SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

September 2026, Vol. 69, Iss. 9, 190302:1–190302:16
https://doi.org/10.1007/s11432-026-5068-2

Special Topic: User-Centric Intelligent Networking

Self-supervised resource-efficient indoor tracking with adaptive anchor scheduling

Haiyao YU, Geng WANG*, Yifan FENG, Gaoyang PANG, Branka VUCETIC & Yonghui LI

School of Electrical and Computer Engineering, The University of Sydney, Sydney NSW 2006, Australia

Received 2 February 2026/Revised 10 June 2026/Accepted 4 August 2026/Published online 26 August 2026

Abstract Wireless indoor localization and tracking systems often implicitly assume that all anchors are continuously available. However, this assumption is often impractical under bandwidth and energy constraints. With limited wireless resources, tracking systems must balance localization accuracy with sensing and communication overhead. Therefore, anchor scheduling is crucial as it enables the system to activate only the most informative anchors for a better accuracy-resource tradeoff. This paper proposes a self-supervised framework for joint tracking and anchor scheduling by first utilizing a physics-guided deep state space model to infer target positions and uncertainty without ground truth trajectories. Building on the learned latent representation, we further design an actor-critic-based reinforcement learning scheduler that proactively selects a resource-budgeted set of informative anchors for future ranging. Experiments on real-world ultra-wideband datasets demonstrate that the proposed method achieves a mean absolute localization error of 0.458 m, outperforming classical filters and learning-based baselines. By activating only 3 of the 8 anchors, the proposed method reduces the number of ranging requests by 62.5%. Complementary device-level measurements further show that this scheduling strategy reduces communication latency from 449.93 to 196.40 ms, corresponding to a 56.35% reduction.

Keywords indoor localization and tracking, self-supervised learning, deep state space model, anchor scheduling and selection, reinforcement learning

Citation Yu H Y, Wang G, Feng Y F, et al. Self-supervised resource-efficient indoor tracking with adaptive anchor scheduling. Sci China Inf Sci, 2026, 69(9): 190302, https://doi.org/10.1007/s11432-026-5068-2

1 Introduction

Indoor localization and tracking have become a fundamental enabling technology for next-generation wireless systems, supporting a wide range of emerging applications in 5G and beyond-5G networks, such as smart buildings, industrial automation, and intelligent transportation [1,2]. Beyond conventional connectivity services, future wireless networks are expected to natively support integrated sensing and communication (ISAC), in which localization plays a key role in enabling environment-aware and context-aware applications. As wireless networks continue to scale in user density and service diversity, indoor localization systems are increasingly required to achieve not only high positioning accuracy and low latency, but also high resource efficiency to ensure scalability and practical deployment [3].

Existing indoor localization technologies can be broadly categorized into vision-based, inertial-based, and anchor-based approaches [4]. Vision-based methods can achieve centimeter-level accuracy, but they rely on expensive hardware such as cameras [5] or LiDAR [6], raise privacy concerns, and are challenging to deploy in non-line-of-sight (NLoS) scenarios, all of which limit their large-scale adoption. Inertial measurement unit (IMU)-based localization offers a low-cost and infrastructure-free solution [7]. However, its accuracy degrades over time due to cumulative drift, making it unsuitable for long-term standalone tracking and often necessitating fusion with other technologies [8]. In contrast, anchor-based localization methods, such as Wi-Fi [9,10], Bluetooth Low Energy (BLE) [11], and ultra-wideband (UWB) [12] are widely adopted due to their low deployment cost, reuse of existing infrastructure, and capability to provide diverse measurement features.

In anchor-based tracking systems, user equipment (UE) typically initiates ranging or sensing requests to nearby anchors and receives measurements such as received signal strength (RSS), round-trip time (RTT), or channel state information (CSI) [4] that are subsequently transformed into distance- or angle-related observations for position estimation. However, in practical indoor environments with multipath propagation and NLoS conditions, the quality

* Corresponding author (email: geng.wang@sydney.edu.au)

of these measurements can vary significantly across anchors, leading to degraded localization performance [13, 14]. Extensive efforts have been devoted to mitigating NLoS effects and improving measurement reliability. For example, in [15], the authors proposed a floor plan-assisted positioning algorithm fused with pedestrian dead reckoning to improve accuracy. In [16], a customized hidden Markov model is used to infer the sight condition of the UE, while Ref. [10] employed a deep state space model (DSSM) to estimate error from measurements. However, the existing methods implicitly assume that all anchors are continuously active to provide measurements at every time interval.

Such an assumption becomes increasingly impractical in real-world deployments. In large-scale wireless networks, such as factories or shopping malls, multiple users may simultaneously request localization services. However, anchors, particularly Wi-Fi access points, must prioritize their primary communication tasks under limited bandwidth and energy budgets [17]. Activating all anchors for localization not only increases signaling overhead and energy consumption but also introduces latency and scalability challenges, thereby hindering the deployment of real-time and large-scale systems. This motivates the need for a joint tracking and anchor scheduling method, where only a subset of informative anchors is activated to balance tracking accuracy and resource consumption.

Existing studies on anchor scheduling primarily rely on model-based approaches, such as Bayesian filtering, heuristic rules, or optimization-based scheduling strategies [18–21]. These methods typically assume that measurements from all anchors are first collected, and they then evaluate the reliability of individual signal paths, e.g., LoS or NLoS, to improve tracking accuracy. However, such post-processing schemes do not reduce the sensing or communication overhead of the anchors, as all measurements are still acquired prior to selection. Recently, learning-based methods have achieved high positioning accuracy by learning complex, environment-dependent features [22]. For example, in [23], the authors formulate anchor selection for UWB localization as a classification task and train a neural network to predict near-optimal anchor activation probabilities from position estimates. However, conventional supervised-learning methods require extensive ground truth labels, making data collection labor-intensive and costly. Although reinforcement learning (RL)-based methods alleviate the need for labeled data by learning scheduling policies through interaction [24], their performance often degrades significantly when anchor layouts or propagation conditions change, limiting their generalization capability [25].

Motivated by these challenges, we propose a self-supervised framework for joint indoor tracking and anchor scheduling that integrates model-based structure with learning-based adaptability. The main contributions of this paper are summarized as follows.

- **Tracking-aware anchor scheduling formulation.** We formulate resource-constrained anchor scheduling as a tracking-aware sequential decision-making problem. Instead of selecting anchors only by predefined geometric or link-quality criteria, the proposed formulation optimizes a sequence of anchor-selection decisions under a fixed ranging budget, with the objective of improving the likelihood of the tracking observations.
- **Self-supervised likelihood-based scheduling reward.** We design a self-supervised reward computed from the negative DSSM reconstruction loss. This reward allows the scheduler to be trained without ground-truth positions or anchor labels. Since the reconstruction likelihood measures how well the selected ranging observations are explained by the inferred tracking state, it provides a task-driven signal for identifying anchor subsets that are informative and measurement-consistent.
- **Closed-loop DSSM-guided actor-critic scheduler.** We develop an actor-critic scheduler that selects anchors according to the posterior latent tracking state inferred by the DSSM. The selected ranging observations are then fed back to update the DSSM posterior in the next step. Through this interaction, anchor scheduling and probabilistic tracking are optimized as coupled components rather than as two separately designed modules. This closed-loop interaction allows anchor selection to be optimized directly for tracking accuracy under the radio-resource constraint.
- **Comprehensive real-world evaluation.** Experiments on real-world UWB datasets with four scenarios show that the proposed method consistently outperforms existing classical filters and learning-based baselines under the same resource budget, and achieves a favorable accuracy-resource tradeoff compared with random scheduling and the oracle best-anchor selection. By selecting 3 out of 8 anchors, the proposed method reduces the number of ranging requests by 62.5%. Complementary device-level WiFi FTM experiments further demonstrate a 56.35% reduction in communication latency, from 449.93 ms with all 8 anchors to 196.40 ms with 3 selected anchors, while alleviating anchor-side congestion and improving connection reliability.

