchors and RSSI measurement points plus motion data without coordinate denoted as user data. Anchors would then be used as the testing dataset while motion data is utilized by genetic algorithm to generate an estimation of user's coordinate. Those coordinates combine with RSSI measurement points from user data serve as the training dataset for machine learning models and error is evaluated using the anchors dataset mentioned earlier. If the calculated error satisfies the convergence condition, user's estimated coordinates are selected.

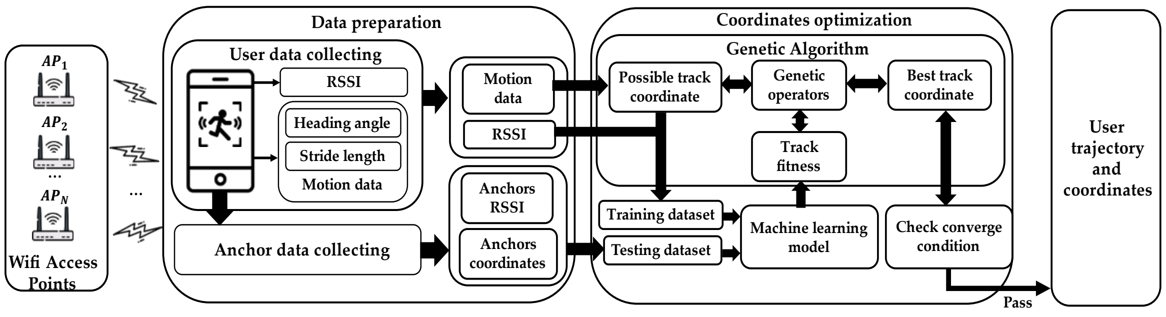

Figure 1. Proposed method system architecture

3.2. Data preparation

The anchor dataset K is collected in a similar way as the radio map of the fingerprinting approach. First, several points in the environment are selected to be anchors. Then a user with a small computing unit equipped with a Wi-Fi transceiver module will go to those designated points and record the RSSI measurements from access points (APs). The process is illustrated in Figure 2, the red markers represent the anchors and for each anchor, RSSI measurements from APs and anchors's coordinate are recorded in form:

$$K_j = \{(x_j, y_j), (RSSI_j^1, RSSI_j^2, RSSI_j^3, \ \dots RSSI_j^i, \ \dots, RSSI_j^M)\} \ j \in [1, N_K], i \in [1, M] \qquad (1)$$

Where:
N_K is the number of anchors point in dataset K
M is the number of access point
$RSSI_k^i$ is the RSSI measurement from AP_i of anchor point i^{th}
(x_i, y_i) is the coordinate of anchor point i^{th}

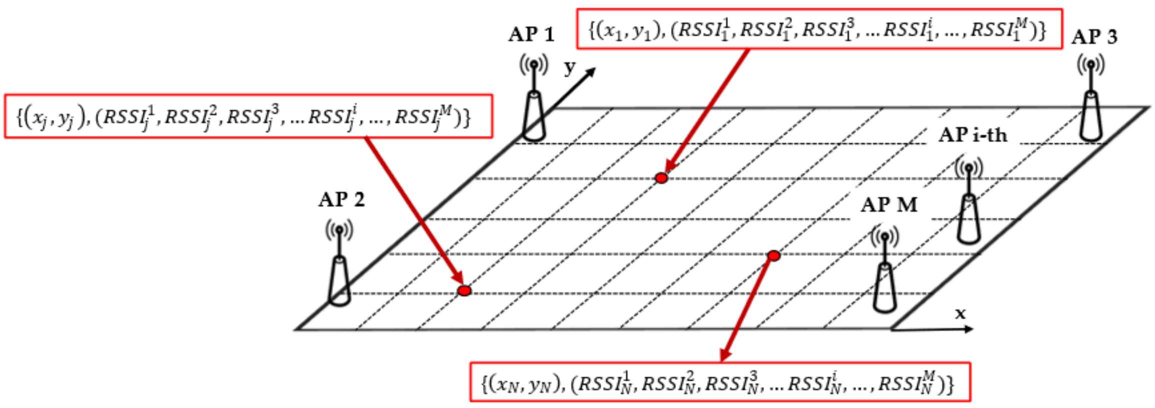
AP 1
AP 2
AP 3
AP i-th
AP M
y
x
$\{(x_1, y_1), (RSSI_1^1, RSSI_1^2, RSSI_1^3, \dots RSSI_1^i, \dots, RSSI_1^M)\}$
$\{(x_j, y_j), (RSSI_j^1, RSSI_j^2, RSSI_j^3, \dots RSSI_j^i, \dots, RSSI_j^M)\}$
$\{(x_N, y_N), (RSSI_N^1, RSSI_N^2, RSSI_N^3, \dots RSSI_N^i, \dots, RSSI_N^M)\}$

Figure 2. Anchor points collecting procedure

User dataset U includes RSSI measurements and motion data of users while they are moving. Users are required to have devices equipped with Wi-Fi module and IMU sensors in this case. For simplicity, an PDR algorithm is assumed to be used to extract heading angle and stride length from user motion data.

In Figure 3, a user moving along a track marked as blue. RSSI measurements points are marked as black on the trajectory. On the user trajectory, straight line between RSSI measurement points and angle form by 2 consecutive vectors were used to represent the result of PDR algorithm on motion data. For each RSSI measurement point in dataset U, the data is in the form:

$$U_k = \{(RSSI_k^1, RSSI_k^2, RSSI_k^3, \ ...RSSI_k^i, \ ..., RSSI_k^M), (\alpha_k, d_k)\}, k \in [1, N_U] \quad (2)$$

Where:
N_U is the number of data point in dataset U
M is the number of access point
$RSSI_k^i$ is the RSSI measurement from AP_i of point k^{th}
α_i is the angle formed by two vectors $\overrightarrow{U_k U_{k+1}}$ and $\overrightarrow{U_{k+1} U_{k+2}}$
d_i is the Euclidean distance between U_k and U_{k+1}

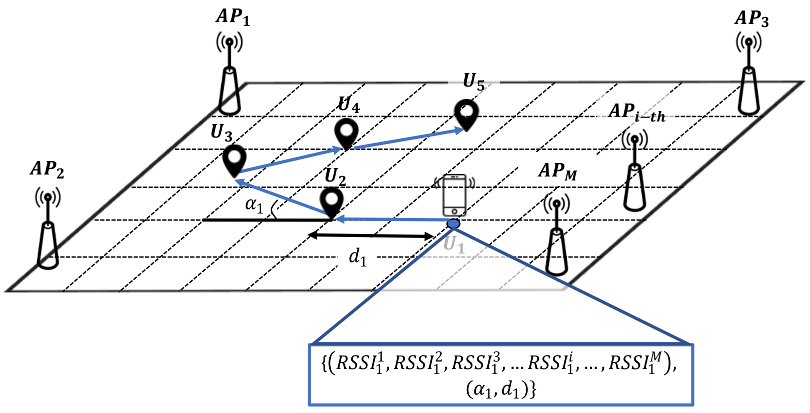

Figure 3. User data collecting procedure

In real cases, the proposed method does not have any knowledge of the user coordinate beforehand. But to perform validation of the method, the corresponding coordinate of RSSI measurement points in dataset U are collected. After that, the track are generated from the coordinate is denoted as,

$$T_{real} = \{(x_k, y_k)\} \ k \in [1, N_U] \quad (3)$$

3.3. Proposed evaluation metric

In dataset U, the angle and the distance between data points are known so the shape of user trajectory can be obtained. This is done by using PDR method, but since the initial location of the user is unknown, so the coordinate of the user track can not be obtained. Assuming that the coordinate of the user track is randomly generated and denoted as $T_{generated} = \{(\tilde{x}_k, \tilde{y}_k)\} \ k \in [1, N_U]$ with the condition that the shape of the generated track must remain the same as the result of PDR method. There will be multiple generated user track in different locations and the task is to select the track which is closest to the real track denoted as $T_{real} = \{(x_k, y_k)\}$. To do this, a new metric is proposed to measure how close $T_{generated}$ and T_{real}. First, the $T_{generated}$ is mapped to dataset U and used as training dataset for the machine learning model, then the dataset K is used as the testing dataset. The error is defined as the average Euclidean distance between the predicted coordinate and the coordinate in dataset K, it is calculated,

$$Err = \frac{\sum_{i=1}^{N_K} \sqrt[2]{(x_i - \tilde{x}_i)^2 + (y_i - \tilde{y}_i)^2}}{N_K} \ (m) \ i \in [1, N_K] \quad (4)$$

Where: $\tilde{x}_i$ and $\tilde{y}_i$ is the model predictions of x_i and y_i in dataset K.

