A Learning-Based Sequence-to-Sequence WiFi Fingerprinting Framework for Accurate Pedestrian Indoor Localization Using Unconstrained RSSI

Yingying Wang, Hu Cheng, and Max Q.-H. Meng, *Fellow, IEEE*

***Abstract*—Indoor location-based services are essential to daily life but lack a standard like outdoor Global Positioning System. Wireless fidelity (WiFi) received signal strength (RSS) is an optimal option thanks to its ubiquitous deployment, though existing research typically uses only a few constrained devices. We present a learning-based WiFi RSS indicator (RSSI) fingerprinting method designed for general environments. RSSI samples are collected in daily scenarios with hundreds of WiFi access points (APs) without disclosing their locations, and with randomly distributed reference points. The continuously measured RSSI values across multiple timestamps are treated as sequences, serving as fingerprints corresponding to location sequences. Instead of using all detected APs, we discard those sensed only in limited timestamps, i.e., APs with confined coverage and limited identification positions. We then employ various 1-D feature extractors to estimate the location sequence from the refined RSSI indices. Our method outperforms state-of-the-art methods on open-access datasets from a small office with densely equipped WiFi APs and a larger university campus with sparse WiFi signals. Real-world experiments on the CUHK campus further demonstrate the statistical consistency of the proposed method. We share the data collection code and self-collected data to facilitate future studies.**

***Index Terms*—Indoor localization, location awareness, ubiquitous computing, wireless fidelity (WiFi) fingerprinting.**

I. Introduction

ROBUST and accurate indoor localization has long been a dream in academia and industry. In addition to location-awareness demand, indoor location lays the foundation for many intelligence-of-things applications, such as navigation, smart retail, and virtual reality [1]. Though Global Positioning System (GPS) has produced excellent outdoor positioning results, its high-indoor application is prevented since it is easily blocked or attenuated by building structures [2], [3]. Anchor-based methods, such as ultrawideband (UWB) and Bluetooth, can offer enhanced performance. However, their effectiveness is highly dependent on the extensive deployment of dedicated infrastructure [4]. The widespread adoption of mobile devices equipped with various unobtrusive sensors presents new opportunities for ubiquitous localization. However, there is still no universal solution that operates effectively on standard devices without specialized hardware. Recent deep learning-based inertial odometry solutions can somehow limit the cumulative error of the inertial measurement unit (IMU) [5], [6], [7], [8], but its long-tracking performance suffers from significant measurement noise and bias.

The research interest in wireless fidelity (WiFi)-based localization is growing thanks to its wide deployment and ubiquity in everyday life. For example, the access points (APs) installed for public WiFi services in Hong Kong increased from 70428 in December 2020 to 85510 in March 2025 [9]. With the highly enhanced speed and extended coverage of the IEEE 802.11 standard, mobile devices can detect various APs at a single location. Channel state information (CSI), round trip time (RTT), and received signal strength (RSS) are three commonly employed WiFi characteristics. CSI provides detailed amplitude and phase differences among multipath components and is sensitive to pedestrian presence [10]. However, CSI is available only on specific chips and can only be extracted using specialized tools [11]. Moreover, CSI captures signal variations caused by dynamic movements between the transmitter and the receiver in the wireless environment [12], making it more suitable for small-scale perception applications, such as in-home localization, human activity classification, and even vital signs recognition. RTT assumes identical forward and reverse signal paths and leverages the RTT and signal velocity for location estimate [13]. However, its stability diminishes over time [14]. Since the IEEE 802.11mc standard was published in 2016, enabling the use of RTT, more mid-to-high-end smart devices have begun to support RTT capture. In contrast, RSS can be measured by most smart devices and is most suitable for ubiquitous localization [15]. Although RSS is sensitive to environmental changes, dynamically combining RSS measurements from multiple APs can enhance localization robustness [16].

A standard WiFi scan result in a normal smart device includes the human-made changeable names of the observed WiFi, the corresponding unique media access control (MAC) address represented by the basic service set identifier (BSSID), and their level, RSS indicator (RSSI) in dBm. Note that

Received 28 April 2025; revised 4 June 2025; accepted 24 June 2025. Date of publication 1 July 2025; date of current version 25 August 2025. This work was supported by the Hong Kong RGC GRF under Grant 14211420. *(Corresponding author: Hu Cheng.)*

Yingying Wang and Hu Cheng are with the Robotics, Perception and Artificial Intelligence Laboratory, Department of Electronic Engineering, The Chinese University of Hong Kong, Hong Kong, SAR, China (e-mail: yingyingw@link.cuhk.edu.hk; hcheng@link.cuhk.edu.hk).

Max Q.-H. Meng is with the Shenzhen Key Laboratory of Robotics Perception and Intelligence, Department of Electronic and Electrical Engineering, Southern University of Science and Technology Shenzhen, China (e-mail: max.meng@ieee.org).

Digital Object Identifier 10.1109/JIOT.2025.3583742

a single physical AP hardware can be equipped with a multiplexed configuration, which is known as virtual APs. To the wireless client, such as a smartphone, each virtual AP appears to be an independent AP with an individual MAC address represented by a unique BSSID [17]. For brevity, the AP in the remainder of this article refers to the virtual AP unless explicitly stated. In other words, we identify an AP by the unique BSSID without considering the physical AP. Two APs with different MAC addresses are treated as two independent ones, even if they correspond to the same physical AP. RSSI is a relative logarithmic indicator of the RSS, and its value is usually between -55 and -90 dBm provided by Android [18]. The closer the RSSI value is to 0, the stronger the received signal. An RSSI value less than -100 dBm typically indicates that the signal is extremely weak at the edge of what the smartphone can receive [19]. The main drawback of RSSI-based indoor positioning is multipath interference and nonline-of-sight (NLOS) propagation, which can be more serious in dynamic environments. Traditional methods typically employ an attenuation model to interpret signal strength, which assumes the RSSI signal of a specific AP can be represented by its relative distance to the AP [20], [21], and neglects the obstacles in between [22].

Fingerprinting-based WiFi localization is a more favorable and practical method that can fully utilize most of the measured RSSI information from multiple APs. This method involves two main phases, offline training and online testing. In the former stage, the RSSI vector samples and the corresponding preknown reference point (RP) form the fingerprint database. During the testing stage, the location is determined from the mapping function formulated in the training stage and the online measured RSSI vector. There are two main fingerprinting approaches. Deterministic fingerprinting algorithms typically rely on the K-nearest neighbor (KNN) algorithm and its variant [14], [23], where the estimated position is defined as the weighted mean of the top K matching locations. The accuracy of such methods is unstable not only due to the wide fluctuations of the RSSI values at a single location but also because it is highly dependent on the number of prior stored fingerprints. Including more fingerprints in the database can enhance the accuracy of the RSSI matching process but also lead to a higher computational burden. Probabilistic fingerprinting approaches employ statistical inference methods, such as Bayesian inference and maximum likelihood estimation, to determine user position from RSSI fingerprints [24]. A notable example of probabilistic techniques is the hidden Markov model (HMM)-based method [25]. HMM-based methods integrate the temporal correlation between successive RSSI measurements by modeling user locations as hidden states within a Markov process, which recursively updates the probability of each location based on the RSSI value observations at each time index. Both deterministic and probabilistic approaches rely heavily on carefully presegmented RPs and wireless signal propagation models [26].

Recent advances in machine learning have presented new possibilities to figure out the associations between location data and fluctuating RSSI samples [27], whose results rely heavily on the database quality [28]. Though crowdsourcing can overcome dataset size limitations, the groundtruth is usually not reliable enough, for example, extracting from inertial odometry [29], [30]. For existing WiFi fingerprinting systems, there are still some limitations. First, existing public fingerprinting datasets, such as UJIIndoorLoc [19], Tampere [31], and UTSIndoorLoc [32], typically feature sparsely distributed RPs. This sparse distribution can potentially affect the robustness of the trained deep learning-based fingerprint models, especially when adapting to continuous location changes in test data [33]. Second, most fingerprinting approaches are fine-tuned on datasets with number-limited and location-provided APs, typically no more than 20 [23], [27], [34], [35], [36]. This contrasts with normal buildings, where numerous APs exist without disclosed installation details. The discrepancy can result in the methodologies performing inconsistently in normal buildings, diverging significantly from performances reported in controlled settings.

In this article, we propose a data-driven WiFi RSSI fingerprinting method. Instead of focusing on modifying deep learning models and then validating them on a dataset with carefully designed RPs and enumerated APs, we concentrate on collecting the database in a more general and realistic setting. Specifically, this involves a pedestrian walking inside a typical building containing hundreds of APs, with each accessible location point being equally likely to be visited by the pedestrian. Our focus for the mapping model is on the preprocessing procedure to establish a more general fingerprinting framework. The fingerprint database collection is inspired by recent advances in the inertial odometry community, i.e., fixing the Tango phone on the chest for position information collection, whilst carrying an Android phone for saving WiFi RSSI readings. The WiFi samples are all from existing equipped APs without hardware or location modification. Neither the AP location nor the map of the testing environment is given before collecting the fingerprints. Our system can easily adapt to new environments without requiring dedicated infrastructure modifications by nonexperts and is capable of functioning effectively under dynamic conditions, such as the removal of certain APs. For example, fingerprints and mapping models can be easily collected and trained for a normal shopping mall. Even if some stores cease operations and remove their WiFi APs, the trained model continues to provide robust location estimation. The main contributions of this article are summarized as follows.

1) We employ randomly collected RSSI samples during normal walking instead of uniformly distributed RPs as location fingerprints and provide a 20-h self-collected dataset along with the data collection code.
2) We utilize RSSIs from hundreds of existing APs without disclosing their configurations as input features, which demonstrates competitive performance even after manually removing some RSSIs.
3) We propose mapping a fixed-length sequence of RSSI samples to a sequence of location information using various basic deep learning models.
4) The effectiveness of the proposed fingerprinting system is validated using both public datasets and the self-collected dataset.