Notations. In this paper, we use uppercase letters, e.g., X, to represent given constant numbers or parameters. Lowercase letters, e.g., x, are used to represent scalar variables. We use a calligraphic uppercase letter, e.g., $\mathcal{A}$, to denote a set, and $|\mathcal{A}|$ is its cardinality. In particular, $\{x_k\}_{k=1}^K$ denotes a set with given elements, i.e., $\{x_k\}_{k=1}^K = \{x_1, \ldots, x_K\}$, and $\mathbb{R}$ is the set of real numbers. We use an uppercase bold letter, e.g., $\mathbf{A}$, to represent a matrix, and a lowercase bold letter, e.g., $\mathbf{a}$, to denote a row vector. Superscript $\top$ denotes the transposition of a matrix or vector, e.g., $\mathbf{A}^\top$ and $\mathbf{a}^\top$, respectively. For matrices $\mathbf{A}$ and $\mathbf{B}$, we define $[\mathbf{A}, \mathbf{B}]$ as their horizontal

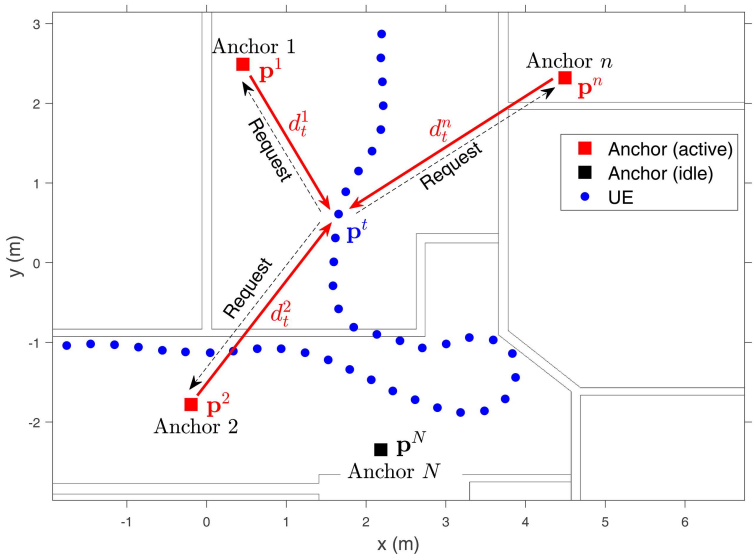

Figure 1 (Color online) Illustration of the indoor tracking scenario with scheduled and idle anchors.

concatenation, while $\begin{bmatrix}\mathbf{A}\\ \mathbf{B}\end{bmatrix} \triangleq [\mathbf{A}^\top, \mathbf{B}^\top]^\top$ is the vertical concatenation. Other notations will be specifically stated.

2 System model and problem formulation

As shown in Figure 1, we consider a total of N anchors in the scenario. The position of the n-th anchor is denoted by $\mathbf{p}^n = [x^n, y^n]$, which is fixed and known by the UE. Time is discretized into time intervals. In each time interval, the UE initiates ranging requests to selected anchors and receives corresponding distance measurements, where the selected set of anchors is denoted by $\mathcal{R}_t$ with $|\mathcal{R}_t| = R$, $N \geqslant R \geqslant 3$. We use a one-hot vector $\mathbf{r}_t = [r_t^1, \ldots, r_t^N]$ to represent the scheduling policy. Specifically, $r_t^n = 1$ for sending the request to the n-th anchor and $r_t^n = 0$ otherwise with $\sum_{n=1}^N r_t = R$. The received measurement at the UE from the n-th anchor in the t-th time interval is denoted by d_t^n, which is modeled as

$$d_t^n = \|\mathbf{p}_t - \mathbf{p}^n\| + \epsilon_t^n, \tag{1}$$

where ϵ_t^n is the measurement noise mainly from multipath and NLoS conditions.

The goal is to achieve accurate UE tracking while activating only a subset of anchors in each time step to conserve resources. We formulate this as a joint optimization problem that simultaneously infers the target positions and selects informative anchors. Let $\mathbf{s}_t = [\mathbf{p}_t, v_t^x, v_t^y]^\top \in \mathbb{R}^4$ denote the latent state of the target at time t, which includes the position $\mathbf{p}_t = [x_t, y_t]$ and velocity components v_t^x, v_t^y of UE. The tracking process is described by a state space model

$$\mathbf{s}_t = f(\mathbf{s}_{t-1}), \tag{2}$$
$$\mathbf{y}_t = g(\mathbf{s}_t, \mathbf{r}_t), \tag{3}$$

where $f(\cdot)$ and $g(\cdot)$ are the state transition and observation functions, respectively. The vector $\mathbf{y}_t = \{d_t^n\}, \forall n \in \mathcal{R}_t$ is the observation vector determined by the scheduling vector $\mathbf{r}_t$. To jointly optimize tracking and scheduling without ground truth positions, we maximize the marginal likelihood of the observed distance measurements under the selected anchors

$$\max_{\theta, \mathbf{r}_{1:T}} \log p_\theta(\mathbf{y}_{1:T} \mid \mathbf{r}_{1:T}), \quad \text{s.t.} \quad \sum_{n=1}^N r_t^n = R, \quad \forall t, \tag{4}$$

where θ denotes the parameters of the tracking module and scheduling module. The marginal likelihood can be factorized using the state space model

$$\log p_\theta(\mathbf{y}_{1:T} \mid \mathbf{r}_{1:T}) = \log \int \prod_{t=1}^T g_\theta(\mathbf{y}_t \mid \mathbf{s}_t, \mathbf{r}_t)\, f_\theta(\mathbf{s}_t \mid \mathbf{s}_{t-1})\, d\mathbf{s}_{1:T}. \tag{5}$$

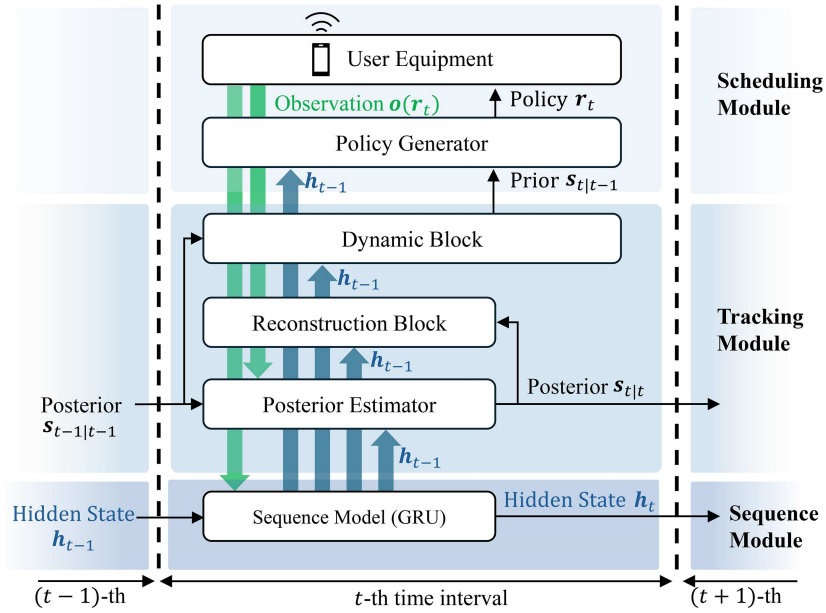

Figure 2 (Color online) Proposed joint tracking and anchor scheduling framework.

Intuitively, this objective encourages the scheduler to select anchors that yield measurements most consistent with the inferred UE positions, thereby improving tracking accuracy while respecting the resource constraint R. Since direct optimization of (4) is intractable due to the integration over latent states, we employ a DSSM with variational inference to approximate the posterior and enable efficient training, as detailed in the following sections.

3 Proposed joint indoor tracking and scheduling framework

Figure 2 illustrates the framework of our proposed joint tracking and scheduling framework that consists of three main modules: (1) the sequence module, (2) the tracking module, and (3) the scheduling module. At the t-th time interval, the inputs to the system are the posterior state $\mathbf{s}_{t-1|t-1}$ and the hidden state of the sequence module $\mathbf{h}_{t-1}$ from the previous interval. The tracking module then takes $\mathbf{h}_{t-1}$ with $\mathbf{s}_{t-1|t-1}$ to estimate the prior state $\mathbf{s}_{t|t-1}$. Subsequently, $\mathbf{s}_{t|t-1}$ and $\mathbf{h}_{t-1}$ are passed to the scheduling module, which outputs the anchor selection policy $\mathbf{r}_t$. Then, based on the observation $\mathbf{o}(\mathbf{r}_t)$, the posterior state $\mathbf{s}_{t|t}$ and the reconstructed measurements are computed. Finally, the gated recurrent unit (GRU)-based sequence module updates the hidden state $\mathbf{h}_t$ using observations $\mathbf{o}(\mathbf{r}_t)$ and $\mathbf{h}_{t-1}$. Details of these modules will be elaborated in subsequent sections.