Conventionally, calculated value from (4) show how well the trained model performs on testing dataset K. However, it is observed that this value is also reflects how good training dataset is. The idea is that if a machine learning model was trained using accurate training dataset, it would perform better than the one receives poor and inaccurate training data. Using this idea, many $T_{generated}$ can be created and the one with the lowest Err will be selected. However, it's worth mention that the selected track is not guaranteed to be the closest track to T_{real} because the evaluation is on dataset K. Therefore, the result depends largely on how good and accurate anchor dataset K is when they were collected. A simulation was carried out where anchor dataset K was collected and a predefined track T_{real} is simulated inside the experimental area, then 100

$T_{generated}$ tracks were randomly created with the same shape as T_{real}. Besides the proposed metric, the real average distance error (closeness), denoted as Err_{real} , between $T_{generated}$ and T_{real} is calculated using (5). The relationship between Err and Err_{real} is illustrated in Figure 4.

$$Err_{real} = \frac{\sum_{i=1}^{N_U} \sqrt[2]{(x_k-\tilde{x}_k)^2+(y_k-\tilde{y}_k)^2}}{N_U} \ (m) k \in [1, N_U] \quad (5)$$

Where:
$\tilde{x}_k$ and $\tilde{y}_k$ are the coordinate from $T_{generated}$
x_k and y_k are the coordinate from T_{real}

It is clear that there is a linear relationship as when value from of Err increases, the real error Err_{real} also grows. This proves that if the $T_{generated}$ is far from the T_{real}, then the Err would be high and when $T_{generated}$ is close to the real track, Err would be low. As mentioned ealier that the closest track may not be obtained and by looking at the lower left of Figure 4, it can be observed that the track with the smallest Err_{real} value has Err value a bit higher than it's neigbors. However, this is not a serious problem as the difference between the track with the lowest Err_{real} and the track with the lowest Err is not significant. If the track with lowest Err is found, it would always be neighbor with the track with the lowest Err_{real}. The next section proposes a solution to generate multiple $T_{generated}$.

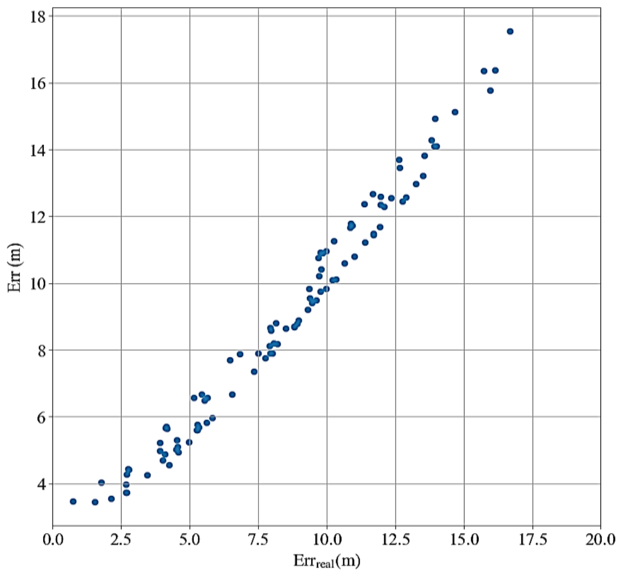

Figure 4. Relationship between Err_{real} and Err

3.4. Genetic algorithm

Genetic algorithm (GA) is a family of computational models that was first introduced by Holland in 1960 in [48] and it was applied to a wide range of optimized problems. Based on Darwinian evolution theory, the algorithm can be simplified into the following steps. At first, the initial population is generated, each individual in this population holds a unique chromosome. In other words, the population is the representation of the possible solutions, and every chromosome is the potential candidate for the optimized problem [49]. Next, to measure how good a chromosome is compared to others, a metric called fitness is used and calculated using the fitness function [50]. After fitness evaluation, individuals in the population are selected to generate the next generation. There are many available approaches for selection algorithms [51]. The simplest method is to select randomly but this method does not guarantee that the next generation would be better than the current one, so it's rarely used in practice. Commonly used selection method namely roulette wheel selection (RWS), stochastic universal sampling (SUS) [52], linear rank selection (LRS) [53]. Then, crossover operator is used to generate the next generation. To choose an appropriate crossover operator, it is important to take into consideration the problem type and the chromosome encoding method [54]. Then to maintain the diversity of the successors and prevent premature convergence, a mutation operator is applied. Because the risk of ruining an individual when performing mutation is high, the mutation rate is usually set to be small. In certain cases, the mutation rate would increase when population fitness has no improvement which is also known as adaptive mutation rate [55].

3.5. Proposed genetic algorithm

Based on the steps mentioned earlier, a genetic algorithm is implemented with the flow chart illustrated in Figure 5. At initialization, genetic algorithm parameters are set up: population size, number of generations, crossover rate, mutation rate, individual index and generation index, population container P, environment bounds, and motion data. The population initialization is emphasized in Figure 5 as the coordinate is generated randomly to build the track and the position must be within a predefined boundary to be appended to the population. Then, when the number of individuals in the population has met the requirement, the loop of selection, crossover, and mutate will occur until the convergence condition is satisfied or the number of generations exceeds. Details of the implementation will be described in the following sections.

Chromosome	$(\boldsymbol{x_1}, \boldsymbol{y_1})$	$(\boldsymbol{x_2}, \boldsymbol{y_2})$	…	$(\boldsymbol{x_k}, \boldsymbol{y_k})$	…	$(\boldsymbol{x_{N_U}}, \boldsymbol{y_{N_U}})$

Figure 5. Chromosome representation

3.5.1. Chromosome encoding

At first, a representation of an individual (chromosome) in the population is considered. An array with each element consisting of two floating-point numbers was chosen as the representation of coordinate to be the chromosome as in Figure 6. Using this form, genetic operators are easier to be implemented later.

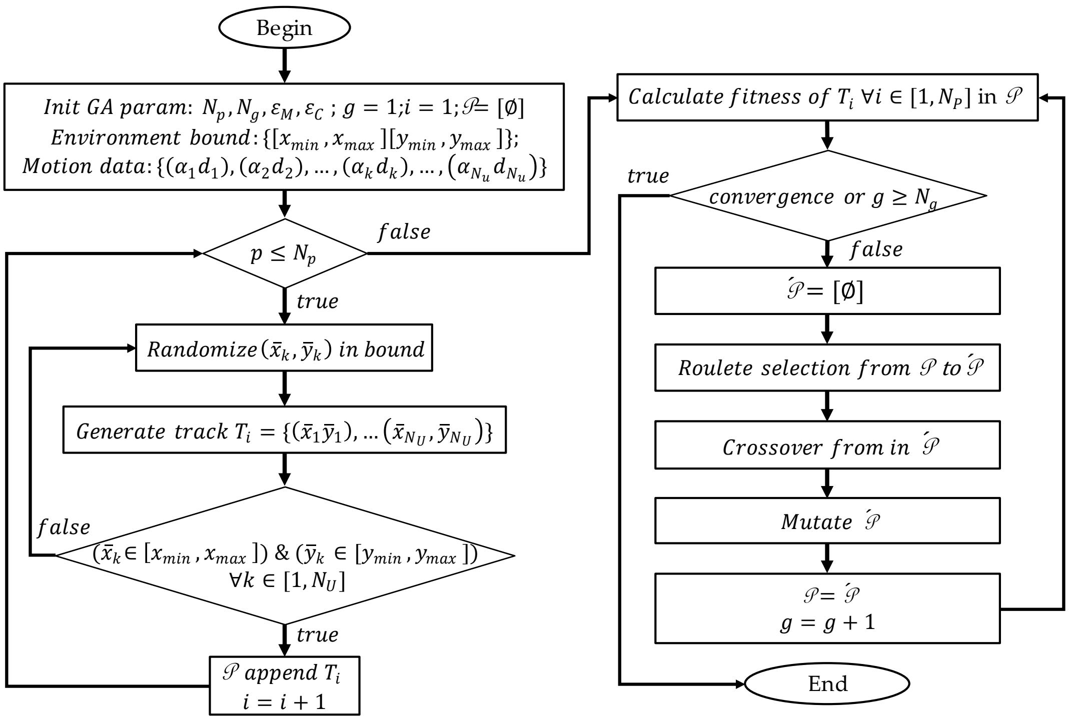
Begin
$Init\ GA\ param: N_p, N_g, \varepsilon_M, \varepsilon_C\ ; g = 1; i = 1; \mathscr{P} = [\emptyset]$
$Environment\ bound: \{[x_{min}, x_{max}][y_{min}, y_{max}]\};$
$Motion\ data: \{(\alpha_1 d_1), (\alpha_2 d_2), \dots, (\alpha_k d_k), \dots, (\alpha_{N_u} d_{N_u})\}$
$p \le N_p$
false
true
$Randomize(\bar{x}_k, \bar{y}_k)\ in\ bound$
$Generate\ track\ T_i = \{(\bar{x}_1 \bar{y}_1), \dots (\bar{x}_{N_U}, \bar{y}_{N_U})\}$
false
$(\bar{x}_k \in [x_{min}, x_{max}])\ \&\ (\bar{y}_k \in [y_{min}, y_{max}])\ \forall k \in [1, N_U]$
true
$\mathscr{P}\ append\ T_i$
$i = i + 1$
$Calculate\ fitness\ of\ T_i\ \forall i \in [1, N_P]\ in\ \mathscr{P}$
true
$convergence\ or\ g \ge N_g$
false
$\acute{\mathscr{P}} = [\emptyset]$
$Roulete\ selection\ from\ \mathscr{P}\ to\ \acute{\mathscr{P}}$
$Crossover\ from\ in\ \acute{\mathscr{P}}$
$Mutate\ \acute{\mathscr{P}}$
$\mathscr{P} = \acute{\mathscr{P}}$
$g = g + 1$
End

Figure 6. Proposed genetic algorithm flow chart

3.5.2. Population initialization

After chromosome encoding, the next step is to initialize the population. An important parameter needed to consider is population size [54]. Although there is no default number to use when selecting population size, major factors that need to be examined are the size of search space, how much processing capability is available, and initialization constraints. Large searching space would need more generated solutions to perform faster, but there would be trade-off for the consumption of processing power. After trials, a fixed population size of around 100 throughout the process is found to be optimal. In addition, to prevent the generated solutions to be too far off the environment, an area under constraints where value will be bounded by the maximum and minimum coordinate value in the x-axis and the y-axis. If the track is out of bounds, it'll be recreated. This ensures the solutions to be confined in the search space and the method would converge.