II. Related Work

In this section, we introduce how machine learning has improved the WiFi RSSI-based fingerprinting performance from the perspective of model architecture design, preprocessing methods, and database construction. Battiti et al. first employed learning technology as the matching function between the RSSI vector and the location label [37]. They propose to utilize a three-layer multilayer perceptron (MLP) to find the association between RSSI values from three APs and the planar coordinate position. To improve the generalization ability, MLP with three hidden layers [38] and support vector regression (SVM) [39] have also been implemented as the matching function. MLP and SVM are more suitable for small-scale datasets, thus the systems are validated by a limited number of fingerprints. Benefiting from the development and high performance of deep learning technology, researchers started to explore the associations between a large amount of position and RSSI samples based on deep models. CNNLoc [32] employs the stacked autoencoder with 1-D convolutional neural network (CNN) for deterministic fingerprint generation. Recurrent neural network (RNN)-style model is trained for planar position estimation, requiring RSSIs from six different APs providing accurate initial location [27]. Path distance enabled deep metric learning [26] is designed to incorporate spatial information among various RPs, thus improving the localization accuracy of the KNN-based deterministic approach. Moreover, the generative adversarial network model has been applied to augment training samples from sparse collected samples [40].

Though adopting different deep learning architectures can enhance fingerprinting localization performance, some researchers find that carefully designed RSSI input features, i.e., preprocessing, can contribute greatly to localization accuracy. Reference [34] proposes focusing only on the RSSI samples when the device is detected as static. However, static periods are not so suitable for time durations when indoor localization is demanded, e.g., finding a specific store in a shopping mall. Sun et al. [41] combined the adjacent RSSI samples as a sequence, and integrate adjacent ones into new virtual sequences for data augmentation. However, the RSSI variations caused by mobile objects are neglected. Reference [42] demonstrates RSSI variance among adjacent RPs as input features can provide a higher localization accuracy than raw RSSI samples. The RSSI variance radio map for each MAC address is established, which is time-consuming and labor-intensive [43]. Li et al. [44] employed the RSSI differences among sparse labeled RPs as the features, which also leverages the model-based signal attenuation and neglects the signal fluctuations caused by the moving objects.

Database is a key component that highly affects the reported accuracy of machine learning methods, either for learning directly from raw samples or for augmentation-based approaches [45]. Database construction is labor-intensive and time-consuming, even for a small area [46]. Self-explored robots equipped with sensors are designed to reduce data collection manpower [47], which adds additional device-holding constraints to human-centered localization. Tang et al. claim that closed and private databases typically contain a limited number of APs and RPs [48], which may affect the realistic localization performance. Multiple public WiFi RSSI fingerprinting datasets exist to make a fairer comparison among different algorithms. UJIIndoorLoc [19] is one of the most popular public datasets [49] containing hundreds of APs, i.e., 520 APs over an area of 108 703 m^2, with groundtruth location determined by predefined RPs. However, the RP number of 933 is limited for such a large area. RPs are randomly collected without grid-based or preestablished mapping in the Tampere dataset [31]. However, there are only 4648 samples over 22 570 m^2 area, and the location of each RP is manually reported by the smartphone user. Hoang et al. [27] include only six different APs. SODIndoorLoc [15] reduces the distance between any two adjacent RPs to less than 1.2 m. The 105 APs among 8000 m^2 area, however, are carefully designed instead of the existing random ones. MTLoc [50] labels the first round of collected data, and then employs a crowdsourcing technique for database enhancement by comparing the new samples to the labeled ones according to the frequency of detected APs. It is assumed that the signal attenuation models for the APs detected in the first round of data collection remain unchanged.

In this article, we propose a novel, low-cost, and easy-to-implement WiFi RSSI fingerprinting benchmark. We focus on data preprocessing and database construction design. For the former part, inspired by previous studies [41], [42], we employ a sequence-to-sequence (seq2seq) model architecture to map RSSI samples to location points. Unlike methods that manually calculate RSSI variance, we adopt raw RSSI measurements, which contain the signal fluctuations caused by the moving signal collector, as the input features. Hundreds of selected sensible off-the-shelf APs are considered to form the input of the deep model, which is designed to capture the temporal and spatial RSSI variations in an end-to-end manner. For database construction, pedestrian random walking points are sampled and selected as RPs, without any constraints on their distributions. The groundtruth is obtained from a high-frequency visual inertial module, which is inspired by the inertial-odometry community. The proposed system is easily reproducible for nonexpert users. Evaluations on both public and self-collected datasets demonstrate the robustness and effectiveness of our method over time and the reduction of WiFi devices.

III. Proposed Method

The recent development of the IEEE 802.11ax (WiFi 6) standard and the upcoming WiFi 7 have increased the capability of a single AP with greater transmission distance [51]. This allows mobile devices at specific locations to detect APs located at greater distances, thereby increasing the number of detectable MAC addresses in environments with established AP infrastructures. In this work, we intend to explore the planar WiFi localization accuracy under the recent WiFi standard with the smartphone measuring the RSSI carried in a restriction-free manner, i.e., with randomly distributed RPs and typical daily phone carrying practices. Our system is mainly beneficial from two components, i.e., the preprocessing

Algorithm 1 Construction of the BSSID Feature Indexes

Input: $\{\boldsymbol{R}_1, \boldsymbol{R}_2, \cdots, \boldsymbol{R}_n\}$.

Output: Top k most frequently detected MAC addresses.

1: df$\longleftarrow$DataFrame($\{\boldsymbol{R}_1, \boldsymbol{R}_2, \cdots, \boldsymbol{R}_n\}$).
2: v_c$\longleftarrow$df.value_counts().
3: $l = len$(v_c).
4: $\boldsymbol{M}\longleftarrow$v_c[:$k \cdot l$]

module and deep learning-based location sequence prediction module. We will give a detailed description of these two modules in this section.

A. Data Preprocessing

In the fingerprints collecting procedure, the planar position $\boldsymbol{p}_t$ and the RSSI samples $\boldsymbol{R}_t$ at time t are recorded. $\boldsymbol{R}_t = \{r_{t1}, r_{t2}, \ldots, r_{td}\} \in \mathbb{R}^{d_t}$ contains RSSI data from d different MAC addresses, i.e., virtual APs. Continuously tracking a trajectory, we have $\{\boldsymbol{R}_1, \boldsymbol{R}_2, \ldots, \boldsymbol{R}_n\}$. Note that the length of $\boldsymbol{R}$ at different time indexes is not the same, because the device is within the connection range of different APs and thus can sense RSSI data from different MAC addresses at different points. If the smartphone remains quasi-static within the same region, it may detect an identical set of APs across different time instances. This scenario represents a special case that is inherently covered by our general assumptions. In each WiFi scanning, we have manually discarded the RSSI readings that are received after over 0.5 s than the receiving time of most of the other samples. These samples are usually caused by the NLOS propagation and can not reflect the real signal strength. Without the abuse of notation, we still represent the selected RSSI samples by the same notations as the raw vectors, i.e., $\{\boldsymbol{R}_1, \boldsymbol{R}_2, \ldots, \boldsymbol{R}_n\}$. Note that we assume that a sufficient number of LOS signals are available from adquently distributed APs. The assumption is reasonable because of the extensive and ubiquitous WiFi deployment in modern environments.

After saving all the prescreened RSSI samples and groundtruth positions, we first combine all the MAC addresses that appeared in the entire training set and transform it to a Pandas Dataframe [52] represented as df, which is then sorted by the corresponding detected number of times in descending order. Instead of using the RSSI data from all MAC addresses as the feature of the deep learning module, we propose to only employ k part of their values, i.e., $0 < k < 1$ of the total l BSSIDs. On the one hand, more RSSI samples at each sample time index form a more comprehensive location fingerprint. On the other hand, some RSSI samples are only sensed in very limited spaces and can be treated as disturbances [53]. Thus, we only choose a subset of sensed APs' corresponding RSSIs according to their sensed frequencies. The detailed choosing law of BSSID map indexes represented by BSSID feature array $\boldsymbol{M}$ is shown in Algorithm 1. Specifically, we adopt the value_counts() function and form the series v_c containing counts of each unique BSSID, which is in descending order of the occurring frequency, i.e., the first element of v_c is the most frequently detected MAC address with its detected counts.

Since only a subset of the total sensed APs is used to construct the input features, determining the appropriate subset proportion, i.e., the value of k, is crucial. To evaluate how localization performance varies with different k values, we iterate k from 0 to 1 in increments of 0.1. This iterative approach allows us to observe trends in localization performance across varying subset proportions, thereby identifying the optimal value of k for our model. When loading the RSSI data from a specific trajectory, we first find the RSSI values of the MAC addresses contained in the BSSID feature indexes array $\boldsymbol{M}$. For the MAC address presented in $\boldsymbol{M}$ but not detected at a specific time index, we set the RSSI value to -100 dBm, which indicates a nearly unreachable connection distance [18]. All RSSI values are ordered according to the MAC address sequence of $\boldsymbol{M}$. Note that our method aims to train the learning module to map the raw RSSI values to planar position, thus we do not manually adjust the sensed RSSI readings but only change the order of corresponding APs. Except for only parts of the MAC addresses providing input features, we make both the input RSSI features and the output coordinate positions as continuous sequences, i.e., from continuously measured RSSI data to estimate the pedestrian walking trajectory over a short period.