In practical deployment, the proposed framework is executed on the UE (e.g., a smartphone) as an on-device inference pipeline, while anchors operate as standard infrastructure nodes that only respond to ranging requests. When entering a new environment, the UE first collects a set of ranging measurements by interacting with nearby anchors, and the model is trained offline using the recorded ranging measurements. After the training converges, the trained model parameters are deployed on the UE to enable online tracking and proactive anchor scheduling in real time.

4 Deep learning-based sequence and tracking modules

Following the DSSM literature in [10,26–28], we customized a DSSM backbone as the sequence and tracking modules. Specifically, the DSSM leverages stochastic recurrent latent dynamics, physics-guided structure, and variational inference to jointly model UE positions and uncertainty for tracking from unlabeled measurement sequences. In this section, we first elaborate on the sequence and tracking modules, respectively. Then, we detail the training process for them.

4.1 Tracking module

The input of the tracking module includes the hidden state $\mathbf{h}_{t-1}$ from the sequence model, the posterior of the previous latent state $\mathbf{s}_{t-1|t-1}$, and the observation $\mathbf{o}(\mathbf{r}_t)$. The output is the posterior state $\mathbf{s}_{t|t}$, which includes the estimated location of the UE and corresponding velocity in the t-th time interval. The module consists of three blocks: (1) the dynamic block, (2) the posterior estimator block, and (3) the reconstruction block.

4.1.1 *Dynamic block*

The dynamic block predicts the prior state $\mathbf{s}_{t|t-1}$ of the UE based on the hidden state $\mathbf{h}_{t-1}$ of the sequence module and the posterior state $\mathbf{s}_{t-1|t-1}$ of the previous time interval. The prior state is modeled as a Gaussian distribution with diagonal covariance

$$\mathbf{s}_{t|t-1} \sim \mathcal{N}\Big(\boldsymbol{\mu}_{t|t-1}, \mathrm{diag}\big\{\big(\boldsymbol{\sigma}_{t|t-1}\big)^2\big\}\Big), \tag{6}$$

where $\boldsymbol{\mu}_{t|t-1} \in \mathbb{R}^{4\times 1}$ is the mean vector and $\mathrm{diag}\big\{\big(\boldsymbol{\sigma}_{t|t-1}\big)^2\big\} \in \mathbb{R}^{4\times 4}$ is the diagonal covariance matrix. Specifically, the mean vector $\boldsymbol{\mu}_{t|t-1}$ is estimated based on a physics-based consistency and a learnable correction

$$\boldsymbol{\mu}_{t|t-1} = f^{\mathrm{phy}}(\mathbf{s}_{t-1|t-1}) + f_{\mu}^{\mathrm{prior}}(\mathbf{h}_{t-1}, \mathbf{s}_{t-1|t-1}; \boldsymbol{\theta}_{\boldsymbol{\mu}}^{\mathrm{prior}}) = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \mathbf{s}_{t-1|t-1} + f_{\mu}^{\mathrm{prior}}(\mathbf{h}_{t-1}, \mathbf{s}_{t-1|t-1}; \boldsymbol{\theta}_{\boldsymbol{\mu}}^{\mathrm{prior}}), \tag{7}$$

where $f^{\mathrm{phy}}(\mathbf{s}_{t-1|t-1})$ is a deterministic physics-based predictor with a constant-velocity model [29]. It guides the prior prediction to follow physically plausible motion. To capture deviations from the model, we introduce a learnable residual correction that is modelled as a feed-forward neural networks (FNNs) $f_{\mu}^{\mathrm{prior}}(\cdot; \boldsymbol{\theta}_{\mu}^{\mathrm{prior}})$ with corresponding trainable parameters $\boldsymbol{\theta}_{\boldsymbol{\mu}}^{\mathrm{prior}}$. The prior uncertainty $\boldsymbol{\sigma}_{t|t-1}$ is predicted by

$$\boldsymbol{\sigma}_{t|t-1} = f_{\sigma}^{\mathrm{prior}}(\mathbf{h}_{t-1}, \mathbf{s}_{t-1|t-1}; \boldsymbol{\theta}_{\boldsymbol{\sigma}}^{\mathrm{prior}}), \tag{8}$$

where the $f_{\sigma}^{\mathrm{prior}}(\cdot; \boldsymbol{\theta}_{\boldsymbol{\sigma}}^{\mathrm{prior}})$ is the FNN with corresponding trainable parameters $\boldsymbol{\theta}_{\boldsymbol{\sigma}}^{\mathrm{prior}}$.

4.1.2 *Posterior estimator*

The posterior estimator predicts the posterior state $\mathbf{s}_{t|t}$ of the UE based on activated anchor location $\mathbf{P}(\mathbf{r}_t)$ and corresponding observations $\mathbf{o}(\mathbf{r}_t)$, the hidden state of the sequence model $\mathbf{h}_{t-1}$, and its previous posterior state $\mathbf{s}_{t-1|t-1}$. In the t-th time interval, the posterior state $\mathbf{s}_{t|t}$ is estimated as a Gaussian-distributed random variable

$$\mathbf{s}_{t|t} \sim \mathcal{N}\Big(\boldsymbol{\mu}_{t|t}, \mathrm{diag}\big\{\big(\boldsymbol{\sigma}_{t|t}\big)^2\big\}\Big), \tag{9}$$

where $\boldsymbol{\mu}_{t|t} \in \mathbb{R}^{4\times 1}$ is the mean vector and $\mathrm{diag}\big\{\big(\boldsymbol{\sigma}_{t|t}\big)^2\big\} \in \mathbb{R}^{4\times 4}$ is the diagonal covariance matrix of the posterior state. These parameters are expressed as

$$[\boldsymbol{\mu}_{t|t}; \boldsymbol{\sigma}_{t|t}] = f^{\mathrm{pos}}\left(\mathbf{s}_{t-1|t-1}, \mathbf{h}_{t-1}, \mathbf{o}(\mathbf{r}_t), \mathbf{P}(\mathbf{r}_t); \boldsymbol{\theta}^{\mathrm{pos}}\right), \tag{10}$$

where $\mathbf{r}_t$ is the node selection policy in the t-th time interval, and details of the policy generation will be elaborated in Section 5. The activated anchor location $\mathbf{P}(\mathbf{r}_t) = [\mathbf{p}^1, \ldots, \mathbf{p}^N]$, where $\mathbf{p}^n = [0, 0]$, for $\forall n \notin \mathcal{R}_t$. The $\mathbf{o}(\mathbf{r}_t)$ represents the corresponding observations, $\mathbf{o}(\mathbf{r}_t) = [d_t^1, \ldots, d_t^n]$, where $d_t^n = 0$, for $\forall n \notin \mathcal{R}_t$. The $f^{\mathrm{pos}}(\cdot; \boldsymbol{\theta}^{\mathrm{pos}})$ is the attention-based neural network with the corresponding trainable parameters $\boldsymbol{\theta}^{\mathrm{pos}}$. Specifically, for each anchor $n \in \{1, \ldots, N\}$, we construct an anchor-wise token by concatenating the previous posterior state, the ranging observation, the anchor location, and the sequence-model hidden state as

$$\mathbf{u}_t^n = f_{\mathrm{u}}\big([\mathbf{s}_{t-1|t-1},\, d_t^n,\, \mathbf{p}^n,\, \mathbf{h}_{t-1}]; \boldsymbol{\theta}_{\mathrm{u}}\big) \in \mathbb{R}^{U\times 1}, \tag{11}$$

where $f_{\mathrm{u}}(\cdot; \boldsymbol{\theta}_{\mathrm{u}})$ is the FNN with corresponding trainable parameters $\boldsymbol{\theta}_{\mathrm{u}}$. The resulting embedding is concatenated as a matrix and then processed by a multi-head self-attention module to capture inter-anchor dependencies and measurement consistency as

$$[\tilde{\mathbf{u}}_t^1, \tilde{\mathbf{u}}_t^2, \ldots, \tilde{\mathbf{u}}_t^N]^{\top} = f_{\mathrm{MHA}}\big([\mathbf{u}_t^1, \mathbf{u}_t^2, \ldots, \mathbf{u}_t^N]^{\top}, \mathbf{r}_t; \boldsymbol{\theta}_{\mathrm{M}}\big), \tag{12}$$