3.5.2. Fitness evaluation

Fitness value must indicate how well an individual performs compared to others. It can be estimated using the proposed metric in (4). However, it is better to keep the intuition that individuals with better performance will have higher fitness value, so the fitness value is defined in (6). A metric called fitness sum is also used to evaluate the performance of a generation and it is computed (7).

$$F(T_i) = \frac{1}{Err(T_i)} \; i \in [1, N_P] \tag{6}$$

Where:
T_i is the track of i^{th} individual.
$F(T_i)$ is the fitness value of i^{th}individual in the population.
$Err(T_i)$ is the error calculated using (4) of the track of i^{th} individual
N_P is the population size

$$FS = \sum_{j=1}^{N_p} F(T_j) \tag{7}$$

3.5.4. Selection

We chose to use Roulette Selection, which is one of the most popular selection techniques. It ensures that the individuals with high fitness value would have a better chance of being chosen. The probability of the individual to be selected can be calculated as,

$$p(T_i) = \frac{F(T_i)}{\sum_{j=1}^{N_p} F(T_j)} \; i \in [1, N_P] \tag{8}$$

Where: $p(T_i)$ is the probability of track of i^{th} individual being chosen?

3.5.5. Crossover

The crossover operator is formulated with an idea that an individual which has higher fitness would pass down more of its traits to the offspring. Therefore, a coefficient is defined to determine the amount of contribution of each individual of a parent. If chromosome A with fitness value $F(T_A)$ and chromosome B with fitness value $F(T_A)$ are selected as parents, then the coefficients of A and B are computed in (9). We denote the offspring chromosome as C and the coordinate of T_C is calcualted in (10)-(12).

$$coeff_A = \frac{F(T_A)}{F(T_A)+F(T_B)}; \; coeff_B = \frac{F(T_B)}{F(T_A)+F(T_B)} \tag{9}$$

$$x_k^C = coff_A * x_k^A + coff_B * x_k^B \tag{10}$$

$$y_k^C = coff_A * y_k^A + coff_B * y_k^B \tag{11}$$

$$T_C = \{(x_k^C, y_k^C)\} \; k \in [1, N_U] \tag{12}$$

Where:
x_k^C, y_k^C is the coordinate of k^{th} point in T_C
x_k^A, y_k^A is the coordinate of k^{th} point in T_A
x_k^B, y_k^B is the coordinate of k^{th} point in T_B

The result of crossover operator is a track that lies between parent tracks as illustrated in Figure 7. In Figure 7 it is assumed that the track B has a higher fitness value than the fitness of track A so the offspring C would be closer to track B as (10)-(12) are used to calculate the coordinate of track C. As a result, the offspring would retain the best traits from its parent and pass it down to next generation.

3.5.6. Mutation

It is better not to make big changes to the track position since the space where the experiment took place is not large, so mutation rate is selected to be a constant throughout generations. In addition, using constant instead of adaptive numbers would help the solution to converge faster and more stable. Compared to when the mutation rate increases, it would also increase the risk of ruining the individuals in the population.

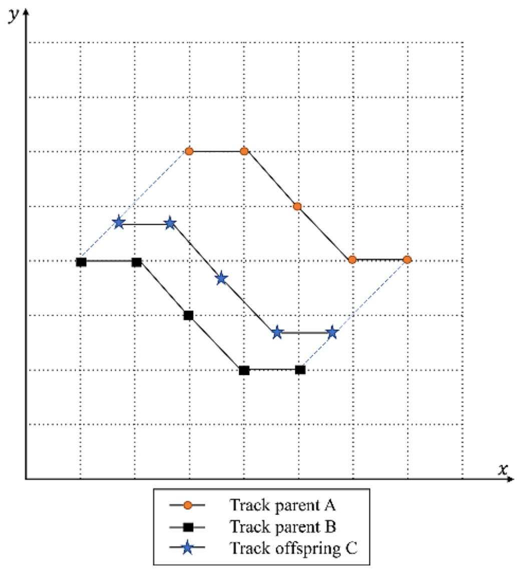

Figure 7. Illustration of result of crossover operator

4. RESULTS AND DISCUSSION

4.1. Experiment setups

The experiment is carried out in the laboratory which is a room 10 meters wide by 10 meters long. The environment is set up with tables, chairs, computers, and other networking devices to mimic real scenarios. The Wi-Fi network is set up and 8 APs are scattered around the room with unknown coordinates as the proposed method doesn't rely on the location of APs.

To collect dataset K, 8 points are desinated in the experiment area as anchor points and they are marked as red stars in Figure 8(a). Then, a device equipped with Wi-Fi transceiver module and software to record Wi-Fi signal strength were used. The device was carried to each of the designated anchor point to measure the RSSI and send the measurements to another computer to generate dataset K.

The user track is simulated by using real RSSI measurements and simulated motion data. Figure 8(b) shows the simulated user tracks with blue dots are RSSI measurements points. It can be seen from Figure 8(b) that the anchor points were omitted in the user track. In practice, users track RSSI measurement points and anchors can overlap. In this case, the performance of the proposed method was tested on tracks that contain no anchor points because when the machine learning model trains and validates on the non-overlapped dataset, the result of the model will be less biased. Earlier assumption that the PDR algorithm is used on motion data, the angle and the distance between RSSI measurement points in the user track are obtained. As the result, dataset U is collected.

Later in the genetic algorithm, multiple trajectory coordinates for the RSSI measurement points in dataset U would be generated. If the algorithm runs without the boundary constraints, many $T_{generated}$ would drift out of the experiment are. To optimize the proposed genetic algorithm, a constraint was set up to force the generated coordinate to be in the range of $[-5, 15]$ (meter) in both axes. The reason for the bound to extends to negative coordinates is that out of bound tracks may not be useless in certain cases. In the proposed genetic algorithm, the crossover operator could find a track that lies between parent's tracks. From this idea, the result of crossover between 2 tracks that far away and out of bound may still lies inside the area. As a result of the restriction, the population would be more diverse and potentially achieve faster convergence.

4.2. Machine learning model selection and configuration

Two popular regression models are selected which are Epsilon-support vector regression (SVR) and k-nearest neighbors regressor (KNN). They are used to evaluate the performance of our proposed method. The hyperparameter configuration for both models is shown in Table 1 and Table 2.

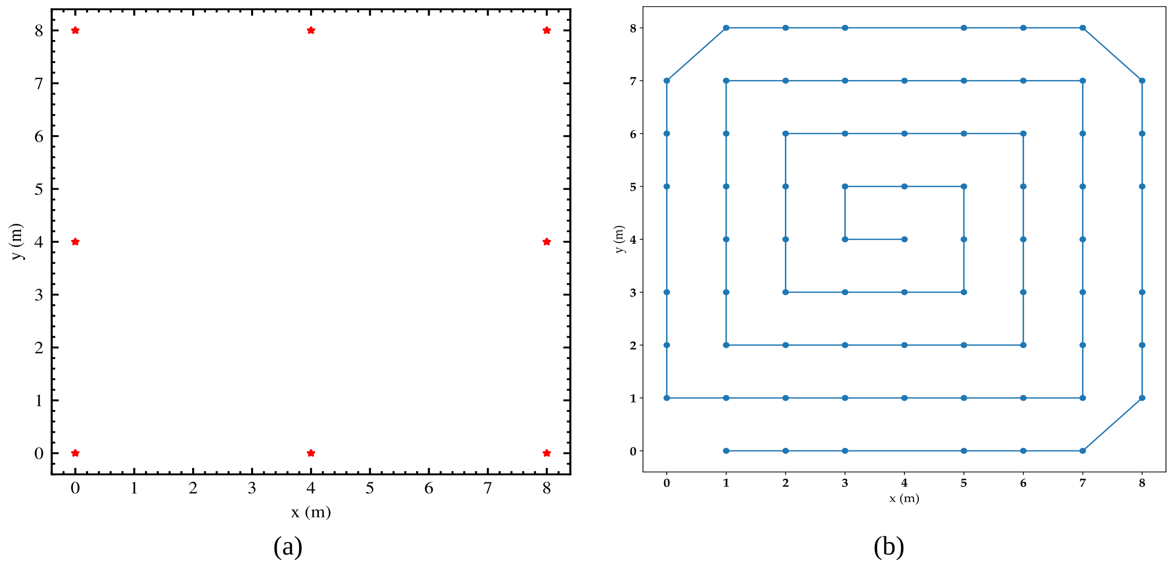

Figure 8. Experimental settings with, (a) designated anchor points and (b) User simulated trajectory

Table 1. SVR and hyperparameter configuration

Hyperparameter	Value
Kernel	Radical basis function kernel
Regularization parameter	1.0
Epsilon	0.1

Table 2. KNN and hyperparameter configuration

Hyperparameter	Value
N (number of neighbors)	5
Weight function	Uniform
Algorithm	Select best from Ball-Tree, kd tree and brute
Leaf size	30
Metric	Minkowski

4.3. Results without motion error

The performance of the proposed method was examined using simulated motion data without error. The genetic algorithm using 2 machine learning models with their configuration in the previous section and the results can be seen in Figure 9. Looking at Figures 9(a) and 9(b), it is apparent that KNN performs better than SVR with errors calculated using (5) are 0.9366 meters and 1.16865 meters respectively. The result of the proposed method is compared to the conventional fingerprinting method with machine learning where dataset K is used as the training dataset and model will predict the coordinate of RSSI measurement points in testing dataset U. The same machine learning models as well as the same configuration mentioned earlier for the method were used and the result can be seen in Figure 10.