B. Deep Neural Network

The deep neural network (DNN) aims to find the associations between a sequence of preprocessed RSSI data and a sequence of position points. A key insight into adopting continuous RSSI data is that the sensory samples are temporally consistent in nature and contain long-range dependencies [54]. Adjacent position estimations are also correlated with each other. Adopting both input and output as sequences enables the model to better understand the correlations among different timestamps while maintaining position consistency across consecutive timestamps [55]. In addition to the benefits of using sequences for both input and output, the sequence-to-sequence formulation also captures certain trajectory patterns. For instance, sudden large changes in RSSI values are usually related to specific coordinate locations. Note that pedestrian states are not specifically considered. The signal processing pipeline remains consistent whether the pedestrian walks at varying speeds or stands still. The DNN module regresses the position in an end-to-end manner, which is expected to automatically capture location correlations in the position sequence regressed by each DNN forward process, whether the pedestrian is moving quickly with significant location differences or standing still with no movement. Instead of designing a complicated model architecture for comprehensive feature extraction, we directly apply the 1-D ResNet-18 as the feature extractor and focus on modifying the form of input and output to improve localization accuracy. As one of our main contributions, we propose to estimate the locations of multiple continuous timestamps from continuously collected WiFi RSSI readings, rather than the typical planar position at a single time index as the network output. A person can not go too far in a limited time duration, thus the positions of adjacent time slices nearly stay in a similar range.

The time slices represent the collections of RSSI samples across various time indexes. For simple representation, we use time slices with a number to indicate the included time indexes. Specifically, the measured RSSI data tracking a trajectory after the preprocessing process as described in Section III-A is represented by $\{\bar{\boldsymbol{R}}_1, \bar{\boldsymbol{R}}_2, \ldots, \bar{\boldsymbol{R}}_n\}$, where $\bar{\boldsymbol{R}}_t \in \mathbb{R}^{k \cdot l}$ has the same dimension. Typical fingerprinting approaches map $\bar{\boldsymbol{R}}_t$ to $\boldsymbol{p}_t$, the accuracy of which may be strongly affected by temporal disturbance [56], e.g., when another person walks past the line of sight between the AP and the device.

We propose to apply the 1-D ResNet-18 to find the associations between the RSSI values of i timestamps and the coordinate position at the corresponding time, i.e., from $[\bar{\boldsymbol{R}}_{t-i+1}, \bar{\boldsymbol{R}}_{t-i+2}, \ldots, \bar{\boldsymbol{R}}_t]$ to $[\boldsymbol{p}_{t-i+1}, \boldsymbol{p}_{t-i+2}, \ldots, \boldsymbol{p}_t]$. The value of i is also obtained through grid search. After the standard ResNet-18 backbone, we attach a basic 1-D CNN architecture, including a convolutional layer, batch normalization, and a ReLU activation. The parameters of the convolutional layer like kernel size, stride, and padding are modified to resize the length of the extracted features. The last step is a shared-weights MLP with two hidden layers transforming the feature of each channel to a 2-D output position. Specifically, for input RSSI values of length $k \cdot l$ across i timestamps, the initial input block is defined as

$$\boldsymbol{F}\left(\{\bar{\boldsymbol{R}}_{t-i+1:t}\}\right) = \text{MaxPool}(\max(\boldsymbol{0}, BN(f_{64}^{i}(\{\bar{\boldsymbol{R}}_{t-i+1:t}\})) \quad (1)$$

where f_{64}^{i} denotes a 1-D convolution with i input channels and 64 output channels. The kernel size, stride, and padding for the convolutional layer and the maxpooling layer are (7, 2, 3) and (3, 2, 1), respectively, resulting in a temporal downsampling factor of four, i.e., $\boldsymbol{F}(\{\bar{\boldsymbol{R}}_{t-i+1:t}\}) \in \mathbb{R}^{64 \times (\lceil k \cdot l \rceil / 4)}$. Subsequent to the input block, the network comprises four sequential residual groups, each consisting of two BasicBlock1D units, each of which incorporates two 1-D convolutional layers with kernel size 3, followed by batch normalization and ReLU activation. The first residual group uses a stride of 1, whereas the remaining three groups employ a stride of 2 to enable downsampling. The number of feature channels begins at 64 and doubles with each successive group, reaching 512 channels in the final group. Consistent with the standard ResNet architecture, a 1-D convolutional layer with a kernel size of 1 and the appropriate stride is applied to the residual connection to align the input and output dimensions. Following the residual groups, a specialized 1-D convolutional layer for each dataset transforms the resulting feature map from size $512 \times (kl/8)$ to $256 \times i$. Finally, a three-layer perceptron with units 256, 128, and 2, each intermediate layer activated by a ReLU function, is utilized to generate the final planar positions over i timestamps. Take the input RSSI feature numbers i as 4 for example, the detailed architecture of the neural network model is shown in Fig. 1. The loss function is designed as the mean-squared error (MSE) loss between the groundtruth positions and the estimated ones.

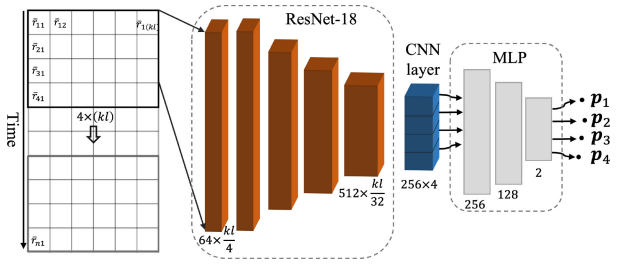


Fig. 1. Structure of the proposed WiFi fingerprinting model, where the input RSSI feature number is set to 4.

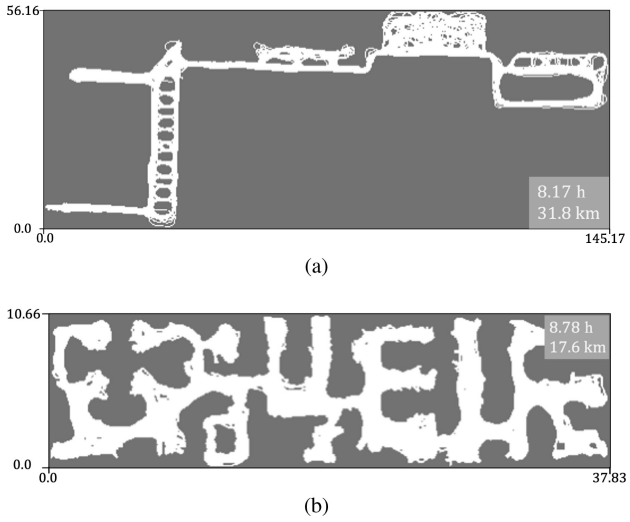


Fig. 2. Occupancy map of dataset named (a) University B and (b) Office C in NILoc.

IV. Experiments

A. Dataset

Our idea of continuously estimating the position is inspired by the inertial odometry community. We adopt a most relevant recent inertial odometry dataset, NILoc [7], to evaluate our method, where the groundtruth positions are from the visual-inertial module of a Tango phone and the sampling rate of WiFi signals is about 1/4 Hz, i.e., 4 s for saving an individual RSSI vector sample. The low-WiFi frequency is a result of the power protection of Android mechanisms. The inertial tracking results obtained from the benchmark method RoNIN [57] are also provided in NILoc. There are three different scenarios, including two university campuses and an office in NILoc. However, the dataset of a university does not contain WiFi readings but only IMU signals. We thus, employ the other two, i.e., named University B and Office C, as our evaluation datasets. The occupancy maps constructed from the training sets of these two datasets are provided in Fig. 2. We can observe that the trajectories contain complex curved paths. There are 554 different APs in University B and 170 in Office C. The lengths of the test sets are (3.71 h, 14.52 km) and (3.08 h, 6.16 km), respectively, which are sufficient enough to validate the generalization ability of the trained model. The University B and Office C datasets comprise 12 775 and 16 989 scans, respectively. In each scan, all RSSI samples are effective, as all readings are obtained within 0.5 s.

B. Implementation Details

To realize estimating locations of different time slices from different RSSI feature numbers, we set different parameters of the last convolutional layer in the DNN module to ensure that units of the first layer in the final MLP shown in Fig. 1 is 256. To determine the optimal proportion and time slices, we first evaluate localization performance by varying k from 0.1 to 0.9 in increments of 0.1 and adjusting the sequence length i from 1 to 6 using the University B dataset. After observing the basic fluctuations in localization performance across different k and i value pairs in the preliminary experiments, we further detail how to realize key k and i node pairs for both the University B and Office C datasets. PyTorch is used to implement the entire system. We apply Adam as the optimizer with the batch size 128 on a single GTX 2080Ti GPU and Intel Core i9-9900K @ 3.60 GHz CPU. During the training process, we apply dropout regularization with the keep probability of 0.5 in the last MLP module. All parameters are initialized by the default PyTorch random initialization. The learning rate is set to $1e^{-4}$ initially and then reduced by a factor of 10 if the validation location error does not decrease in ten epochs. The whole training process is terminated at the 150th epoch. Additionally, if the learning rate is reduced after two consecutive ten-epoch periods, indicating that the validation error becomes higher even with a lower learning rate, the training process is also terminated.

C. Comparison and Metrics

As far as the authors know, there are rare existing works that estimate positions from continuously collected RSSI readings within a single inference. The most similar work to this article is [27], where a synthetic trajectory is generated according to the relative distances among the groundtruth positions. There are five different RNN models in [27], where three of which require initial location states. However, an accurate initial position is not accessible easily in indoor scenarios. We thus, set the MIMO model in [27] as the comparing method, where a sequence of RSSI values from all the APs are the input of the two-layer LSTM model with 100 hidden units. A fully connected layer is after the LSTM layers for the output of the position sequence. Instead of generating a virtual trajectory, we directly apply the real collected RSSI vector sequence. Moreover, the MAC addresses corresponding to the input RSSI samples in this article are far more than the initial 11 in [27], we thus change the last step fully connected layer to a three-layer MLP like our method. The length of the input sequence and estimated position is set to 4, which is the best length setting with the lowest errors for [27] in our preliminary experiments. We also adopted another two comparing methods with the best sequence length and selected APs setting, i.e., random forest learning with 120 trees, maximum depth of 20, and random seed of 50 [58], and randomized neural network (RanNN) with 512 hidden nodes and ReLU activation [59].

We utilize two metrics to quantitatively evaluate the performance of our system, absolute trajectory error (ATE) and localization mean error (ME). ATE is calculated by the root mean square error between the groundtruth location and the estimated position. The ME is the mean squared location error over all location points. The cumulative distribution function (CDF) of the location error is also provided for better performance visualization.