where the $\mathbf{r}_t$ serves as the mask to avoid the attention focus on the invalid ranging observations. The $f_{\mathrm{MHA}}(\cdot;\boldsymbol{\theta}_{\mathrm{M}})$ is the multi-head attention model, which has the same architecture as the transformer encoder approach in [30]. Finally, we apply masked mean pooling over the active anchors to obtain a fixed-length representation

$$\bar{\mathbf{u}}_t = \frac{1}{R}\sum_{n=1}^{N} r_t^n\, \tilde{\mathbf{u}}_t^n, \tag{13}$$

and the posterior parameters are produced via two FNNs $f_{\mu}^{\mathrm{pos}}(\cdot;\boldsymbol{\theta}_{\mu}^{\mathrm{pos}})$ and $f_{\sigma}^{\mathrm{pos}}(\cdot;\boldsymbol{\theta}_{\sigma}^{\mathrm{pos}})$, respectively, which are given by

$$\boldsymbol{\mu}_{t|t} = f_{\mu}^{\mathrm{pos}}(\bar{\mathbf{u}}_t;\boldsymbol{\theta}_{\mu}^{\mathrm{pos}}), \quad \boldsymbol{\sigma}_{t|t} = \mathrm{softplus}(f_{\sigma}^{\mathrm{pos}}(\bar{\mathbf{u}}_t;\boldsymbol{\theta}_{\sigma}^{\mathrm{pos}})). \tag{14}$$

We note that the location in the posterior state is the predicted location of the UE in the t-th time interval, given by $[x_t^{\mathrm{pos}}, y_t^{\mathrm{pos}}] = \mathbf{s}_{t|t}[:2]$.

4.1.3 *Reconstruction block*

The reconstruction block evaluates the likelihood of each selected measurement and uses its log-likelihood as the self-supervised learning objective. Specifically, the reconstructed ranging measurements at the t-th time interval $\{\hat{d}_t^n\}\forall n \in \mathcal{R}_t$, are modeled as a Gaussian-distributed random variable, given by

$$\hat{d}_t^n \sim \mathcal{N}(\mu_{\mathrm{d},t}^n, (\sigma_{\mathrm{d},t}^n)^2), \forall n \in \mathcal{R}_t, \tag{15}$$

and the mean value is estimated by

$$\mu_{\mathrm{d},t}^n = ||\mathbf{p}_t^{\mathrm{pos}} - \mathbf{p}^n||, \tag{16}$$

where $\mathbf{p}_t^{\mathrm{pos}} = [x_t^{\mathrm{pos}}, y_t^{\mathrm{pos}}] = \mathbf{s}_{t|t}[:2]$ is the posterior location. The variance is estimated by

$$\sigma_{\mathrm{d},t}^n = f^{\mathrm{re}}(\mathbf{s}_{t|t}, \mathbf{h}_{t-1}, \mathbf{p}^n; \boldsymbol{\theta}^{\mathrm{re}}), \tag{17}$$

where $f^{\mathrm{re}}(\cdot;\boldsymbol{\theta}^{\mathrm{re}})$ is the FNN with corresponding trainable parameters $\boldsymbol{\theta}^{\mathrm{re}}$.

4.2 Sequence module

The sequence module is implemented by a GRU that acts as the recurrent memory of the DSSM. It updates $\mathbf{h}_t$ by summarizing the historical measurements, thereby capturing long-term motion context and environment-dependent ranging behaviors. The input of the sequence module includes the current selected ranging observations $\mathbf{o}(\mathbf{r}_t)$ and the hidden state $\mathbf{h}_{t-1}$ of the previous time interval. The output is the updated hidden state $\mathbf{h}_t$. Specifically, we use a double-layer GRU with a hidden size $\mathbf{h}_t \in \mathbb{R}^{H\times 1}$. Let $f_{\mathrm{h}}(\cdot)$ denote the GRU operation; then the hidden state output is expressed as

$$\mathbf{h}_t = f_{\mathrm{h}}(\mathbf{o}(\mathbf{r}_t), \mathbf{h}_{t-1}; \boldsymbol{\theta}^{\mathrm{h}}), \tag{18}$$

where $\boldsymbol{\theta}^{\mathrm{h}}$ denotes the trainable parameters of the GRU.

4.3 Training of sequence and tracking module

In reality, obtaining ground truth trajectories in indoor environments is costly or infeasible. Therefore, we train the sequence and tracking module in a self-supervised manner by maximizing the likelihood of the observed ranges conditioned on the posterior state. Specifically, we minimize the following self-supervised loss function over a trajectory of length T as

$$\mathcal{L}(\boldsymbol{\theta}^{\mathrm{h}}, \boldsymbol{\theta}_{\boldsymbol{\mu}}^{\mathrm{prior}}, \boldsymbol{\theta}_{\boldsymbol{\sigma}}^{\mathrm{prior}}, \boldsymbol{\theta}^{\mathrm{pos}}, \boldsymbol{\theta}^{\mathrm{re}}) = \sum_{t=1}^{T}(\mathcal{L}_t^{\mathrm{re}} + \mathcal{L}_t^{\mathrm{dyn}}), \tag{19}$$

where $\mathcal{L}_t^{\mathrm{re}}$ is the reconstruction loss and $\mathcal{L}_t^{\mathrm{dyn}}$ is the dynamic loss. We note that minimizing the self-supervised loss in (19) is equivalent to maximizing the evidence lower bound of the marginal-likelihood objective in (4) as derived in [31].

Reconstruction loss. The reconstruction loss $\mathcal{L}_t^{\mathrm{re}}$ computes the conditional likelihood of the measurements, which is defined as the log-likelihood over the selected anchors. It is given by

$$\mathcal{L}_t^{\mathrm{re}} = -\mathbb{E}\left(\sum_{n\in\mathcal{R}_t} \log p(d_t^n \mid \mathbf{s}_{t|t}, \mathbf{h}_{t-1}, \mathbf{p}^n)\right), \tag{20}$$

where

$$p(d_t^n \mid \mathbf{s}_{t|t}, \mathbf{h}_{t-1}, \mathbf{p}^n) = \mathcal{N}\Big(d_t^n; \mu_{\mathrm{d},t}^n, \big(\sigma_{\mathrm{d},t}^n\big)^2\Big), \quad n \in \mathcal{R}_t. \tag{21}$$

Dynamic loss. $\mathcal{L}_t^{\mathrm{dyn}}$ is the dynamic loss that regularizes the posterior state to be consistent with the learned dynamics prior. Specifically, we leverage the Kullback-Leibler (KL) divergence between the prior and posterior state distribution as

$$\mathcal{L}_t^{\mathrm{dyn}} = \mathrm{KL}\Big(\mathcal{N}(\boldsymbol{\mu}_{t|t}, \mathrm{diag}(\boldsymbol{\sigma}_{t|t}^2)) \,\Big\|\, \mathcal{N}(\boldsymbol{\mu}_{t|t-1}, \mathrm{diag}(\boldsymbol{\sigma}_{t|t-1}^2))\Big). \tag{22}$$

The reconstruction loss encourages the latent state $\mathbf{s}_{t|t}$ to explain the current ranging observation d_t^n by maximizing the measurement likelihood, so that the learned representation captures environment- and time-dependent measurement characteristics. The dynamic loss $\mathcal{L}_t^{\mathrm{dyn}}$, implemented as a KL divergence between the variational posterior and the transition prior, regularizes the inferred state to be consistent with the learned dynamics model. This prevents the posterior from overfitting to per-step noisy measurements and enforces temporal coherence, thereby enabling the model to learn robust tracking.

5 RL-based scheduling module

In each time interval, the UE is only allowed to activate a subset of anchors due to the resource constraint in (4). This is a standard decision-making problem that can be formulated as a Markov decision process and solved by RL. To proactively select informative anchors without relying on ground truth trajectories, we develop an RL-based scheduling module that leverages the internal latent representation of the sequence and tracking modules. The key idea is to treat the learned tracking module as a differentiable "environment" that provides a self-supervised reward based on measurement reconstruction likelihood, such that the scheduler is encouraged to select anchors that are most informative for tracking. The actor, critic model, and the corresponding training process are defined as follows.