While SVR is more flexible and could represent complex functions, the result in Figure 10(a) is chaotic and unusable. Because there are only 8 training datapoint in dataset K, this causes the KNN model to perform with multiple overlaps like in Figure 10(b). The error calculated using (5) of SVR and KNN, in this case, are 2.42199 and 2.64846 meters respectively. In general, it is expected that if the conventional fingerprinting method is used on a small training dataset, the trained models would suffer from underfitting and produce unwanted results.

Besides the accuracy and error of the results, the performance of the genetic algorithm was also measured. The number of generations it takes to converge and the fluctuation of fitness and error over each generation are considered. In addition to fitness sum in (7), the error sum is defined,

$$ES = \sum_{j=1}^{N_p} Err_{real}(T_j) \tag{13}$$

Where: $Err_{real}(T_j)$ is the real error calculated using $T_{generated}$ of j^{th} individual in the population and T_{real}

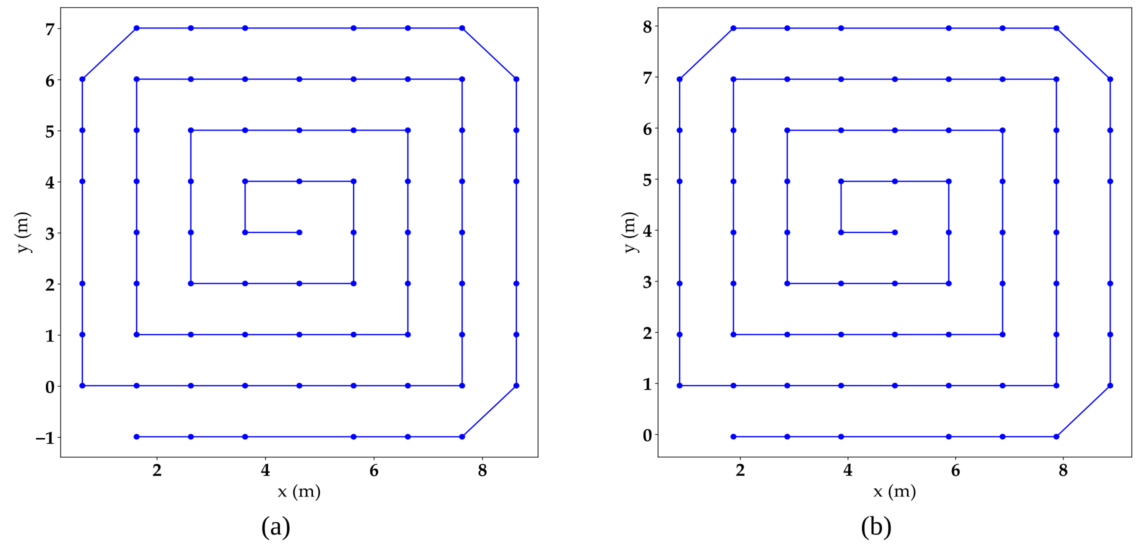

Figure 9. Results from the proposed method using different machine learning models, (a) SVR, (b) KNN

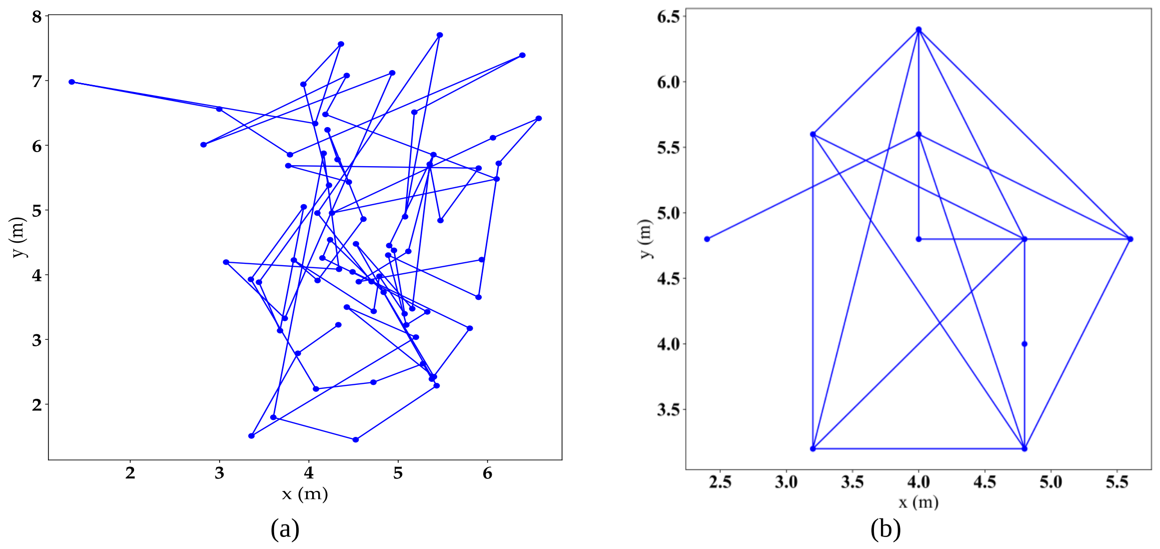

Figure 10. Results from using conventional fingerprinting method, (a) SVR, (b) KNN

Looking at the error sum in Figure 11(a), a similar case can be seen, but this time SVR had an increase of error from 27-ith generation. Some factors can contribute to the fluctuation such as the harmful mutation that happened on the good individuals or randomness in the initialization. As can be seen in Figure 11(b), the fitness sum increases rapidly at first couple generation and from generation 10^{th}, the solution converged and KNN achieve higher fitness sum than SVR.

The proposed method results depend on the accuracy of dataset K so the error in dataset K also have an impact. It may happen because of the mutation and the genetic algorithm doesn't know the real error of the $T_{generated}$ so there is a chance that it would select the track with a higher fitness value but having higher Err_{real}. However, as mentioned earlier that the algorithm ensures the selected track to be the closest track's neighbors, so the accuracy only drops slightly.

4.4. Results with motion error

In the previous section, it is assumed that the results of the PDR algorithm do not have errors. In this section, the proposed method performance is evaluated when IMU sensors are affected by noise, drift, and bias and produce errors in the results of the PDR algorithm. The error of the distances and angles in the user track is set to be in the range from 5 percent to 30 percent. The error was created by a random process that follows Gaussian distribution and then added to the distances and angles generated earlier. Using the proposed method with the same configuration of genetic algorithm and machine learning models as the previous section and the results when using the SVR model are illustrated in Figures 12(a) to 12(f) while the results of the KNN model are shown from Figure 13(a) to 13(f).

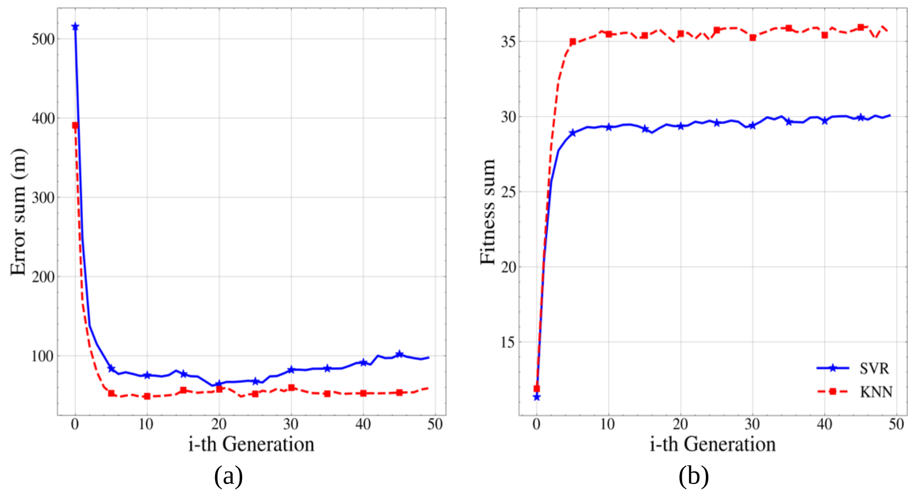

Figure 11. Plot of error sum and fitness sum with respect to generation ($N_p = 50$), (a) Error sum, (b) Fitness sum