TABLE I
LOCALIZATION ERRORS OF DIFFERENT k AND i VALUES

	i \ k	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
ATE	1	9.51	7.92	6.73	6.73	6.65	6.50	6.78	7.30	8.27
	2	7.13	6.62	5.92	5.45	5.59	5.28	5.38	5.63	5.67
	3	6.73	5.62	5.32	5.37	5.34	5.25	5.38	4.56	5.68
	4	6.62	5.46	5.28	5.22	5.27	5.15	5.39	5.44	5.68
	5	6.93	5.82	5.54	5.67	5.50	5.44	5.56	5.79	5.89
	6	7.83	6.23	5.95	5.73	5.96	5.66	6.32	6.38	6.51
ME	1	7.16	6.43	5.51	5.47	5.46	5.44	5.60	6.07	6.78
	2	5.68	5.34	4.78	4.54	4.68	4.34	4.44	4.62	4.69
	3	5.35	4.57	4.41	4.48	4.49	4.39	4.49	4.64	4.75
	4	5.33	4.44	4.39	4.28	4.41	4.24	4.45	4.47	4.65
	5	5.63	4.75	4.54	4.70	4.57	4.53	4.63	4.84	4.90
	6	4.86	5.10	4.83	4.70	4.91	4.68	5.25	5.19	5.38

V. RESULTS AND ANALYSIS

The preliminary results of varying the proportion parameter k from 0.1 to 0.9 and the number of time slices i from 1 to 6 on the University B dataset are presented in Table I. We observe that, for each k setting, the localization error decreases as i increases from 1 to 4 and begins to rise when i exceeds 4. Regarding the k values, a low proportion of 0.1 consistently results in the highest localization errors across all sequence settings. Conversely, the localization error decreases as more APs are incorporated into the input features, stabilizing with slight improvements for k values of 0.3, 0.4, and 0.5. A k value of 0.6 achieves the best performance, while higher proportions lead to reduced performance. Based on these observations, we identify k values of 0.3, 0.5, and 0.6, along with i values of 1, 2, 4, and 6, as critical parameters. Specifically, for k, the localization error decreases with increasing k, reaching its minimum at $k = 0.6$. In the final comparative analysis, we set the proportions to $(1/3)$, $(1/2)$, and $(2/3)$, and additionally included a k value of 1 to facilitate a comprehensive comparison that encompasses all APs in the input features. For the time slices, i values of 1, 2, and 4 are identified as key points showing localization performance improvements from the sequence learning problem formulation, whereas a sequence length of 6 indicates an overextension of the optimal sequence length. We further elaborate on the implementation of key k and i pairs in Table II.

The quantitative localization results of key k and i parameter settings, i.e., $k \in (1/3, 1/2, 2/3, 1)$ and $i \in (1, 2, 4, 6)$, are demonstrated in Table III, where the best performances among different settings are in bold. University B and Office C are two representative datasets, where the former occupies a large area and contains fewer evenly distributed APs, and the latter has a denser spread of WiFi transmitters ($554/(56.16 \times 145.17)$ versus $170/(10.66 \times 37.83)$). For University B with a sparse AP configuration, utilizing all APs, i.e., $k = 1$, consistently results in the highest or second-highest ATE and ME. For Office C, which covers a smaller area with a denser AP setting, the parameter of $k = 1$ yields better performance but does not

TABLE II
PARAMETER SETTINGS FOR DIFFERENT RSSI FEATURE PROPORTIONS AND TIME SLICES

Metric	*1/3*				*1/2*				*2/3*				*1*			
	1	2	4	6	1	2	4	6	1	2	4	6	1	2	4	6
University B Dataset:	*184*				*277*				*365*				*554*			
Kernel Size	5	5	7	7	7	7	7	7	7	7	7	7	7	7	7	7
Stride	1	1	1	1	2	2	2	2	4	4	2	2	4	6	4	3
Padding	0	0	2	3	0	0	2	4	0	0	1	3	0	0	1	2
Office C Dataset:	*57*				*85*				*113*				*170*			
Kernel Size	2	1	1	1	2	2	2	2	3	3	3	1	3	3	3	3
Stride	1	1	1	1	1	1	1	1	2	2	1	1	2	3	1	1
Padding	0	0	1	2	0	0	1	2	0	1	1	1	0	0	0	1

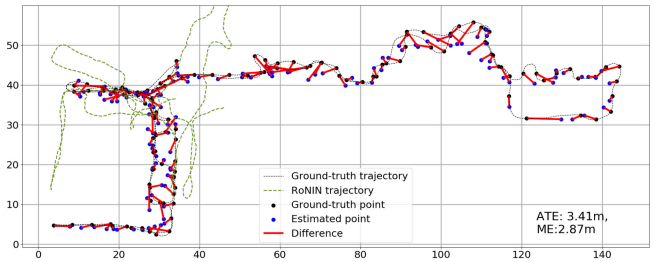
achieve optimal localization results. The dense AP setting is much more aligned with some studies that adopt limited APs which are sensible at each RP. We are surprised to find that the smallest ATE and ME of these two datasets are located on the same settings, i.e., the input RSSI vector number is 4 and the utilized proportion of the total APs is 2/3. The performance discrepancy between these two datasets is that when the input RSSI feature proportion is 1/3, the localization errors of the University *B* dataset stay quasi-consistent for different input time slices. For the Office *C* dataset with denser equipped WiFi APs, the localization error becomes more than doubled when the input sequence length is 4 or 6 compared with that of 1 and 2. The reason may be that for the Office *C* dataset, when the input feature number is 57, the size of the feature map after the ResNet backbone becomes 512×2. To transform it to the feature map with a size of 256×4 or 256×6, we need to set the padding value to be equal to or larger than the kernel size in the final CNN layer, which may introduce elements that are totally irrelevant to the previous input features. We can observe that for each proportion of input RSSI features, excluding the Office *C* dataset with input feature number 57, localization errors decrease as the sequence length increases from 1 to 4. However, the performance declines when the sequence length further increases to 6. The reduction in localization error demonstrates the effectiveness of sequence-to-sequence learning, while the subsequent performance decrease highlights the importance of appropriately setting the sequence length. Similarly, we can notice that when the input sequence length is fixed, the proportion of 2/3 almost always provides the smallest localization errors.

The qualitative results of applying 2/3 of the total MAC addresses to estimate four continuous positions are displayed in Fig. 3. We also demonstrate the trajectory of the RoNIN inertial tracking, which is provided in the NILoc dataset. We set the initial position of RoNIN as the groundtruth point. It can be observed that the orientation of RoNIN quickly deviates significantly from the real route. The proposed WiFi localization result is generally on the right track, despite positioning errors at several points. One limitation of WiFi localization is the sparse sampling points, which can cause unpredictable nonstraight paths among consecutive samples. Fusing inertial tracking with WiFi localization can provide a smooth trajectory estimation. We leave this as our future work and focus on purely WiFi localization in this article.

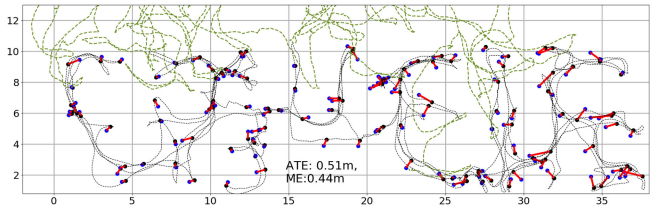
To quantitatively evaluate the proposed method, we demonstrate the CDF of location error of our method with

TABLE III
ATE AND ME EVALUATIONS (IN METERS) OF DIFFERENT RSSI FEATURE NUMBERS AND TIME SLICES

Metric	i \ k	*1/3*	*1/2*	*2/3*	*1*
University B Dataset:					
ATE	1	6.88	6.65	6.50	6.66
	2	5.83	5.59	5.34	6.35
	4	5.25	5.27	**5.07**	6.49
	6	5.76	5.96	6.07	6.68
ME	1	5.72	5.46	5.35	5.52
	2	4.75	4.68	4.46	5.32
	4	4.37	4.41	**4.15**	5.26
	6	4.76	4.91	5.03	5.52
Office C Dataset:					
ATE	1	3.11	2.80	2.77	2.75
	2	2.76	2.61	2.35	2.40
	4	7.63	2.38	**2.18**	2.25
	6	8.65	6.34	6.31	2.39
ME	1	2.71	2.42	2.39	2.38
	2	2.36	2.22	2.00	2.06
	4	5.63	2.04	**1.85**	1.94
	6	6.77	4.35	4.41	2.08



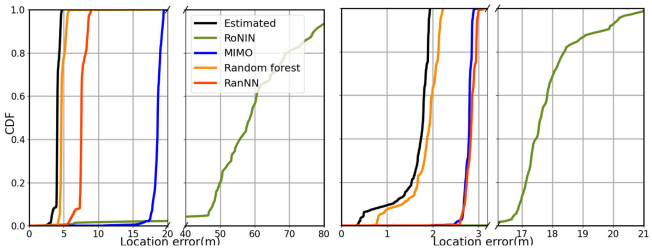
(a)



(b)

Fig. 3. Selected visualizations of point localization result from (a) University *B* dataset and (b) Office *C* dataset in Fig. 2. Part of the inertial tracking results of RoNIN is cropped to avoid room abuse.

the best parameter setting, MIMO, random forest, RanNN, and RoNIN inertial tracking in Fig. 4. The much poorer performance of RoNIN than other methods is mainly due to the serious direction deviation of the inertial tracking. The average location error of MIMO is much worse than our method on the University *B* dataset. However, it is only nearly $1m$ larger than



(a) (b)

Fig. 4. CDF of localization error for (a) University *B* dataset and (b) Office *C* dataset.