5.1 Actor-critic RL model

The actor outputs the scheduling policy $\mathbf{r}_t$ of the UE based on the sampled latent variable from the previous interval and the GRU hidden state $\mathbf{h}_{t-1}$. A feature extractor first produces

$$\boldsymbol{\ell}_t = f^{\mathrm{s}}(\mathbf{x}_t; \boldsymbol{\theta}^{\mathrm{s}}), \tag{23}$$

where $\mathbf{x}_t = [\mathbf{s}_{t|t-1}^\top, \mathbf{h}_{t-1}^\top]^\top$, $f^{\mathrm{s}}(\cdot; \boldsymbol{\theta}^{\mathrm{s}})$ is the FNN with corresponding trainable parameters $\boldsymbol{\theta}^{\mathrm{s}}$. Then the actor head outputs

$$\boldsymbol{a}_t = f^{\pi}(\boldsymbol{\ell}_t; \boldsymbol{\theta}^{\pi}), \tag{24}$$

where $\boldsymbol{a}_t \in \mathbb{R}^{N\times 1}$ is the anchor-wise logits, $f^{\pi}(\cdot; \boldsymbol{\theta}^{\pi})$ is the FNN with corresponding trainable parameters $\boldsymbol{\theta}^{\pi}$. To satisfy the cardinality constraint, we select R anchors by sequential sampling from the categorical distribution without replacement

$$\mathbf{r}_t \sim \mathrm{Cat}(\mathrm{softmax}(\boldsymbol{a}_t)). \tag{25}$$

The critic model learns to evaluate the expected reward to guide the actor based on the rewards. We use reconstruction loss in (20) as the reward $R_t = -\mathcal{L}_t^{\mathrm{re}}$ in the t-th time interval. This likelihood-based reward has two advantages for anchor scheduling. First, it is self-supervised and can be computed directly from the selected ranging measurements without requiring ground truth positions. Second, it reflects the consistency between the received measurements and the posterior latent tracking state, thereby encouraging the scheduler to select anchors that are both informative and reliable for tracking. Specifically, when a selected anchor $n \in \mathcal{R}_t$ is suddenly affected by obstacles, the corresponding ranging measurement d_t^n is likely to be inconsistent with the posterior tracking state $\mathbf{s}_{t|t}$ inferred by the DSSM. In addition, the DSSM models the measurement noise stochastically through the reconstruction distribution, so an abnormal NLoS measurement is associated with a larger reconstruction variance. The larger uncertainty reduces the reconstruction likelihood, resulting in a larger reconstruction loss and a lower reward. As a result, the scheduler is discouraged from repeatedly selecting unreliable anchors. The critic head outputs a scalar state value

$$V_t = f^{\mathrm{v}}(\boldsymbol{\ell}_t; \boldsymbol{\theta}^{\mathrm{v}}), \tag{26}$$

where $f^{\mathrm{v}}(\cdot; \boldsymbol{\theta}^{\mathrm{v}})$ is the FNN with corresponding trainable parameters $\boldsymbol{\theta}^{\mathrm{v}}$, and the output V_t serves as a baseline to reduce the variance of policy gradients.

5.2 Training objective and optimization

For a trajectory, the discounted return from time t is defined as

$$G_t = \sum_{\tau=t}^{T} \gamma^{\tau-t} R_\tau, \tag{27}$$

where $\gamma \in (0, 1]$ is the discount factor. The advantage function is computed by

$$A_t = G_t - V_t. \tag{28}$$

Actor loss. The actor model is trained to maximize the expected return using an advantage-weighted policy gradient

$$\mathcal{L}_t^{\text{act}} = -\mathbb{E}\left[\log \pi_{\theta_\pi}(\mathbf{r}_t|\mathbf{x}_t)\, A_t\right]. \tag{29}$$

Critic loss. The critic model is trained by regression to the Monte-Carlo return

$$\mathcal{L}_t^{\text{crt}} = \mathbb{E}\left[(V_t - G_t)^2\right]. \tag{30}$$

We note that the scheduling module is trained jointly with the DSSM. Specifically, the DSSM parameters are updated by minimizing the self-supervised loss in (19), while the actor and critic are updated by minimizing $\mathcal{L}(\boldsymbol{\theta}^{\text{s}}, \boldsymbol{\theta}^{\pi}, \boldsymbol{\theta}^{\text{v}}) = \mathcal{L}_{\text{act}} + \mathcal{L}_{\text{crt}}$. Following the implementation, we detach the scheduling state $\mathbf{x}_t$ and the reward R_t from the DSSM computation graph, such that the RL gradients do not back-propagate into the tracking module. This decoupled update stabilizes training and allows the scheduler to learn from the likelihood-based reward while preserving the variational learning of the DSSM.

The complete training procedure of the proposed joint indoor tracking and anchor scheduling framework is summarized in Algorithm 1.

Algorithm 1 Proposed joint indoor tracking and anchor scheduling framework.

Input: Number of training epochs E.
Initialization: $\boldsymbol{\theta}^{\text{h}}, \boldsymbol{\theta}_{\boldsymbol{\mu}}^{\text{prior}}, \boldsymbol{\theta}_{\boldsymbol{\sigma}}^{\text{prior}}, \boldsymbol{\theta}^{\text{pos}}, \boldsymbol{\theta}^{\text{re}}, \boldsymbol{\theta}^{\text{s}}, \boldsymbol{\theta}^{\pi}, \boldsymbol{\theta}^{\text{v}}$.

for epoch in range $[1, E]$ **do**
  Initialize posterior $\mathbf{s}_{0|0}$ and hidden state $\mathbf{h}_0$;
  for $t = 1 : T$ **do**
    The dynamic block estimates the prior state $\mathbf{s}_{t|t-1} \sim \mathcal{N}\Big(\boldsymbol{\mu}_{t|t-1}, \text{diag}\big\{\big(\boldsymbol{\sigma}_{t|t-1}\big)^2\big\}\Big)$, where $\boldsymbol{\mu}_{t|t-1}$ and $\boldsymbol{\sigma}_{t|t-1}$ are estimated via (7) and (8), respectively;
    The policy generator construct the $\mathbf{x}_t = [\mathbf{s}_{t|t-1}^\top, \mathbf{h}_{t-1}^\top]^\top$ and generates the scheduling strategy $\mathbf{r}_t$ and scalar state value V_t via (23)–(26), respectively;
    The posterior estimator estimate the posterior state $\mathbf{s}_{t|t} \sim \mathcal{N}\Big(\boldsymbol{\mu}_{t|t}, \text{diag}\big\{\big(\boldsymbol{\sigma}_{t|t}\big)^2\big\}\Big)$, where $\boldsymbol{\mu}_{t|t}$ and $\boldsymbol{\sigma}_{t|t}$ are estimated via $[\boldsymbol{\mu}_{t|t}; \boldsymbol{\sigma}_{t|t}] = f^{\text{pos}}\big(\mathbf{s}_{t-1|t-1}, \mathbf{h}_{t-1}, \mathbf{o}(\mathbf{r}_t), \mathbf{P}(\mathbf{r}_t); \boldsymbol{\theta}^{\text{pos}}\big)$;
    The reconstruction block reconstructs measurements $\hat{d}_t^n \sim \mathcal{N}(\mu_{\text{d},t}^n, (\sigma_{\text{d},t}^n)^2), \forall n \in \mathcal{R}_t$;
    Compute the reconstruction and dynamic loss $\mathcal{L}_t^{\text{re}}$ and $\mathcal{L}_t^{\text{dyn}}$ via (20) and (22), respectively;
    Compute the actor loss $\mathcal{L}_t^{\text{act}}$ and critic loss $\mathcal{L}_t^{\text{crt}}$ via (29) and (30), respectively;
    Update the hidden state of the sequence model $\mathbf{h}_t = f_{\text{h}}(\mathbf{o}(\mathbf{r}_t), \mathbf{h}_{t-1}; \boldsymbol{\theta}^{\text{h}})$;
  end for
  Compute the sequence and tracking module loss $\mathcal{L}(\boldsymbol{\theta}^{\text{h}}, \boldsymbol{\theta}_{\boldsymbol{\mu}}^{\text{prior}}, \boldsymbol{\theta}_{\boldsymbol{\sigma}}^{\text{prior}}, \boldsymbol{\theta}^{\text{pos}}, \boldsymbol{\theta}^{\text{re}}) = \sum_{t=1}^{T}(\mathcal{L}_t^{\text{re}} + \mathcal{L}_t^{\text{dyn}})$;
  Compute scheduling model loss $\mathcal{L}(\boldsymbol{\theta}^{\text{s}}, \boldsymbol{\theta}^{\pi}, \boldsymbol{\theta}^{\text{v}}) = \sum_{t=1}^{T}(\mathcal{L}_t^{\text{act}} + \mathcal{L}_t^{\text{crt}})$;
  Update the parameters of $\boldsymbol{\theta}^{\text{h}}, \boldsymbol{\theta}_{\boldsymbol{\mu}}^{\text{prior}}, \boldsymbol{\theta}_{\boldsymbol{\sigma}}^{\text{prior}}, \boldsymbol{\theta}^{\text{pos}}, \boldsymbol{\theta}^{\text{re}}$ based on $\mathcal{L}(\boldsymbol{\theta}^{\text{h}}, \boldsymbol{\theta}_{\boldsymbol{\mu}}^{\text{prior}}, \boldsymbol{\theta}_{\boldsymbol{\sigma}}^{\text{prior}}, \boldsymbol{\theta}^{\text{pos}}, \boldsymbol{\theta}^{\text{re}})$;
  Update the parameters $\boldsymbol{\theta}^{\text{s}}, \boldsymbol{\theta}^{\pi}, \boldsymbol{\theta}^{\text{v}}$ based on $\mathcal{L}(\boldsymbol{\theta}^{\text{s}}, \boldsymbol{\theta}^{\pi}, \boldsymbol{\theta}^{\text{v}})$;
end for