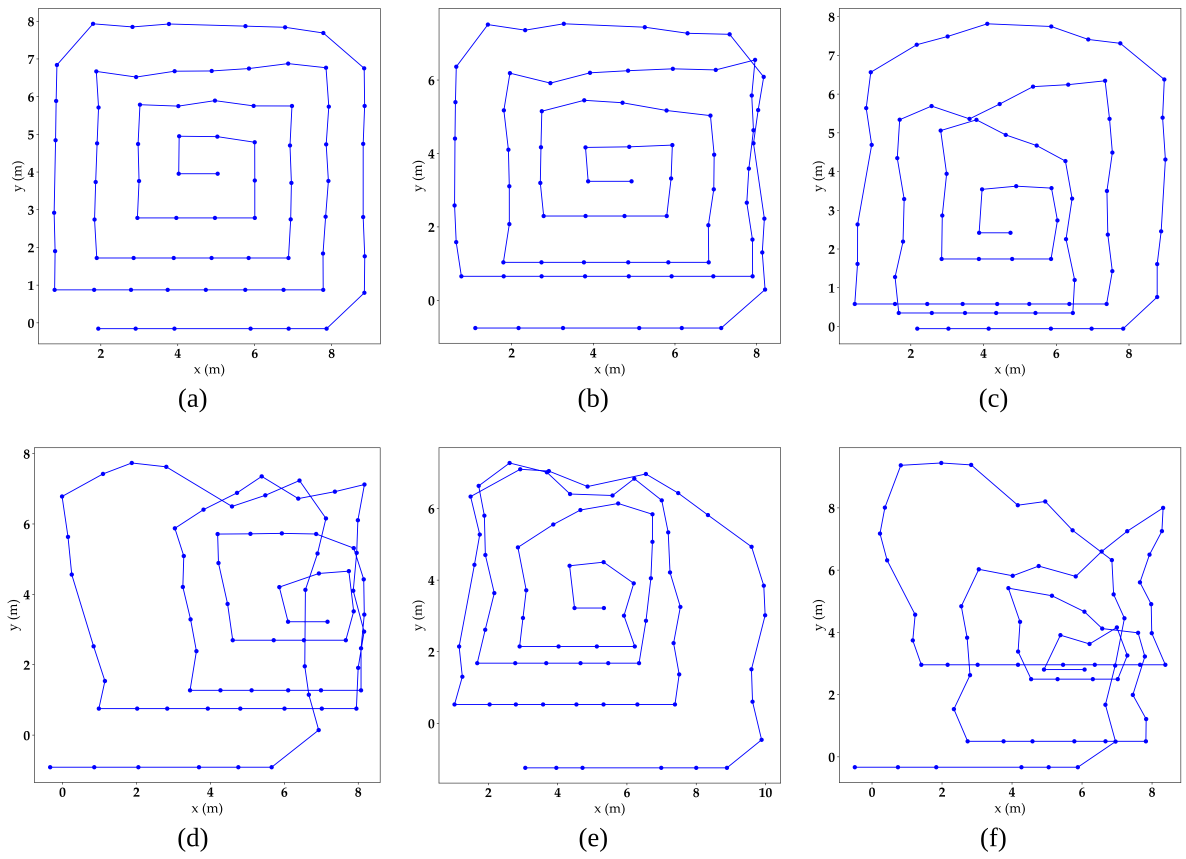

Figure 12. Results of the proposed method using SVR with different amount of motion error, (a) 5%, (b) 10%, (c) 15%, (d) 20%, (e) 25%, (f) 30%

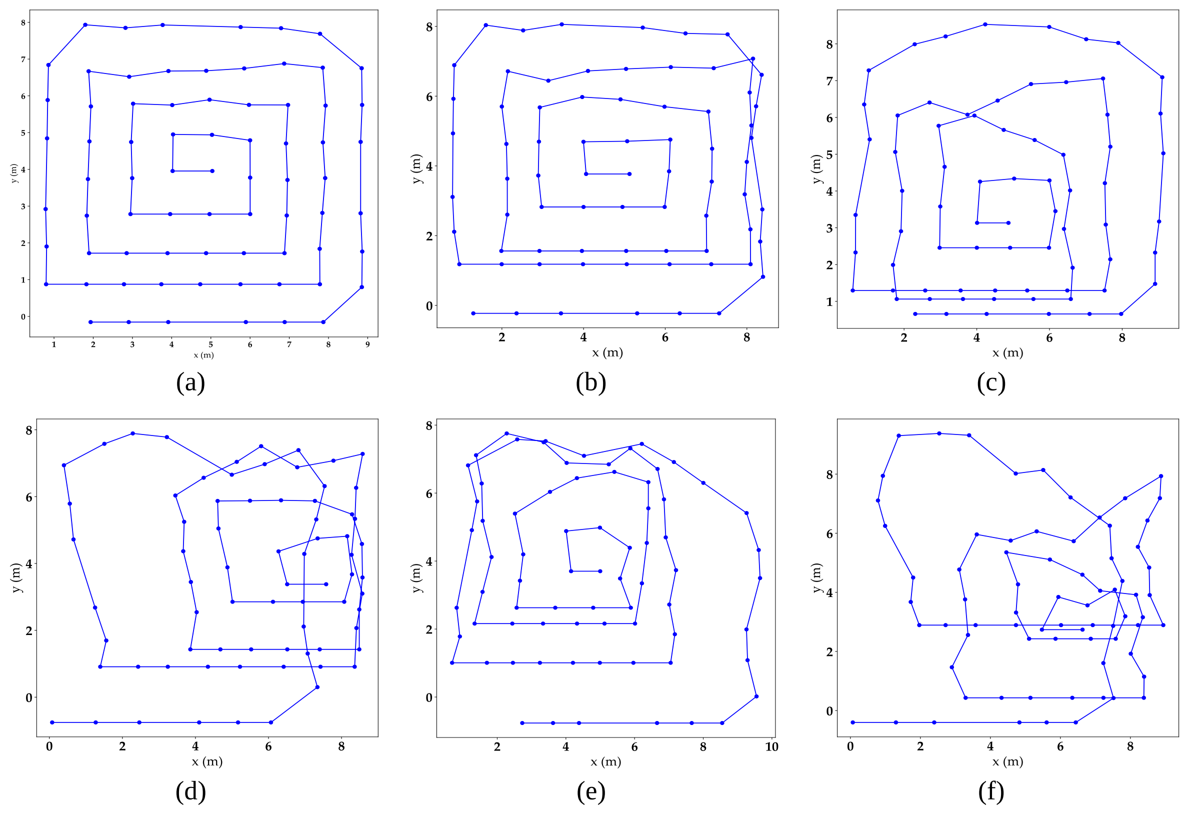

Figure 13. Results of the proposed method using KNN with different amount of motion error, (a) 5%, (b) 10%, (c) 15%, (d) 20%, (e) 25%, (f) 30%

As can be seen from Figure 12 and Figure 13, when the motion error increases, the deformation, and distortion of the user trajectory is also greater. If they are compared to the tracks using the conventional fingerprinting method, it is clear that there is less chaos in terms of user track shape and trajectory. In certain positions where motion error is not large, the shape of the user track can still be recognized. As the shape of the user track depends entirely on motion data and PDR algorithm, the only way to improve and counter the deformation is to use better sensors and apply a better PDR method.

To be consistent, once the error was added, the proposed method using SVR and KNN was used. That explains why the shape of the track for every level of error when using SVR and KNN is the same. Due to the existence of random processes in the simulation, 20 independent runs were performed with the same configuration and the average of average distance error from 20 runs is calculated. This value is refered in the graph and later as average error for convenience, results are shown in Figure 14.

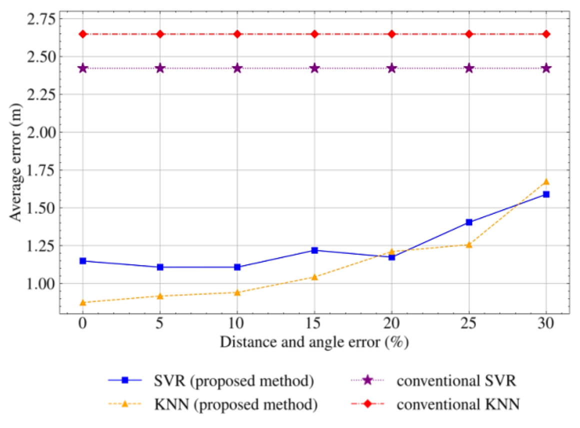

Figure 14. Plot of average error with respect to angle and distance error of the proposed method and conventional fingerprinting method

From Figure 14, it is clear that when angle and distance error increases, the average error also rises. There is an exception at 20% of angle and distance error that may be caused by big motion error generated by the random process or harmful mutation in genetic algorithm. In general, it's expected that the user track average error is affected by motion error. Details can be seen in Table 3, it can be observed that even when motion error is 30%, the proposed method was able to achieve average distance error at 1.589 meters and 1.674 meters using the SVR model and KNN model. Compared to the conventional fingerprinting method mentioned earlier, the proposed method's average error is nearly 34.4% and 36.7% less, which is a significant improvement.

Table 3. Detail average error of the proposed method with different amount of motion error

Motion error (%)	Proposed method average error (m)	
	SVR	KNN
0	1.145	0.875
5	1.108	0.917
10	1.108	0.941
15	1.219	1.043
20	1.174	1.211
25	1.404	1.257
30	1.589	1.674

The results are further analyzed by comparing the empirical cumulative distribution function (CDF) of the distance error of points in the user's track. The distance error of points is defined as the Euclidean distance from the one point in $T_{generated}$ and the corresponding point in T_{real}. Results can be seen in Figure 15(a) and Figure 15(b). In Figure 15(a), it can be seen that distance and angle error affect the distribution of distance error of points in the user's track. When motion error is low, for example at 5 percent motion error, most of the points in user track would have distance error under 1.0 meters. On the other hand, when motion error is large, the range also increases and even reaching over 4.0 meters of distance error at 30% motion error for both machine learning models.

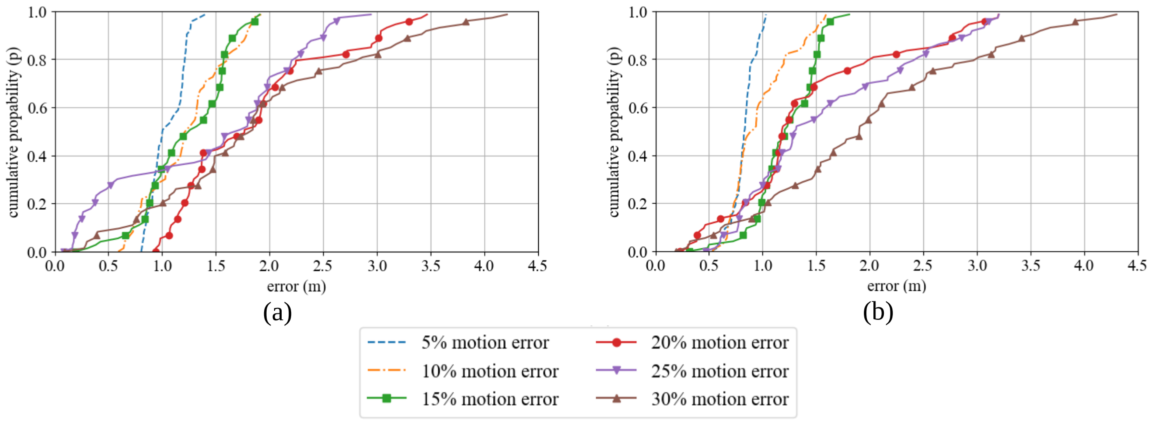

Figure 15. Plot of CDF of distance error of points in user track when applying proposed method using different machine learning models. (a) SVR, (b) KNN

To study the effect of angle and distance error separately, 20 independent runs were done using the same procedure and configuration as mentioned but with angle error and distance error separately. The result with only angle error is shown in Figure 16(a) and distance error only is shown in Figure 16(b). Looking at Figure 16(a), a similar pattern can be seen as in Figure 14. When the angle error ranges from 0 to 15 percent, KNN performs better and achieves a lower average error. But as the motion error gradually increases, the difference in performance for SVR and KNN is insignificant.