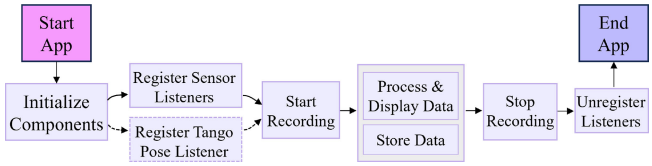


Fig. 5. Flowchart of data collection. Note that the Tango pose data is only collected using the ASUS Tango phone.

(a) (b)

(c) (d)

Fig. 6. Selected visualizations of collecting groundtruth position and WiFi readings in the real-scenario experiment. The ASUS Tango phone is fastened tightly on the pedestrian chest by an elastic strap. The WiFi RSSI collected device is Xiaomi 10S in (a) and (c), and Pixel 5 in (b) and (d), respectively.

the proposed method on the Office *C* dataset. The different performance of MIMO on these two datasets may be because MIMO is initially designed for WiFi fingerprinting with only 6 APs. It can provide relatively reliable localization results for small areas equipped with dense WiFi APs, but can not work accurately for a larger area with sparse WiFi transmitters. The smaller area of the Office *C* dataset and the more complex environment of the University *B* dataset also result in RanNN achieving similar performance to MIMO on the former dataset, yet approximately $10m$ lower localization errors on the latter dataset. RanNN demonstrates a stronger ability to avoid overfitting and extract deeper features compared with MIMO when the mapping from RSSI to location becomes more challenging, i.e., on the University *B* dataset. Among the four comparison methods, the random forest achieves the best performance, possibly benefiting from its insensitivity to outliers. In contrast, our method provides the best localization performance and produces statistically consistent positioning results on both the University *B* and Office *C* datasets.

VI. Self-Collected Data Evaluation

We also collect our own data in the CUHK campus to further validate the effectiveness of the proposed approach. We employ the visual-inertial fusion module of the Tango phone to track the groundtruth position of the pedestrian body.[1] The entire data collection software is based on the open-source code from the RIDI approach [60], which also serves as the data collection software platform for RoNIN [57]. Additionally, we have added WiFi RSSI data collection using the getScanResult method of the WiFiManager class within the Android protocol. The life cycle for the data collection application is illustrated in Fig. 5. Note that the only manual intervention for data collection is to control the start and stop of the data recording. Once recording begins, the pedestrian simply carries the phone and walks at random paces, typically below 1.5 m/s. Modern WiFi exhibits a median latency of approximately 3 ms, with 90% of latency values around 20 ms [61]. At the quasi-maximum walking speed of 1.5 m/s, the 90% latency corresponds to a displacement of approximately 30 mm, which is negligible for typical pedestrian movements. We have made the data collection code available for researchers to collect their own data.[2] From the hardware perspective, the ASUS ZenFone AR Tango phone is hung tightly on the chest of a pedestrian for collecting the groundtruth positions, while a Xiaomi 10S or a Google Pixel 5 is held freely for the acquisition of WiFi RSSI readings, as shown in Fig. 6. The Pixel 5 is on WiFi 5 while the Xiaomi 10S is on WiFi 6. WiFi 5 and WiFi 6 are the two most common standards for general commercial phones nowadays. The WiFi sampling rate of our collected data is the same as that of the NILoc dataset. Though the Android 10 or higher Android version allows toggling the WiFi scanning throttling (4 times every 2 s) off [62], we find that the highest frequency of the RSSI recordings still maintains a frequency of 1/4 Hz.

The data is collected on the 4th floor of the SHB building in the CUHK campus where long corridors and intricately decorated offices are contained. We name our dataset SHB4 dataset, where all groundtruth trajectories are aligned to the same coordinate frame by a presaved Tango Area Description File. The entire preprocessing, including data-labeling and input RSSI features transformation, is summarized in Algorithm 2. The temporal alignment between RSSI and position label is based on the closed neighbor method. Since the visual-inertial positioning outputs are at 200 Hz, the resulting time offset after synchronization is less than 2.5 ms. We share our dataset

[1]The groundtruth pose data is obtained from the TangoPoseData class in the Tango software development kit (SDK).

[2]Link for data collection code.

Algorithm 2 Preprocessing of SHB4 Dataset

for each file ∈ SHB4 **do**
  for each modality ∈ file **do**
    if modality == 'Tango' **then**
      $\{\boldsymbol{p}_\tau\} \leftarrow$ Load(file/pose.txt)
  for each modality ∈ file **do**
    $\{\boldsymbol{R}_t\} \leftarrow$ Load(modality/wifi.txt)
    $\left\{\overline{\boldsymbol{R}}_t\right\} \leftarrow \{\boldsymbol{R}_t\}$ indexed by RSSI feature array $\boldsymbol{M}$
    for $\boldsymbol{R}_t$ in $\left\{\overline{\boldsymbol{R}}_t\right\}$ **do**
      $t_p \leftarrow \min(t, \{\tau\})$
      $\boldsymbol{p}_t \leftarrow \{\boldsymbol{p}_\tau\}[t_p]$
    Save $\{\boldsymbol{p}_t\}, \left\{\overline{\boldsymbol{R}}_t\right\}$

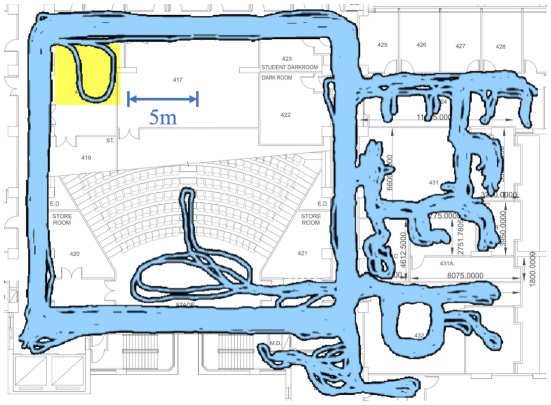


Fig. 7. Occupancy map of the SHB4 dataset. The floor map is aligned to the trajectories manually and is not utilized in our system.

to facilitate further research.[3] Detailed information about the SHB4 dataset is given in Table IV. To increase the scale of the feeding data, the WiFi readings from the Tango phone are also contained. There are nearly 20-h data in our SHB4 dataset, which are collected within a month, including different periods throughout different days. The occupancy map constructed from the SHB4 dataset is provided in Fig. 7. To evaluate the generalization of the trained model more comprehensively, the trajectories with a yellow background in the up-left corner in Fig. 7 are only contained in the test set. Compared with the dataset in [27] where RSSI readings are from 11 different MAC addresses and are collected at only 365 predefined RPs, the SHB4 dataset contains more realistic RPs that are randomly placed without interferences. Thus, the inferences between the RSSIs and positions can be more realistic. In fact, the authors are also surprised to find that there are a total of 899 BSSIDs from different MAC addresses in the whole of our dataset (833 for the training set, 13 and 53 additional MAC addresses for the validation set and test set, respectively). Specifically, for the training set, there are more than 400 BSSIDs sensed less than 100 times in the total of 16948 scans. One possible reason for the huge sensed BSSIDs is that there are at least 7 public WiFi services in the SHB building, i.e., CUHK1x, CUguest, Wi-Fi.HK via CUHK, eduroam, CSL, Y5ZONE, and CSL Wi-Fi Roam. Each public WiFi service has at least six routers on a single floor, with each router having its own MAC address for both the 2.4 GHz and 5 GHz frequency bands. Upon experiment evaluation, we found that we can at least sense the WiFi APs physically located on the 2nd, 3rd, 4th, 5th, and 6th floors on the 4th floor of SHB. There are 8 and 7 physical APs on the 2nd and 3rd floors, respectively. Thus, there are at least $7 \times 2 \times (8 + 7 + 6 \times 3) = 462$ BSSIDs from public WiFi services. There are also public WiFi services from other floors or buildings, and some personal WiFi services. Thus, the total BSSID number of 899 is reasonable. The huge BSSID number also indicates the ubiquitous WiFi signals and the potential for accurate and robust WiFi indoor localization.

[3]Link for SHB4 dataset.

TABLE IV
DETAILS OF THE SHB4 DATASET

	Scans	*RSSIs*	*BSSIDs*	*Length (km)*	*Times (h)*
Training	16948	746844	833	22.28	12.33
Validation	2741	118294	585	3.93	1.61
Test	7343	343214	747	10.42	5.79

To conduct a more thorough investigation of RSSI feature, we visualize RSSI values across different time periods, as shown in Fig. 8. The start data collection time is indicated in the title of each subfigure. The plotted dense RSSI values are obtained through linear interpolation of the raw sparse RSSI readings. The AP is the most frequently sensed one in the entire training set and is approximately located near the planar point (25, 15), as observed in the figures. Although its RSSI value at the same location varies across different sensing times, the three heatmaps exhibit similar overall color trends, with the same area showing high-RSSI values and similar regions almost unsensible. These consistent color patterns further demonstrate that, despite temporal variations in RSSI, it remains valuable for region localization. A combination of various APs is capable of providing detailed location information. By analyzing the heatmap of APs in the AP feature map $\boldsymbol{M}$, we found that more APs are sensed inside the laboratories than along the corridors, as indicated by the red colors in Fig. 8. Intuitively, a higher number of APs can provide better localization accuracy compared to fewer APs covering areas, due to multiple overlapping regions from different APs.

The first step of the preprocessing for RSSI features is the construction of the BSSID feature map using the top two-thirds of the most frequently sensed APs within the training set of the SHB4 dataset, as illustrated in Algorithm 1. A sequence of transformed RSSI samples, consisting of a length of 4 with each containing $\lfloor 833 \times (2/3) \rfloor = 555$ items, which represents RSSI values across four consecutive sample time indexes from 555 different APs, forms the input to the ResNet-style feature extractor. Using the optimal sequence length setting and the best proportion of APs for input feature construction, subsequent model training and evaluation are consistent with those of the University *B* and Office *C* datasets. Unlike the two public datasets, where all RSSI samples are effective, the SHB4 dataset contains 14 scans with a total of 231 RSSI readings abandoned due to receiving time intervals.