6 Experimental results

In this section, we present extensive experimental results to evaluate the performance of the proposed framework. First, the experiment setup and the hyper-parameters of various blocks in the proposed framework are summarized, and the baselines and performance metrics are introduced. We then evaluate and discuss the localization accuracy of the proposed joint tracking and scheduling framework in the four different scenarios in Subsection 6.2. Beyond tracking accuracy, we further evaluate the resource efficiency in terms of FTM communication latency and anchor-side service capacity in Subsection 6.3, while Subsection 6.4 quantifies the practical on-device computational overhead of the proposed framework.

Table 1 The hyperparameters of the sequence, tracking, and scheduling module.

Block	Layer	Input shape	Output shape	Remark
Sequence module				
GRU	GRU1	8	50	First GRU layer, $H = 50$
	GRU2	50	50	Second GRU layer with 0.2 dropout rate
Tracking module				
Dynamic block	Shared embedding	54	50	A feed forward neural network with layer norm and Relu
		50	50	Second fully-connected layer
	Mean head	50	4	A feed forward neural network
	Variance head	50	4	A feed forward neural network with Softplus
Posterior estimator	Measurement embedding	8×57	8×64	A feed forward neural network with layer norm and Relu
		8×64	8×64	Second feed forward neural network Relu
		8×64	8×64	Third feed forward neural network with Relu
	Attention	8×64	8×64	A self-attention mechanism
	Mean	8×64	1×64	Mean function
	Mean head	1×64	1×4	A feed forward neural network
	Variance head	1×64	1×4	A feed forward neural network
Reconstruction block	Variance head	56	1	A feed forward neural network with Softplus
Scheduling module				
Actor	Shared feature extractor	54	128	A feed forward neural network with Relu
	Actor head	128	8	A feed forward neural network
	Sequential sampling	8	3	Sample $k = 3$ anchors w/o replacement
Critic	Value head	128	1	A feed forward neural network

6.1 Experiment setup

6.1.1 *Dataset and system parameters*

We evaluate the proposed framework on the UWB indoor localization dataset in [32]. The dataset was collected using commercial DecaWave DW1000 UWB modules deployed as the anchor. Measurements are obtained from reference signals with carrier frequencies ranging from 3.4944 to 6.4896 GHz and bandwidths of 499.2 MHz or 900 MHz. To evaluate the proposed framework, we consider four different indoor scenarios. The number of selected anchors is set to $R = 3$, and the total number of anchors is $N = 8$. The sequence length is set to $T = 50$. The discount factor is set to $\gamma = 0.99$. During the training of the proposed framework, we use Adam optimizer, cosine annealing scheduler with a learning rate of 1×10^{-3}, and a weight decay of 1×10^{-3} [33]. The number of epochs is set to 200, and the batch size is set to 32. The hardware for data processing and training is a desktop with Windows 11 Pro, Intel(R) Core(TM) i9-12900KF CPU @ 3.20 GHz, NVIDIA GeForce RTX 3090, 64 GB Memory, and 2 TB SSD. The main hyperparameters of each module are summarized in Table 1.

6.1.2 *Baselines*

We introduce the following methods for comparison with our proposed framework.

• Proposed-*full anchors*. The DSSM that is trained with full N anchors without the scheduling module. This baseline represents the existing methods, which implicitly assume that all anchors are continuously active to provide measurements at every time interval.

• Proposed-*random scheduling*. The proposed indoor localization system is evaluated based on the ranging measurements from 3 anchors that are randomly selected.

• Proposed-*oracle*. The proposed localization and scheduling strategies use $R = 3$ anchors selected at each time step as those with the smallest absolute ranging errors, where the ranging error of anchor n is defined as $|d_t^n - \|\mathbf{p}_t^{\mathrm{GT}} - \mathbf{p}^n\||$. We note that this baseline is reported as an oracle selection for the proposed system, since the anchors are selected based on ground truth distances.

• Proposed scheduling. The proposed joint tracking and scheduling framework that proactively selects $R = 3$ anchors at each time step using the learned actor-critic scheduler.

• EKF-*full anchors*. An extended Kalman filter (EKF) that utilizes ranging measurements from all anchors to track the UE at each time interval. Specifically, a constant velocity transition model is adopted [10]. The covariance matrix of the acceleration-driven process noise $\mathbf{Q} \in \mathbb{R}^{4\times4}$ is set to $\sigma_{\mathrm{acc}} = 0.5$ m/s^2. The standard deviation of the

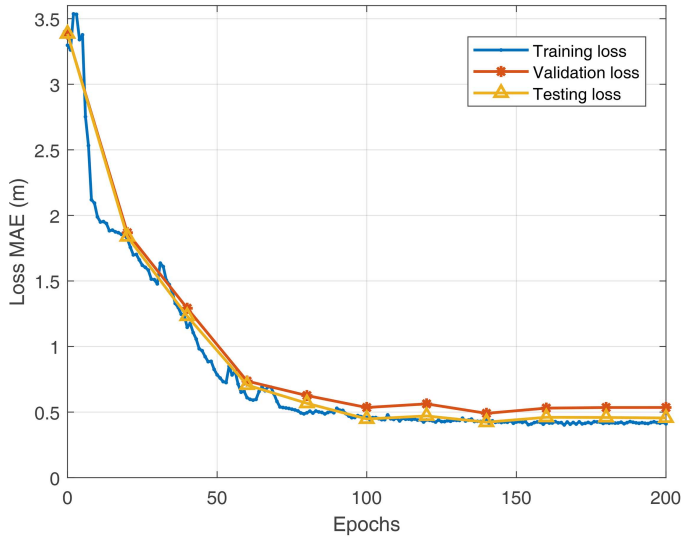

Figure 3 (Color online) Loss of the proposed indoor localization system.

measurement noise in the measurement covariance matrix $\mathbf{R} \in \mathbb{R}^{4\times4}$ is set to $\sigma_n = 1$ m.

• EKF-*random scheduling*. EKF is evaluated based on $R = 3$ anchors randomly selected from all anchors to track the UE.

• EKF-*oracle*. It uses $R = 3$ anchors whose ranging measurements have the smallest absolute ranging errors among all anchors at each time step. We note that this involves the use of ground truth data and is reported as an oracle anchor selection.

• PF-*full anchors*. A particle filter (PF) is employed to track the UE using ranging measurements from all anchors at each time interval. Specifically, a constant-velocity state transition model is adopted with $N_p = 1000$ particles. The standard deviation of the acceleration-driven process noise is set to $\sigma_a = 0.2$ m/s^2. The measurement likelihood is modeled as a Gaussian distribution with standard deviation $\sigma_n = 2$ m. Systematic resampling is triggered when the effective sample size falls below $N_p/2$.

• MLPNS. A machine learning-based probabilistic node selection (MLPNS) strategy in [23]. We note that the MLPNS is a supervised learning method that requires the ground truth locations of UE.