The average error when only distance error is added is lower than when the only angle is added as the results show in Figure 16(b). It is clear that the average error increases gradually when more and more distance error is put in thus the fluctuation between levels of distance error is very small. From the observation, angle error is the source of motion error which contributes more to the average error.

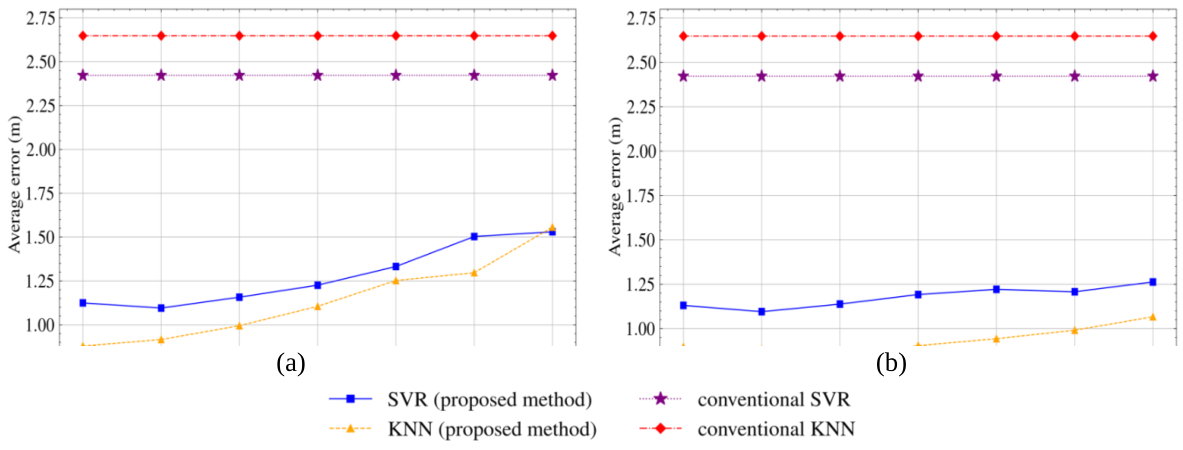

Figure 16. Plot of average error with respect to angle and distance error seperately of the proposed method and conventional fingerprinting method, (a) Angle error, (b) Distance error

4.5. Results with different number of anchor points

Because the proposed method generates user track coordinate by measuring how well the machine learning model performs on anchor dataset K, it's important to show how much performance is affected when the number of anchor points in dataset K changes. Using the same method to add distance and angle error and all the same configurations from the previous section, the number of anchor points is lowered from 8 to 6 and 4 and measured the difference in performance. The results of the proposed method using SVR are illustrated in Figure 17(a) and the one using KNN is illustrated in Figure 17(b).

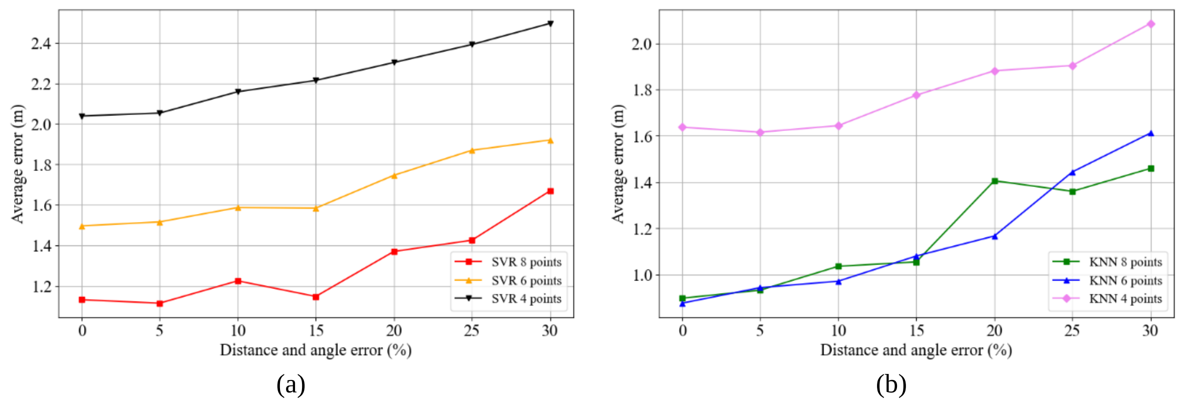

Figure 17. Plot of average error with respect to distance and angle error of the proposed method using different machine learning model and different number of anchor points. (a) SVR, (b) KNN

For both cases of SVR and KNN, it is obvious that the number of anchor points would greatly affect the overall performance of the proposed method. In the case of SVR, there is a big difference in terms of average error between using 8 anchor points and using 4 anchor points. As for KNN, the trend is similar to SVR, however, the difference between using 6 and 8 anchor points is insignificant. Detailed results are shown in Table 4.

From the results, even though the proposed method requires a small number of RSSI measurement points with known coordinates, it is ideal to collect a reasonable number of data points for the proposed method to perform well. Because when the number of data points in dataset K is set too low, there is not enough information for the machine learning models to perform a correct validation. Using more data points would certainly increase the performance of the proposed method but it would require more effort in collecting data. In real cases, the number of data points in dataset K can be adjusted considering several factors like how large the environment is, how dense the AP network and how the proposed method currently performs to satisfy the requirements.

Table 4. Detail average distance error of the proposed method with different number of anchor points

Model	Motion error (%)	Average distance error (m)	
		6 anchor points	4 anchor points
SVR	0	1.497	2.039
	5	1.516	2.053
	10	1.587	2.159
	15	1.584	2.215
	20	1.746	2.303
	25	1.870	2.392
	30	1.921	2.496
KNN	0	0.876	1.637
	5	0.943	1.616
	10	0.971	1.644
	15	1.080	1.776
	20	1.167	1.882
	25	1.444	1.904
	30	1.612	2.086

Comparison of CDF of distance error of points in user track obtained by our method across different number of anchors and motion error are presented in Figures 18(a) to 18(d). Results in Figure 18(a) and Figure 18(b) can be compared with the results in Figure 15(a) and Figure 15(b). When the number of anchor points decreases many distribution functions of distance error of RSSI measurement points also tend to shift to the right. In other words, they would have more points than containing a higher error. This is expected to happen because it has been discussed earlier how the performance of the proposed method is affected by the number of anchor points.

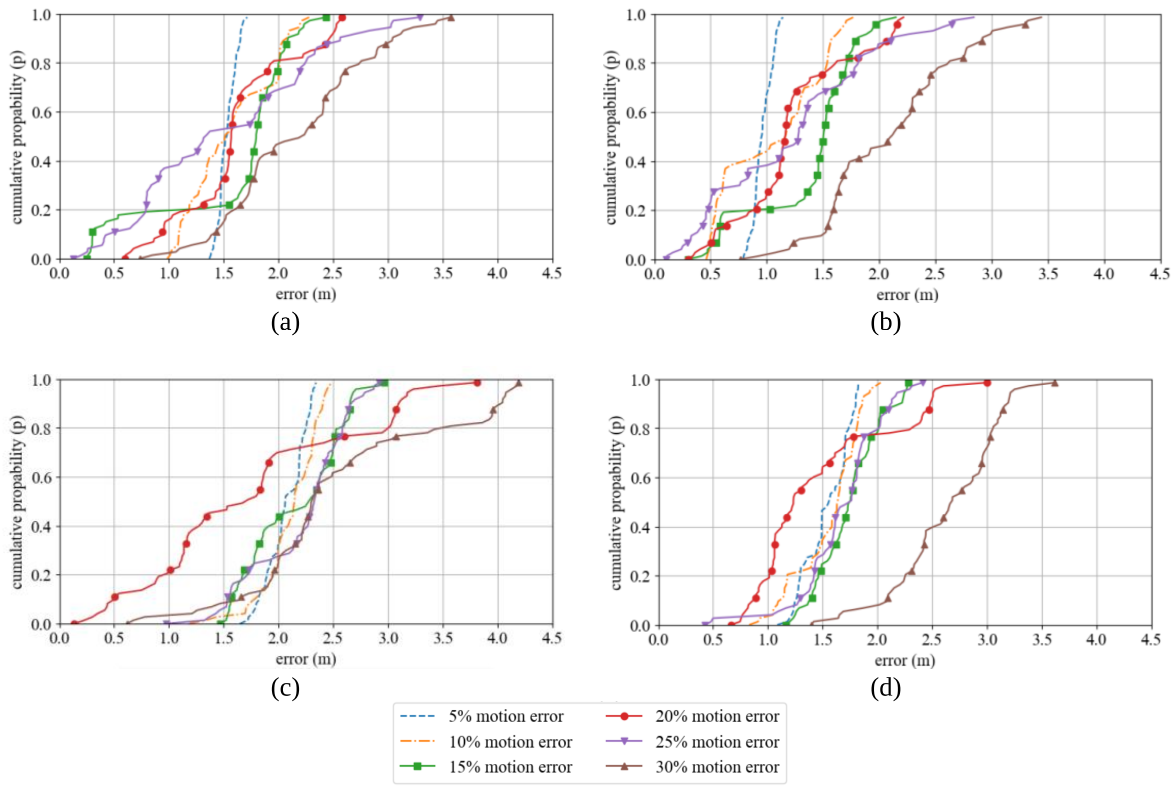

Figure 18. Plot of CDF of distance error of points in user track when applying proposed method using different machine learning models and number of anchor points, (a) SVR with 6 anchors, (b) KNN with 6 anchors, (c) SVR with 4 anchors, (d) KNN with 4 anchors