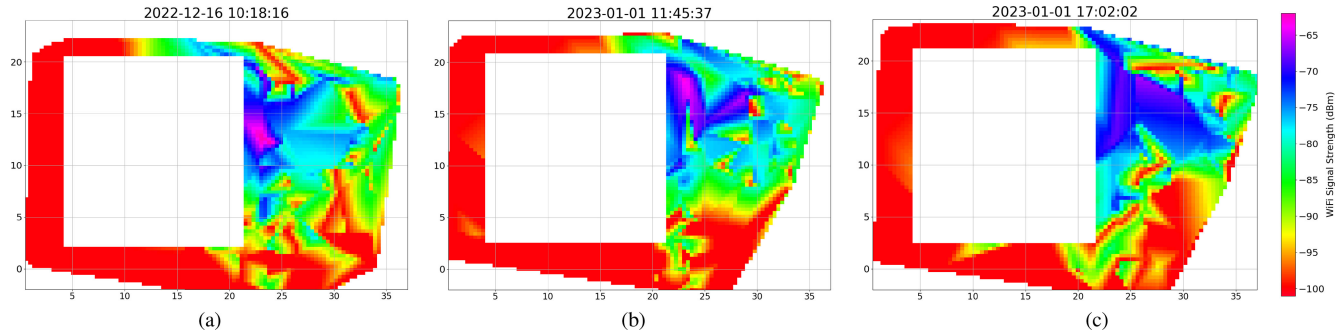


Fig. 8. Visualization of RSSI values. (a) and (b) are collected in the morning on different days, while (b) and (c) are gathered on the same day but at different times.

Specifically, there are seven scans containing 61 RSSI samples in the training set, 1 scan containing 18 RSSI samples in the validation set, and seven scans containing 152 RSSI samples in the test set. Including these features resulted in higher localization errors in our preliminary experiments. Our system leverages the basic learning framework but focuses more on the data reformulation module. Thus, there is no significant computational burden. Even for the SHB4 dataset with the longest input features, the network has 6.08 million parameters. The average runtime for a single forward pass with a batch size of 1 is 0.18 ms on our device, which meets real-time requirements. The parallel computation of the GPU and the velocity sequence generation during each network forward propagation can further improve the computational efficiency. For example, when the batch size is set to 32, the average inference time, including GPU loading, forward processing, and transferring back to the CPU, for each single position point over 500 positions is 0.058 ms. The single point inference time is comparable to the random forest operating at full CPU utilization, i.e., 0.054 ms.

Note that in both the public datasets and our self-collected SHB4 dataset, signals are collected in open, real-world areas where pedestrians may across. Although dynamic objects, such as moving pedestrians are not specifically tracked or counted during data collection, our localization framework based on hundreds of APs is capable of managing dynamic changes by utilizing temporal variations in RSSI measurements across different time periods as input. By learning from the fluctuating signal patterns in the same area caused by moving objects, the DNN effectively captures and interprets the inherent dynamics of the environment. Therefore, the mean absolute error of planar localization remains nearly consistent across different time instances, as shown in Fig. 9.

The CDF of location error is provided in Fig. 10. The proposed method has much higher accuracy than MIMO and RanNN and also slightly outperforms random forest. This observation is consistent with the localization results obtained on the University *B* and Office *C* datasets. We further conduct two sets of ablation experiments to demonstrate the robustness of our localization framework. The first set involved applying various model architectures for feature extraction. Specifically, we replace the ResNet-18 model with a two-layer bidirectional LSTM (BiLSTM) with 128 units per layer and the temporal convolutional network (TCN) [63], respectively.

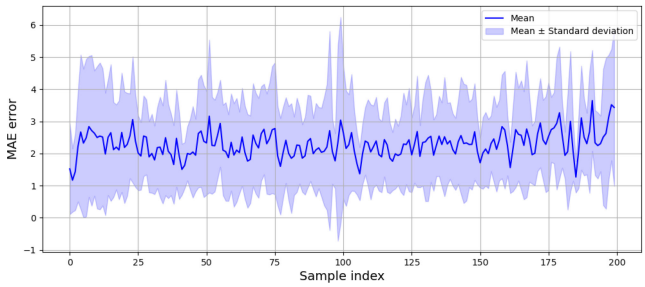


Fig. 9. Mean and standard deviation of absolute error across sample indexes.

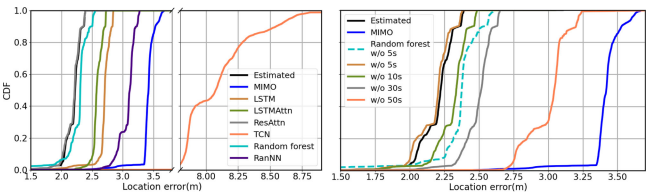


Fig. 10. CDF of location error for SHB4 dataset through (a) different mapping models and (b) manual reduction of APs.

The TCN comprises seven residual blocks, where the first four blocks have 64 channels each, and the subsequent three have 128, 64, and 32 channels, respectively. We further incorporate attention mechanisms into both the ResNet-18 and LSTM backbones. For the ResNet-18 module, we added a convolutional block attention module [64] to the input feature map (ResAttn), and for the LSTM module, we applied the dot-product attention [65] to the extracted features (LSTMAttn). Fig. 10(a) presents the CDF of localization error for these four DNN models. The TCN module exhibits the poorest performance among the three basic model architectures and the comparative MIMO method. Therefore, we do not incorporate an attention mechanism into the TCN model. The ResAttn model reduces localization error for approximately the first 2% to 10% of the data partition, but its performance was nearly equivalent to the initial ResNet-18 model when additional data were considered. In contrast, the dot-product attention significantly enhances the localization performance of the basic LSTM architecture, yet it remains inferior to ResNet-18. Both ResNet and LSTM architectures outperform the MIMO method substantially. Specifically, the TCN model

results in more than double the localization errors for each percentile compared to the MIMO method. Considering both accuracy and network simplicity, ResNet-18 demonstrated the best overall performance.

On the other hand, we also simulate the situations in which some APs are turned off and are not applicable during the online testing phase. Specifically, we manually set the RSSI values for some MAC addresses to -100 dBm. The selected MAC addresses are the first p values in the feature indexes array $\boldsymbol{M}$. We have shown the CDF of location error for $p \in \{5, 10, 30, 50\}$ in Fig. 10(b). We are delighted to find that when the 5 most frequently detected MAC addresses in the training set are not available in the test set (w/o 5s in Fig. 10), the location error is slightly smaller than the normal testing. The possible reason is that even the most frequently detected RSSI is not available everywhere. Thus, the model can better focus on the entire RSSI values instead of some specific ones. We also show the CDF of the location error of the random forest model when the five most frequently sensed APs are manually removed, i.e., random forest w/o 5s in Fig. 10(b). A noticeable performance degradation is observed. Though the random forest with comprehensive feature input performs comparably to our method, its location error increases significantly when the five most frequently sensed APs are removed, becoming even higher than that of our method when the ten most frequenctly sensed APs are excluded, as observed in almost all data partitions in Fig. 10(b). The relatively smaller increase in location error further highlights the greater robustness of our method. Even when 50 MAC addresses are turned off (w/o 50s in Fig. 10(b)), our model can still provide a better localization performance than MIMO, which indicates the robustness of our method. Our method is consistent across different pedestrian walking speeds, which can be observed from Fig. 11(b). Most of the localization errors for the test set are smaller than 5.0 m. Though the number of samples with pedestrian speed between 0.5 to 1.0 m/s is fewer than those with velocities lower than 0.5 m/s, i.e., 2478 versus 3130, the localization error outliers at higher velocities are as large as 20 m. A key observation is that pedestrians are likely to walk slowly inside the laboratories and move quickly along the corridors, where the former areas contain more AP settings and can provide more robust localization performance. The performance is consistent with intuition, i.e., more APs can provide better localization performance no matter what the pedestrian state is. In contrast, the largest localization error for MIMO occurs when the pedestrian is stationary, as indicated in Fig. 11(a). Except for the highest instant localization error, MIMO demonstrates lower robustness across all velocity distributions, with significantly more samples experiencing localization errors exceeding $5m$. On the one hand, our exhaustive AP-dependent localization framework generates more detailed features for the DNN module. On the other hand, the sequence-to-sequence learning pipeline leverages the correlation among continuous RSSI samples and adjacent walking points. The superior localization performance across all velocity distributions further underscores the higher accuracy and robustness of our system.

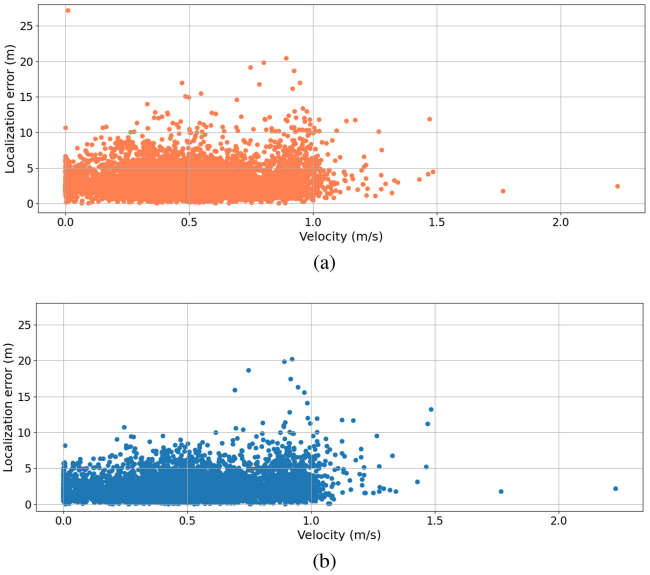


Fig. 11. Localization error over different walking speeds on the test set for (a) MIMO and (b) our method.