• GDOP. The geometric dilution of precision (GDOP) [34] baseline evaluates the candidate anchor subsets according to their geometric layout with respect to the current UE position estimate. It selects the subset with the smallest GDOP value, which corresponds to well-spread geometric constraints that are less likely to amplify small ranging errors into large position-estimation errors.

6.1.3 *Performance metrics*

We use the localization error to evaluate the performance of the proposed tracking and scheduling module. Specifically, the localization error at the t-th time interval is defined as $e_t = \sqrt{(x_t - x_t^{\text{pos}})^2 + (y_t - y_t^{\text{pos}})^2}$. From the empirical cumulative distribution function (CDF) of the error e_t, we report the mean absolute error (MAE), the median error (50% CDF), and the 90th-percentile error (90% CDF). We also compute the root-mean-square error (RMSE) as $E = \sqrt{\frac{1}{N_t}\sum_{t=1}^{N_t} e_t^2}$, where N_t denotes the total number of evaluation samples.

6.2 Evaluation of the proposed joint tracking and scheduling framework

We first evaluate the training and testing loss of the proposed joint tracking and scheduling framework. Specifically, we show the results in scenario 1, which represents an office with a residential floor plan of size 9.18 m × 12.06 m, with brick interior and exterior walls. A total of 126410 measurements were recorded at 85 distinct UE positions, and we use 80% data samples for training and the remaining 20% for testing. Figure 3 illustrates the training, validation, and testing loss of the proposed indoor localization system. During training, the system converges at 100 epochs and achieves an MAE of around 0.45 m.

Then, we evaluate the performance of the proposed joint tracking and scheduling framework using various performance metrics and compare the performance with different existing baselines. Specifically, we compare the tracking

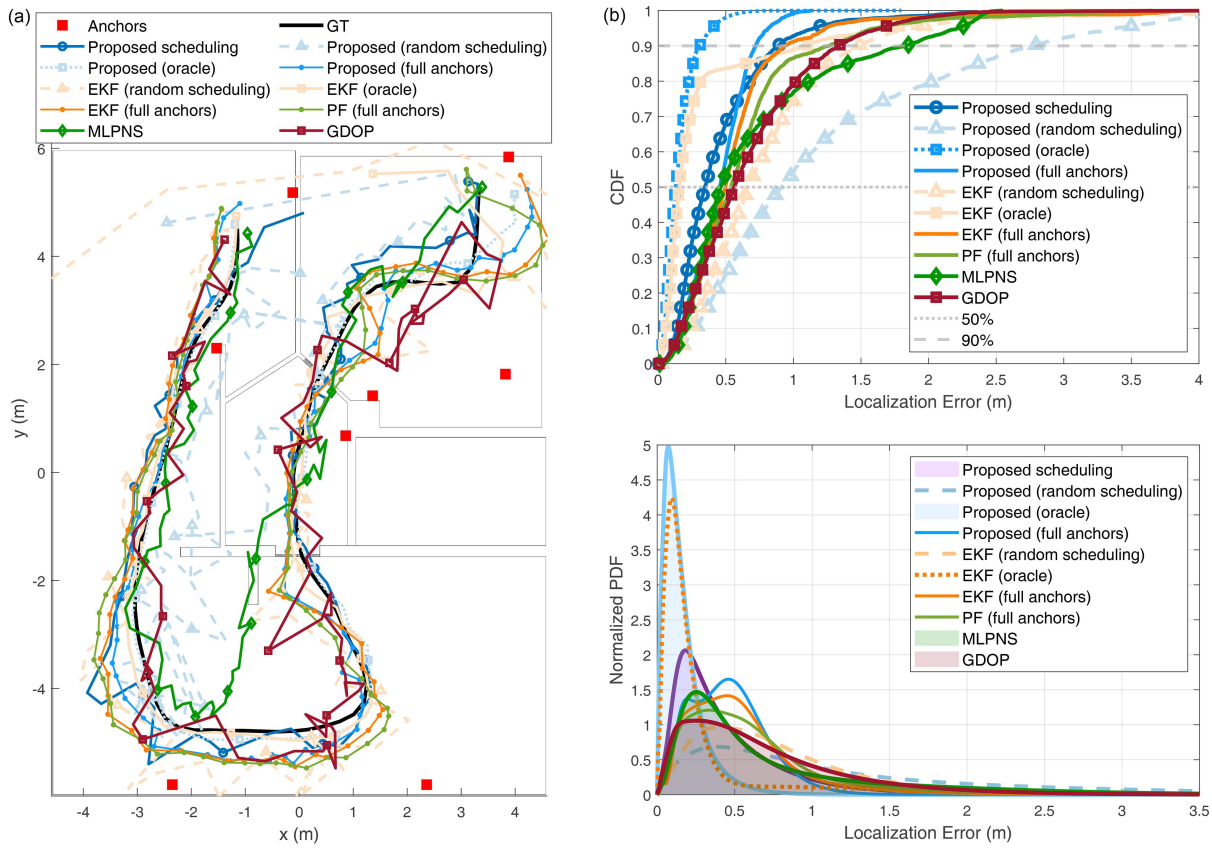

Figure 4 (Color online) Evaluation of the proposed scheduling strategy and baselines. (a) Trajectories; (b) CDF and normalized PDF.

performance of the UE across the different scheduling strategies, including random scheduling, oracle anchor selection, and all anchors of the proposed and existing methods, respectively. Figure 4(a) illustrates the trajectories of the UE predicted by the proposed scheduling strategy and other baselines in scenario 1 (office). We observe that the proposed scheduling strategy produces a trajectory that remains closer to the ground truth (GT) for most of the path, particularly around turns, whereas the EKF, MLPNS, and GDOP exhibit larger deviations and noticeable local overshoots.

To further evaluate the proposed scheduling strategy, the CDF and PDF of localization errors of the proposed and existing scheduling methods are illustrated in Figure 4(b). With our algorithm, the median of localization errors is 0.344 m, and the localization errors are lower than 0.885 m with a probability of 90%. For the other baselines, their medians are larger than 0.45 m, and the 90-th percentile CDFs are larger than 1 m.

We summarize the MAE, RMSE, median, and 90-th percentile CDF obtained in the office of the proposed framework and baselines in Figure 5. From the figure, the proposed scheduling outperforms the existing baselines, including the EKF, PF, MLPNS, and GDOP. For example, the localization MAE and RMSE of the proposed scheduling strategy achieve 0.458 and 0.596 m, respectively, compared to MLPNS and GDOP, which achieve 0.72 and 0.675 m MAE and 0.948 and 0.834 m RMSE, respectively.

We observe that both the proposed and EKF with random scheduling achieve lower localization accuracy. For example, the MAE of the proposed framework achieves 1.239 m, and the EKF with the random scheduling achieves 1.287 m, which are much higher than the methods with scheduling strategy. This observation implies that random scheduling tends to activate anchors/paths with large ranging errors more frequently, thereby introducing unreliable measurements into the system and ultimately degrading the overall localization accuracy.

In addition, compared with the all-anchor setting, the proposed scheduling strategy incurs only a slight loss in accuracy when activating a limited number of anchors. For instance, the MAE increases from 0.452 m with all anchors to 0.458 m with the proposed scheduling strategy. This result indicates that the proposed scheduling strategy can reduce anchor activations while maintaining comparable localization accuracy.

We also evaluate the performance of the proposed system and the baselines across another three scenarios. The corresponding trajectories are illustrated in Figure 6. The corresponding localization accuracy across different performance metrics of the three scenarios is summarized in Table 2, which includes the mean and variance values

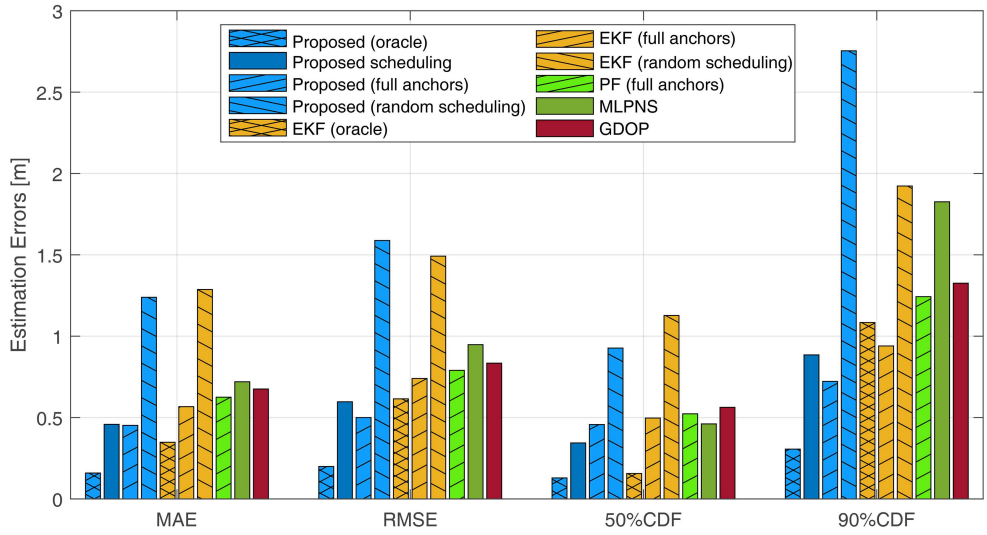

Figure 5 (Color online) Estimation errors of the proposed framework and baselines.