The same trend can be seen in Figure 18(c) and Figure 18(d) as the distribution covers a higher range, and the distance error is higher. Compared to fingerprinting approach in terms of data collection, only 8 data points are needed to make an accurate estimation. This makes the IP system that implements the proposed method to be flexible and could adjust, compensate for the error easily. If the environment has changes, only a few new data points need to be collected and then the accuracy can be adjusted by having more or fewer anchor points in the environment.

5. CONCLUSIONS

This paper introduces a new method to estimate indoor target trajectory location by combining Fingerprinting and PDR methods using a genetic algorithm. Experimentation and simulation results have shown that the proposed method can achieve high localization accuracy even with a limited number of training points with known coordinate and noisy motion instruments. From the indoor positioning perspective, the method would reduce a lot of effort in collecting and maintaining the dataset, which saves a lot of time and resources. In addition, no infrastructure needed to be installed as we take advantage of IMU sensors available in mobile devices and Wi-Fi networks. The main limitation of the proposed method is that using a genetic algorithm and machine learning models would increase the time complexity if parameters like the number of individuals in a population and the number of generations are set too high. By tuning parameters, the genetic algorithm can run faster because results have shown that GA converges fast. But still, mobile device hardware capability is limited, thus running such algorithms would be unsuitable. A better solution is to use a centralized server that is similar to other approaches like fingerprinting or interpolation and extrapolation so that more processing power can be harnessed and workload is reduced for client devices. In the future, further experimentation on real scenarios should be considered to validate the proposed method in different environments and configurations. In addition, studies on how to reduce the computational time would enable the proposed method to be used in real-time tracking applications. Besides traditional genetic algorithms, several algorithms in the field of evolutionary computation can be used for example, ant colony or particle swarm optimization. Thus, it would open many opportunities for research to apply the evolutionary algorithm to the field of indoor positioning.

REFERENCES

[1] S. Nirjon, J. Liu, G. DeJean, B. Priyantha, Y. Jin, and T. Hart, “COIN-GPS: Indoor localization from direct GPS receiving,” *MobiSys 2014 - Proceedings of the 12th Annual International Conference on Mobile Systems, Applications, and Services*, pp. 301–314, 2014, doi: 10.1145/2594368.2594378.

[2] H. Obeidat, W. Shuaieb, O. Obeidat, and R. Abd-Alhameed, “A Review of Indoor Localization Techniques and Wireless Technologies,” *Wireless Personal Communications*, vol. 119, no. 1, pp. 289–327, 2021, doi: 10.1007/s11277-021-08209-5.

[3] P. Pascacio, S. Casteleyn, J. Torres-Sospedra, E. S. Lohan, and J. Nurmi, “Collaborative indoor positioning systems: A systematic review,” *Sensors (Switzerland)*, vol. 21, no. 3, pp. 1–39, 2021, doi: 10.3390/s21031002.

[4] J. Kunhoth, A. G. Karkar, S. Al-Maadeed, and A. Al-Ali, “Indoor positioning and wayfinding systems: a survey,” *Human-centric Computing and Information Sciences*, vol. 10, no. 1, 2020, doi: 10.1186/s13673-020-00222-0.

[5] A. Anjum and M. U. Ilyas, “Activity recognition using smartphone sensors,” *2013 IEEE 10th Consumer Communications and Networking Conference, CCNC 2013*, pp. 914–919, 2013, doi: 10.1109/CCNC.2013.6488584.

[6] W. Kang and Y. Han, “SmartPDR: Smartphone-based pedestrian dead reckoning for indoor localization,” *IEEE Sensors Journal*, vol. 15, no. 5, pp. 2906–2916, 2015, doi: 10.1109/JSEN.2014.2382568.

[7] A. R. Jiménez, F. Seco, C. Prieto, and J. Guevara, “A comparison of pedestrian dead-reckoning algorithms using a low-cost MEMS IMU,” *WISP 2009 - 6th IEEE International Symposium on Intelligent Signal Processing - Proceedings*, pp. 37–42, 2009, doi: 10.1109/WISP.2009.5286542.

[8] A. Brajdic and R. Harle, “Walk detection and step counting on unconstrained smartphones,” *UbiComp 2013 - Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pp. 225–234, 2013, doi: 10.1145/2493432.2493449.

[9] Q. Wang, L. Ye, H. Luo, A. Men, F. Zhao, and Y. Huang, “Pedestrian stride-length estimation based on LSTM and denoising autoencoders,” *Sensors (Switzerland)*, vol. 19, no. 4, 2019, doi: 10.3390/s19040840.

[10] P. D. Tinh, B. H. Hoang, and N. D. Cuong, “Image-based gramian angular field processing for pedestrian stride-length estimation using convolutional neural network,” *IAES International Journal of Artificial Intelligence*, vol. 10, no. 4, pp. 997–1008, 2021, doi: 10.11591/IJAI.V10.I4.PP997-1008.

[11] M. J. Abadi, L. Luceri, M. Hassan, C. T. Chou, and M. Nicoli, “A collaborative approach to heading estimation for smartphone-based PDR indoor localisation,” *IPIN 2014 - 2014 International Conference on Indoor Positioning and Indoor Navigation*, pp. 554–563, 2014, doi: 10.1109/IPIN.2014.7275528.

[12] A. R. Jiménez Ruiz, F. Seco Granja, J. C. Prieto Honorato, and J. I. Guevara Rosas, “Accurate pedestrian indoor navigation by tightly coupling foot-mounted IMU and RFID measurements,” *IEEE Transactions on Instrumentation and Measurement*, vol. 61, no. 1, pp. 178–189, 2012, doi: 10.1109/TIM.2011.2159317.

[13] Y. Zhuang, Y. Li, L. Qi, H. Lan, J. Yang, and N. El-Sheimy, “A Two-Filter Integration of MEMS Sensors and WiFi Fingerprinting for Indoor Positioning,” *IEEE Sensors Journal*, vol. 16, no. 13, pp. 5125–5126, 2016, doi: 10.1109/JSEN.2016.2567224.

[14] Z. Chen, H. Zou, H. Jiang, Q. Zhu, Y. C. Soh, and L. Xie, “Fusion of WiFi, smartphone sensors and landmarks using the kalman filter for indoor localization,” *Sensors (Switzerland)*, vol. 15, no. 1, pp. 715–732, 2015, doi: 10.3390/s150100715.

[15] H. Nurminen, A. Ristimäki, S. Ali-Löytty, and R. Piché, “Particle filter and smoother for indoor localization,” *2013 International Conference on Indoor Positioning and Indoor Navigation, IPIN 2013*, 2013, doi: 10.1109/IPIN.2013.6817903.

[16] G. Hu, W. Zhang, H. Wan, and X. Li, “Improving the heading accuracy in indoor pedestrian navigation based on a decision tree and kalman filter,” *Sensors (Switzerland)*, vol. 20, no. 6, 2020, doi: 10.3390/s20061578.