In addition to helping location-based services, the improved accuracy of indoor fingerprinting technology can open up new avenues for efficiency and creativity in indoor environments. For instance, by reducing the time spent searching for specific areas, accurate navigation within complex structures, such as shopping malls, hospitals, and large office buildings, can significantly enhance the user experience. To help ensure the safety of restricted zones, a more accurate localization system can also strengthen security procedures by tracking both permitted and unauthorized movements within sensitive regions. Furthermore, even with fewer APs, the localization system's increased resilience ensures consistent localization performance across different AP settings, minimizing infrastructure dependencies and reducing deployment costs. Overall, the accuracy improvements in our indoor fingerprinting system have meaningful implications for enhancing user experience, security, and operational effectiveness in indoor settings.

VII. Conclusion and Future Work

In this article, we propose a novel learning-based WiFi RSSI fingerprinting method for pedestrian indoor localization. We focus on localization in a realistic environment where hundreds of WiFi APs exist without their detailed location information. We introduce a data collection framework that enables unconstrained data collection and automatic labeling. Instead of estimating a single position point during each forward pass, our method predicts a position sequence in each propagation. During the training phase, RSSI values from different MAC addresses are sorted and selected to feed a ResNet-style network. Experiments on public datasets and a 20-h self-collected dataset demonstrate the effectiveness and robustness of the proposed method, even when the APs are manually removed to simulate a decrease in available APs. We also share the self-collected dataset along with the data collection code to facilitate future research. The collection of training

data and labeling are automated, and the proposed method is effective in environments where AP information is unknown. The entire system is more realistic and can be adapted to new environments, such as shopping centers without the involvement of experts. By forming the WiFi fingerprinting into a sequence-to-sequence deep learning framework, our research proposes a new method with higher localization performance. In addition, by sharing our self-collected 20-h dataset containing extensive WiFi RSSI information and inertial signals, we contribute a valuable resource that supports and promotes further research and collaboration in the field of indoor localization.

One of the limitations of this article is the sparsity of the sampling of WiFi readings, which may lead to unpredictability in estimating successive localization points. This limitation can degrade localization performance, especially in environments with fast-moving objects that move significantly between RSSI samples. In addition, the proposed system relies only on WiFi RSSI patterns, which may not be able to capture reliable location information in areas with sparse WiFi coverage or where signals are frequently blocked. In future work, we will test and adopt the proposed method in various indoor environments, including multifloor buildings, and study the impact of user-specific factors, such as walking speed and device handling. Moreover, we will investigate various signal processing techniques, such as filtering, to improve the quality of RSSI data fed into the neural network module. We will also incorporate IMU data to complement WiFi RSSI measurements for higher-frequency position estimation.

REFERENCES

[1] A. Zhang, Z. Min, Y. Wang, and M. Q.-H. Meng, "Towards an accurate augmented-reality-assisted orthopedic surgical robotic system using bidirectional generalized point set registration," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, 2023, pp. 4600–4607.

[2] T. Zhang et al., "RLoc: Towards robust indoor localization by quantifying uncertainty," *Proc. ACM Interactive, Mobile, Wearable Ubiquitous Technol.*, vol. 7, no. 4, pp. 1–28, 2024.

[3] P. S. Farahsari, A. Farahzadi, J. Rezazadeh, and A. Bagheri, "A survey on indoor positioning systems for IoT-based applications," *IEEE Internet Things J.*, vol. 9, no. 10, pp. 7680–7699, May 2022.

[4] G. Cerro, L. Ferrigno, M. Laracca, G. Miele, F. Milano, and V. Pingerna, "UWB-based indoor localization: How to optimally design the operating setup?" *IEEE Trans. Instrum. Meas.*, vol. 71, pp. 1–12, 2022.

[5] S. Bai, W. Wen, L.-T. Hsu, and P. Yang, "Factor graph optimization-based smartphone IMU-only indoor SLAM with multi-hypothesis turning behavior loop closures," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 60, no. 6, pp. 8380–8400, Dec. 2024.

[6] Y. Wang, H. Cheng, C. Wang, and M. Q.-H. Meng, "Pose-invariant inertial odometry for pedestrian localization," *IEEE Trans. Instrum. Meas.*, vol. 70, pp. 1–12, 2021.

[7] S. Herath, D. Caruso, C. Liu, Y. Chen, and Y. Furukawa, "Neural inertial Localization," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 6604–6613.

[8] Y. Wang, H. Cheng, and M. Q.-H. Meng, "Spatiotemporal co-attention hybrid neural network for pedestrian Localization based on 6D IMU," *IEEE Trans. Autom. Sci. Eng.*, vol. 20, no. 1, pp. 636–s648, Jan. 2023.

[9] "Public Wi-Fi services," 2025. [Online]. Available: https://www.ofca.gov.hk/en/ news_info/data_statistics/internet/wifi/index.html

[10] Y. Zhou, H. Huang, S. Yuan, H. Zou, L. Xie, and J. Yang, "MetaFi++: WiFi-enabled transformer-based human pose estimation for metaverse avatar simulation," *IEEE Internet Things J.*, vol. 10, no. 16, pp. 14128–14136, Aug. 2023.

[11] R. Du et al., "An overview on IEEE 802.11bf: WLAN sensing," *IEEE Commun. Surveys Tuts.*, vol. 27, no. 1, pp. 184–217, Feb. 2025.

[12] C. Chen, G. Zhou, and Y. Lin, "Cross-domain WiFi sensing with channel state information: A survey," *ACM Comput. Surveys*, vol. 55, no. 11, pp. 1–37, 2023.

[13] M. Mohsen, H. Rizk, H. Yamaguchi, and M. Youssef, "LocFree: WiFi RTT-based device-free indoor Localization system," in *Proc. 2nd ACM SIGSPATIAL Int. Workshop Spatial Big Data AI Ind. Appl.*, 2023, pp. 32–40.

[14] X. Feng, K. A. Nguyen, and Z. Luo, "A Wi-Fi RSS-RTT indoor positioning model based on dynamic model switching algorithm," *IEEE. J. Indoor Seamless Position. Navig.*, vol. 2, pp. 151–165, 2024.

[15] J. Bi, Y. Wang, B. Yu, H. Cao, T. Shi, and L. Huang, "Supplementary open dataset for WiFi indoor localization based on received signal strength," *Satell. Navig.*, vol. 3, no. 1, p. 25, 2022.

[16] J. Bi et al., "Inverse distance weight-assisted particle swarm optimized indoor localization," *Appl. Soft Comput.*, vol. 164, Oct. 2024, Art. no. 112032.

[17] "Overview of wireless virtual access point (VAP)," SonicWALL, 2024. [Online]. Available: https://www.sonicwall.com/support/knowledge-base/overview-of-wireless-virtual-access-point-vap/170505810128245 https://www.sonicwall.com/support/knowledge-base/overview-of-wireless-virtual-access-point-vap/170505810128245

[18] "Android.net.Wifi," 2024. [Online]. Available: https://developer.android.com/reference/android/net/wifi/WifiManager#calculateSignalLevel(int) https://developer.android.com/reference/android/net/wifi/WifiManager#calculateSignalLevel(int)

[19] J. Torres-Sospedra et al., "UJIIndoorLoc: A new multi-building and multi-floor database for WLAN fingerprint-based indoor localization problems," in *Proc. Int. Conf. Indoor Positioning Indoor Navigation (IPIN)*, 2014, pp. 261–270.

[20] J. Liu, H. Liu, Y. Chen, Y. Wang, and C. Wang, "Wireless sensing for human activity: A survey," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 3, pp. 1629–1645, 3rd Quart., 2019.

[21] R. Miyagusuku, A. Yamashita, and H. Asama, "Data information fusion from multiple access points for WiFi-based self-localization," *IEEE Robot. Autom. Lett.*, vol. 4, no. 2, pp. 269–276, Apr. 2019.

[22] P. Choudhary, N. Goel, and M. Saini, "A survey on seismic sensor based target detection, localization, identification, and activity recognition," *ACM Comput. Surveys*, vol. 55, no. 11, pp. 1–36, 2023.

[23] M. T. Hoang et al., "A soft range limited K-nearest neighbors algorithm for indoor localization enhancement," *IEEE Sensors J.*, vol. 18, no. 24, pp. 10208–10216, Dec. 2018.

[24] J. Wang et al., "Adversarial examples against WiFi fingerprint-based localization in the physical world," *IEEE Trans. Inf. Forensics Security*, vol. 19, pp. 8457–8471, 2024.

[25] Ó. Belmonte-Fernández, "Modeling the received signal strength intensity of Wi-Fi signal using hidden Markov models," *Expert Syst. Appl.*, vol. 174, 2021, Art. no. 114726.

[26] P. Chen and S. Zhang, "DeepMetricFi: Improving Wi-Fi fingerprinting Localization by deep metric learning," *IEEE Internet Things J.*, vol. 11, no. 4, pp. 6961–6971, Feb. 2024.

[27] M. T. Hoang et al., "Recurrent neural networks for accurate RSSI indoor localization," *IEEE Internet Things J.*, vol. 6, no. 6, pp. 10639–10651, Dec. 2019.

[28] Y. Tao and L. Zhao, "A novel system for WiFi radio map automatic adaptation and indoor positioning," *IEEE Trans. Veh. Technol.*, vol. 67, no. 11, pp. 10683–10692, Nov. 2018.

[29] X. Du, X. Liao, M. Liu, and Z. Gao, "CRCLoc: A Crowdsourcing-based radio map construction method for WiFi fingerprinting localization," *IEEE Internet Things J.*, vol. 9, no. 14, pp. 12364–12377, Jul. 2022.

[30] Z. Xu, B. Huang, B. Jia, and G. Mao, "Enhancing WiFi fingerprinting localization through a co-teaching approach using crowdsourced sequential RSS and IMU data," *IEEE Internet Things J.*, vol. 11, no. 2, pp. 3550–3562, Jan. 2024.

[31] E. S. Lohan, J. Torres-Sospedra, H. Leppäkoski, P. Richter, Z. Peng, and J. Huerta, "Wi-Fi crowdsourced fingerprinting dataset for indoor positioning," *Data*, vol. 2, no. 4, p. 32, 2017.

[32] X. Song et al., "A novel convolutional neural network based indoor localization framework with WiFi fingerprinting," *IEEE Access*, vol. 7, pp. 110698–110709, 2019.

[33] A. Katharopoulos and F. Fleuret, "Not all samples are created equal: Deep learning with importance sampling," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 2525–2534.

[34] C. Wu, Z. Yang, and C. Xiao, "Automatic radio map adaptation for indoor localization using smartphones," *IEEE Trans. Mobile Comput.*, vol. 17, no. 3, pp. 517–528, Mar. 2018.

[35] K. Liu, Z. Tian, Z. Li, and M. Zhou, "RFLoc: A reflector-assisted indoor Localization system using a single-antenna AP," *IEEE Trans. Instrum. Meas.*, vol. 70, pp. 1–16, 2021.

[36] Y. Tao, B. Huang, R. Yan, L. Zhao, and W. Wang, "CBWF: A lightweight circular-boundary-based WiFi fingerprinting localization system," *IEEE Internet Things J.*, vol. 11, no. 7, pp. 11508–11523, Apr. 2024.
[37] R. Battiti, A. Villani, and T. Le Nhat, "Neural network models for intelligent networks: Deriving the location from signal patterns," in *Proc. AINS*, 2002, pp. 1–13.
[38] H. Dai, W.-H. Ying, and J. Xu, "Multi-layer neural network for received signal strength-based indoor localisation," *IET Commun.*, vol. 10, no. 6, pp. 717–723, 2016.
[39] J. Yoo and H. J. Kim, "Target localization in wireless sensor networks using online semi-supervised support vector regression," *Sensors*, vol. 15, no. 6, pp. 12539–12559, 2015.
[40] S. A. Junoh and J.-Y. Pyun, "Enhancing indoor Localization with semi-crowdsourced fingerprinting and GAN-based data augmentation," *IEEE Internet Things J.*, vol. 11, no. 7, pp. 11945–11959, Apr. 2024.
[41] H. Sun, X. Zhu, Y. Liu, and W. Liu, "WiFi based fingerprinting positioning based on Seq2seq model," *Sensors*, vol. 20, no. 13, p. 3767, 2020.
[42] A. Alitaleshi, H. Jazayeriy, and J. Kazemitabar, "EA-CNN: A smart indoor 3D positioning scheme based on Wi-Fi fingerprinting and deep learning," *Eng. Appl. Artif. Intell.*, vol. 117, Jan. 2023, Art. no. 105509.
[43] J. Choi, G. Lee, S. Choi, and S. Bahk, "Smartphone based indoor path estimation and localization without human intervention," *IEEE Trans. Mobile Comput.*, vol. 21, no. 2, pp. 681–695, Feb. 2022.
[44] R. Li, H. Hu, and Q. Ye, "RFTrack: Stealthy location inference and tracking attack on Wi-Fi devices," *IEEE Trans. Inf. Forensics Security*, vol. 19, pp. 5925–5939, 2024.
[45] W. Liu, Y. Zhang, Z. Deng, and H. Zhou, "Low-cost indoor wireless fingerprint location database construction methods: A review," *IEEE Access*, vol. 11, pp. 37535–37545, 2023.
[46] A. Chatzimichail, A. Tsanousa, G. Meditskos, S. Vrochidis, and I. Kompatsiaris, "RSSI fingerprinting techniques for indoor localization datasets," in *Proc. 13th IMCL Conf. Internet Things, Infrastruct. Mobile Appl.*, 2021, pp. 468–479.
[47] R. C. Luo and T. J. Hsiao, "Dynamic wireless indoor localization incorporating with an autonomous mobile robot based on an adaptive signal model fingerprinting approach," *IEEE Trans. Ind. Electron.*, vol. 66, no. 3, pp. 1940–1951, Mar. 2019.
[48] Z. Tang, R. Gu, S. Li, K. S. Kim, and J. S. Smith, "Static vs. dynamic databases for indoor localization based on Wi-Fi fingerprinting: A discussion from a data perspective," in *Proc. Int. Conf. Artif. Intell. Inf. Commun. (ICAIIC)*, 2024, pp. 760–765.
[49] V. Bellavista-Parent, J. Torres-Sospedra, and A. Pérez-Navarro, "Comprehensive analysis of applied machine learning in indoor positioning based on Wi-Fi: An extended systematic review," *Sensors*, vol. 22, no. 12, p. 4622, 2022.
[50] J. Wang, Z. Zhao, M. Ou, J. Cui, and B. Wu, "Automatic update for Wi-Fi fingerprinting indoor localization via multi-target domain adaptation," *Proc. ACM Interactive, Mobile, Wearable Ubiquitous Technol.*, vol. 7, no. 2, pp. 1–27, 2023.
[51] G. Naik and J.-M. J. Park, "Coexistence of Wi-Fi 6E and 5G NR-U: Can we do better in the 6 GHz bands?" in *Proc. IEEE Conf. Comput. Commun.*, 2021, pp. 1–10.
[52] W. McKinney, "PANDAS: A foundational python library for data analysis and statistics," *Python High Perform. Sci. Comput.*, vol. 14, no. 9, pp. 1–9, 2011.
[53] F. Carpi et al., "Experimental analysis of RSSI-based localization algorithms with NLOS pre-mitigation for IoT applications," *Comput. Netw.*, vol. 225, Apr. 2023, Art. no. 109663.
[54] R. Bhirangi et al., "Hierarchical state space models for continuous sequence-to-sequence modeling," 2024, *arXiv:2402.10211*.
[55] Z. Zhang, A. Liu, I. Reid, R. Hartley, B. Zhuang, and H. Tang, "Motion Mamba: Efficient and long sequence motion generation," in *Proc. Eur. Conf. Comput. Vis.*, 2025, pp. 265–282.
[56] Z. Xiao, H. Wen, A. Markham, N. Trigoni, P. Blunsom, and J. Frolik, "Non-line-of-sight identification and mitigation using received signal strength," *IEEE Trans. Wireless Commun.*, vol. 14, no. 3, pp. 1689–1702, Mar. 2015.
[57] S. Herath, H. Yan, and Y. Furukawa, "RONIN: Robust neural inertial navigation in the wild: Benchmark, evaluations, & new methods," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2020, pp. 3146–3152.
[58] P. S. Varma and V. Anand, "Random forest learning based indoor localization as an IoT service for smart buildings," *Wireless Pers. Commun.*, vol. 117, pp. 3209–3227, Apr. 2021.
[59] V. Tilwari, S. Pack, M. Maduranga, and H. Lakmal, "An improved Wi-Fi RSSI-based indoor localization approach using deep randomized neural network," *IEEE Trans. Veh. Technol.*, Dec. 2024.
[60] H. Yan, Q. Shan, and Y. Furukawa, "RIDI: Robust IMU double integration," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 621–636.
[61] K. Sui et al., "Characterizing and improving WiFi latency in large-scale operational networks," in *Proc. 14th Annu. Int. Conf. Mobile Syst., Appl., Services*, 2016, pp. 347–360.
[62] "Wi-Fi scanning overview," 2024. [Online]. Available: https://developer.android.com/guide/topics/ connectivity/wifi-scan
[63] C. Lea, M. D. Flynn, R. Vidal, A. Reiter, and G. D. Hager, "Temporal convolutional networks for action segmentation and detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 156–165.
[64] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "CBAM: Convolutional block attention module," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018.
[65] A. Vaswani et al., "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1–15.

Yingying Wang received the B.E. degree in electronic engineering and the M.S. degree in signal processing from Northeastern University, Shenyang, Liaoning, China, in 2016 and 2019, respectively. She is currently pursuing the Ph.D. degree with the Department of Electronic Engineering, The Chinese University of Hong Kong, Hong Kong.

Her research interests are sensor fusion, low-cost indoor localization, and inertial navigation.

Hu Cheng received the B.E. degree in automation from Northeastern University, Shenyang, China, in 2016. He is currently pursuing the Ph.D. degree with the Department of Electronic Engineering, The Chinese University of Hong Kong, Hong Kong.

Max Q.-H. Meng (Fellow, IEEE) received the Ph.D. degree in electrical and computer engineering from the University of Victoria, Victoria, BC, Canada, in 1992.

He is currently a Chair Professor and the Head of the Department of Electronic and Electrical Engineering with the Southern University of Science and Technology, Shenzhen, China, on leave from the Department of Electronic Engineering, The Chinese University of Hong Kong, Hong Kong. He joined The Chinese University of Hong Kong, in 2001 as a Professor and later the Chairman with the Department of Electronic Engineering. He was with the Department of Electrical and Computer Engineering, the University of Alberta, Edmonton, AB, Canada, where he served as the Director of the Advanced Robotics and Teleoperation (ART) Laboratory and held the positions of an Assistant Professor in 1994, an Associate Professor in 1998, and a Professor in 2000, respectively. He is an Honorary Chair Professor with Harbin Institute of Technology, Harbin, China, and Zhejiang University, Hangzhou, China, and also the Honorary Dean with the School of Control Science and Engineering, Shandong University, Jinan, China. He has published more than 750 journal and conference papers and book chapters and led more than 60 funded research projects to completion as Principal Investigator. His research interests include medical and service robotics, robotics perception, and intelligence.

Prof. Meng is a recipient of the IEEE Millennium Medal. He has been serving as the Editor-in-Chief and editorial board of a number of international journals, including the Editor-in-Chief of the *Elsevier Journal of Biomimetic Intelligence and Robotics*, and as the General Chair or the Program Chair of many international conferences, including the General Chair of IROS 2005 and ICRA 2021, respectively. He served as an Associate VP for Conferences of the IEEE Robotics and Automation Society from 2004 to 2007, Co-Chair of the Fellow Evaluation Committee and an elected member of the AdCom of IEEE RAS for two terms. He is a Fellow of Hong Kong Institution of Engineers and an Academician of the Canadian Academy of Engineering.