[17] S. O. H. Madgwick, “An efficient orientation filter for inertial and inertial/magnetic sensor arrays,” 2010.
[18] P. D. Tinh and T. T. N. Mai, “Mobile Indoor Positioning System Utilizing WiFi RSSI and Motion Data,” *Proceedings - 2020 4th International Conference on Recent Advances in Signal Processing, Telecommunications and Computing, SigTelCom 2020*, pp. 98–102, 2020, doi: 10.1109/SigTelCom49868.2020.9199021.
[19] R. E. Kalman, “A new approach to linear filtering and prediction problems,” *Journal of Fluids Engineering, Transactions of the ASME*, vol. 82, no. 1, pp. 35–45, 1960, doi: 10.1115/1.3662552.
[20] S. J. Julier and J. K. Uhlmann, “New extension of the Kalman filter to nonlinear systems,” *Signal Processing, Sensor Fusion, and Target Recognition VI*, vol. 3068, p. 182, 1997, doi: 10.1117/12.280797.
[21] F. Gustafsson, “Particle filter theory and practice with positioning applications,” *IEEE Aerospace and Electronic Systems Magazine*, vol. 25, no. 7 PART 2, pp. 53–81, 2010, doi: 10.1109/MAES.2010.5546308.
[22] F. Subhan, H. Hasbullah, A. Rozyyev, and S. T. Bakhsh, “Indoor positioning in Bluetooth networks using fingerprinting and lateration approach,” *2011 International Conference on Information Science and Applications, ICISA 2011*, 2011, doi: 10.1109/ICISA.2011.5772436.
[23] D. Ahmetovic *et al.*, “Achieving practical and accurate indoor navigation for people with visual impairments,” *Proceedings of the 14th Web for All Conference, W4A 2017*, 2017, doi: 10.1145/3058555.3058560.
[24] C. Jihong, “Patient positioning system in hospital based on Zigbee,” *Proceedings - 2011 International Conference on Intelligent Computation and Bio-Medical Instrumentation, ICBMI 2011*, pp. 159–162, 2011, doi: 10.1109/ICBMI.2011.72.
[25] S. Gezici *et al.*, “Localization via ultra-wideband radios: A look at positioning aspects of future sensor networks,” *IEEE Signal Processing Magazine*, vol. 22, no. 4, pp. 70–84, 2005, doi: 10.1109/MSP.2005.1458289.
[26] H. Liu, H. Darabi, P. Banerjee, and J. Liu, “Survey of wireless indoor positioning techniques and systems,” *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, vol. 37, no. 6, pp. 1067–1080, 2007, doi: 10.1109/TSMCC.2007.905750.
[27] S. A. Golden and S. S. Bateman, “Sensor measurements for Wi-Fi location with emphasis on time-of-arrival ranging,” *IEEE Transactions on Mobile Computing*, vol. 6, no. 10, pp. 1185–1198, 2007, doi: 10.1109/TMC.2007.1002.
[28] C. Yang and H. R. Shao, “WiFi-based indoor positioning,” *IEEE Communications Magazine*, vol. 53, no. 3, pp. 150–157, 2015, doi: 10.1109/MCOM.2015.7060497.
[29] F. Wen and C. Liang, “An indoor AOA estimation algorithm for IEEE 802.11ac Wi-Fi signal using single access point,” *IEEE Communications Letters*, vol. 18, no. 12, pp. 2197–2200, 2014, doi: 10.1109/LCOMM.2014.2364852.
[30] K. Lin, W. Wang, Y. Bi, M. Qiu, and M. M. Hassan, “Human localization based on inertial sensors and fingerprints in the Industrial Internet of Things,” *Computer Networks*, vol. 101, pp. 113–126, 2016, doi: 10.1016/j.comnet.2015.11.012.
[31] M. N. Husen and S. Lee, “Indoor human localization with orientation using WiFi fingerprinting,” *Proceedings of the 8th International Conference on Ubiquitous Information Management and Communication, ICUIMC 2014*, 2014, doi: 10.1145/2557977.2557980.
[32] S. He and S. H. G. Chan, “Wi-Fi fingerprint-based indoor positioning: Recent advances and comparisons,” *IEEE Communications Surveys and Tutorials*, vol. 18, no. 1, pp. 466–490, 2016, doi: 10.1109/COMST.2015.2464084.
[33] F. Zafari, A. Gkelias, and K. K. Leung, “A Survey of Indoor Localization Systems and Technologies,” *IEEE Communications Surveys and Tutorials*, vol. 21, no. 3, pp. 2568–2599, 2019, doi: 10.1109/COMST.2019.2911558.
[34] P. D. Tinh and T. T. N. Mai, “Ensemble learning model for wifi indoor positioning systems,” *IAES International Journal of Artificial Intelligence*, vol. 10, no. 1, pp. 200–206, 2021, doi: 10.11591/ijai.v10.i1.pp200-206.
[35] A. Khalajmehrabadi, N. Gatsis, and D. Akopian, “Modern WLAN Fingerprinting Indoor Positioning Methods and Deployment Challenges,” *IEEE Communications Surveys and Tutorials*, vol. 19, no. 3, pp. 1974–2002, 2017, doi: 10.1109/COMST.2017.2671454.
[36] B. Jang and H. Kim, “Indoor positioning technologies without offline fingerprinting map: A survey,” *IEEE Communications Surveys and Tutorials*, vol. 21, no. 1, pp. 508–525, 2019, doi: 10.1109/COMST.2018.2867935.
[37] B. Ferris, D. Fox, and N. Lawrence, “WiFi-SLAM using Gaussian process latent variable models,” *IJCAI International Joint Conference on Artificial Intelligence*, pp. 2480–2485, 2007.
[38] J. Huang, D. Millman, M. Quigley, D. Stavens, S. Thrun, and A. Aggarwal, “Efficient, generalized indoor WiFi GraphSLAM,” *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 1038–1043, 2011, doi: 10.1109/ICRA.2011.5979643.
[39] L. Bruno and P. Robertson, “WiSLAM: Improving FootSLAM with WiFi,” *2011 International Conference on Indoor Positioning and Indoor Navigation, IPIN 2011*, 2011, doi: 10.1109/IPIN.2011.6071916.
[40] P. Robertson, M. Angermann, and B. Krach, “Simultaneous localization and mapping for pedestrians using only foot-mounted inertial sensors,” *ACM International Conference Proceeding Series*, pp. 93–96, 2009, doi: 10.1145/1620545.1620560.
[41] M. Angermann and P. Robertson, “FootSLAM: Pedestrian simultaneous localization and mapping without exteroceptive sensorshitchhiking on human perception and cognition,” *Proceedings of the IEEE*, vol. 100, no. SPL CONTENT, pp. 1840–1848, 2012, doi: 10.1109/JPROC.2012.2189785.
[42] K. P. Murphy, “Bayesian map learning in dynamic environments,” *Advances in Neural Information Processing Systems*, pp. 1015–1021, 2000.
[43] Y. Gwon and R. Jain, “Error characteristics and calibration-free techniques for wireless LAN-based location estimation,” *MobiWac’04 - Proceedings of the Second International Workshop on Mobility Management and Wireless Access Protocols*, pp. 2–9, 2004, doi: 10.1145/1023783.1023786.
[44] H. Lim, L. C. Kung, J. C. Hou, and H. Luo, “Zero-configuration indoor localization over IEEE 802.11 wireless infrastructure,” *Wireless Networks*, vol. 16, no. 2, pp. 405–420, 2010, doi: 10.1007/s11276-008-0140-3.
[45] K. P. Subbu, B. Gozick, and R. Dantu, “LocateMe: Magnetic-fields-based indoor localization using smartphones,” *ACM Transactions on Intelligent Systems and Technology*, vol. 4, no. 4, 2013, doi: 10.1145/2508037.2508054.
[46] M. Lee and D. Han, “QRLoc: User-involved calibration using quick response codes for Wi-Fi based indoor localization,” *Proceedings - 2012 7th International Conference on Computing and Convergence Technology (ICCIT, ICEI and ICACT), ICCCT 2012*, pp. 1460–1465, 2012.
[47] G. Shen, Z. Chen, P. Zhang, T. Moscibroda, and Y. Zhang, “Walkie-Markie: Indoor pathway mapping made easy,” *Proceedings of the 10th USENIX Symposium on Networked Systems Design and Implementation, NSDI 2013*, pp. 85–98, 2013.
[48] J. H. Holland, “An introductory analysis with applications to biology, control, and artificial intelligence,” *Adaptation in Natural and Artificial Systems. First Edition, The University of Michigan, USA*, 1975.
[49] T. Tometzki and S. Engell, “Systematic initialization techniques for hybrid evolutionary algorithms for solving two-stage stochastic mixed-integer programs,” *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 2, pp. 196–214, 2011, doi: 10.1109/TEVC.2010.2058121.

[50] A. L. Nelson, G. J. Barlow, and L. Doitsidis, "Fitness functions in evolutionary robotics: A survey and analysis," *Robotics and Autonomous Systems*, vol. 57, no. 4, pp. 345–370, 2009, doi: 10.1016/j.robot.2008.09.009.
[51] K. Jebari and M. Madiafi, "Selection Methods for Genetic Algorithms," *International Journal of Emerging Sciences*, vol. 3, no. 4, pp. 333–344, 2013.
[52] T. Blickle and L. Thiele, "A comparison of selection schemes used in evolutionary algorithms," *Evolutionary Computation*, vol. 4, no. 4, pp. 361–394, 1996, doi: 10.1162/evco.1996.4.4.361.
[53] D. E. Goldberg and K. Deb, "A Comparative Analysis of Selection Schemes Used in Genetic Algorithms," pp. 69–93, 1991, doi: 10.1016/b978-0-08-050684-5.50008-2.
[54] F. Herrera, M. Lozano, and A. M. Sánchez, "A taxonomy for the crossover operator for real-coded genetic algorithms: An experimental study," *International Journal of Intelligent Systems*, vol. 18, no. 3, pp. 309–338, 2003, doi: 10.1002/int.10091.
[55] D. Thierens, "Adaptive mutation rate control schemes in genetic algorithms," *Proceedings of the 2002 Congress on Evolutionary Computation, CEC 2002*, vol. 1, pp. 980–985, 2002, doi: 10.1109/CEC.2002.1007058.

BIOGRAPHIES OF AUTHORS

Pham Doan Tinh is a Lecturer at the Department of Electronics and Computer Engineering, School of Electronics and Telecommunications, Hanoi University of Science and Technology, Vietnam. In 2011, he earned Doctor of Engineering Degree from Ritsumeikan University, Japan. He also obtained Bachelor and Master Degree in Electronics and Computer Engineering from Hanoi University of Science and Technology (Vietnam) in 1996 and 2000, respectively. He can be contacted at email: tinh.phamdoan@hust.edu.vn

Bui Huy Hoang is a Student in Electronics and Computer Engineering Major of Hanoi University of Science and Technology (Vietnam). His research interests are in fields of Machine Learning Applications and Signal Processing. He can be contacted at email: hoang.bh172568@sis.hust.edu.vn.

Nguyen Duc Cuong is an AI Engineer at Big Data Analytics Center, Viettel Corporation. His research interests are in fields of Machine Learning Applications and Image Processing and Big-Data. He can be contacted at email: Cuongnd36@viettel.com.vn.