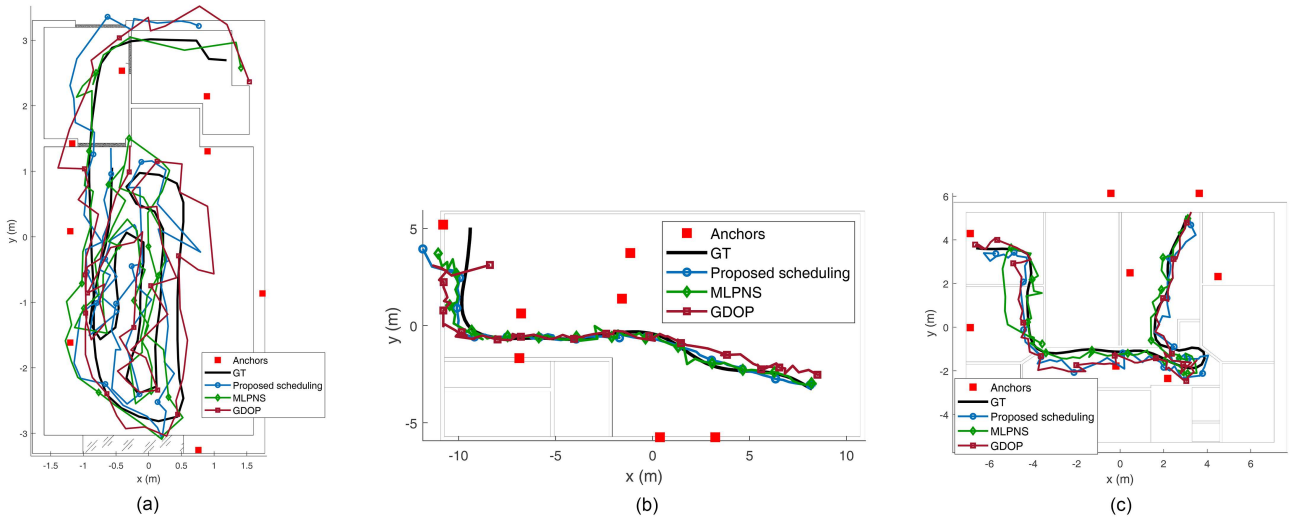

Figure 6 (Color online) Trajectories of the proposed system and baselines in three different scenarios. (a) Scenario 2; (b) scenario 3; (c) scenario 4.

of 100 repeated runs. We observe that the tracking accuracy of the proposed scheduling outperforms the GDOP baseline. For example, the MAE of the proposed scheduling is 0.224, 0.625, and 0.553 m in scenarios 2–4, respectively, compared to the GDOP baseline, which achieves 0.290, 1.125, and 0.617 m, respectively. The proposed self-supervised scheduling is comparable to, or even better than, that of the supervised MLPNS method. For example, in scenario 3, the RMSE of the proposed scheduling is around 0.72 m, while the MLPNS has a mean RMSE of around 0.938 m.

6.3 Resource efficiency analysis

To further quantify the practical resource saving, we conduct complementary device-level measurements using Google Pixel mobile devices as UEs, and Google Nest WiFi devices as WiFi RTT-enabled anchors [35,36]. Although the main tracking evaluation is based on UWB measurements, UWB and WiFi RTT rely on similar time-of-flight-based ranging principles. Anchor selection is therefore expected to yield a similar qualitative resource-saving trend under both protocols, while the specific resource savings may vary depending on the underlying hardware and protocol implementation. The WiFi experiments are conducted to provide complementary device-level evidence of this general resource-saving behavior under controllable settings. The resource saving is evaluated from two perspectives, i.e., the communication latency and the anchor capacities. We note that these measurements are not used to replace the UWB tracking evaluation, but to provide direct evidence of the ranging-request overhead and

Table 2 Estimation errors of the proposed system and baselines in different scenarios. For repeated-run entries, the first line denotes the mean and the second line denotes the variance.

	Scenario 2				Scenario 3				Scenario 4			
	MAE	RMSE	50%	90%	MAE	RMSE	50%	90%	MAE	RMSE	50%	90%
Proposed (full anchors)	0.198	0.223	0.187	0.340	0.729	0.889	0.634	1.294	0.502	0.549	0.492	0.797
	Var.: 0.006	Var.: 0.006	Var.: 0.006	Var.: 0.010	Var.: 0.156	Var.: 0.184	Var.: 0.151	Var.: 0.263	Var.: 0.013	Var.: 0.014	Var.: 0.018	Var.: 0.029
Proposed scheduling	0.224	0.266	0.191	0.411	0.625	0.720	0.529	1.031	0.553	0.656	0.488	1.045
	Var.: 0.035	Var.: 0.036	Var.: 0.037	Var.: 0.047	Var.: 0.170	Var.: 0.340	Var.: 0.135	Var.: 0.117	Var.: 0.051	Var.: 0.068	Var.: 0.039	Var.: 0.142
Proposed (random scheduling)	0.362	0.431	0.314	0.672	1.624	2.447	1.132	3.338	0.801	1.021	0.669	1.420
	Var.: 0.049	Var.: 0.056	Var.: 0.048	Var.: 0.087	Var.: 0.318	Var.: 0.581	Var.: 0.279	Var.: 1.059	Var.: 0.131	Var.: 0.199	Var.: 0.095	Var.: 0.249
Proposed (oracle)	0.074	0.105	0.049	0.169	0.268	0.499	0.214	0.633	0.117	0.152	0.091	0.238
	Var.: 0.018	Var.: 0.030	Var.: 0.008	Var.: 0.059	Var.: 0.079	Var.: 0.035	Var.: 0.039	Var.: 0.025	Var.: 0.034	Var.: 0.041	Var.: 0.030	Var.: 0.068
EKF (full anchors)	0.253	0.307	0.225	0.457	1.281	2.669	0.820	1.669	0.678	1.094	0.522	1.016
EKF (random scheduling)	0.419	0.516	0.358	0.768	1.647	3.333	0.925	2.525	0.928	1.735	0.602	1.517
	Var.: 0.007	Var.: 0.010	Var.: 0.006	Var.: 0.025	Var.: 0.043	Var.: 0.117	Var.: 0.017	Var.: 0.039	Var.: 0.038	Var.: 0.148	Var.: 0.008	Var.: 0.053
EKF (oracle)	0.206	0.328	0.150	0.356	0.521	1.600	0.114	0.882	0.472	1.163	0.210	0.636
PF (full anchors)	0.330	0.387	0.294	0.577	1.894	3.789	0.808	4.682	1.053	1.860	0.583	2.472
	Var.: 0.002	Var.: 0.002	Var.: 0.003	Var.: 0.006	Var.: 0.033	Var.: 0.066	Var.: 0.007	Var.: 0.233	Var.: 0.020	Var.: 0.039	Var.: 0.005	Var.: 0.160
MLPNS	0.292	0.437	0.203	0.562	0.757	0.938	0.434	1.385	0.411	0.552	0.321	0.787
	Var.: 0.117	Var.: 0.307	Var.: 0.045	Var.: 0.206	Var.: 0.183	Var.: 0.276	Var.: 0.176	Var.: 0.425	Var.: 0.174	Var.: 0.156	Var.: 0.035	Var.: 0.146
GDOP	0.290	0.364	0.238	0.550	1.125	2.247	0.864	2.066	0.617	0.711	0.547	1.097
	Var.: 0.044	Var.: 0.054	Var.: 0.038	Var.: 0.079	Var.: 0.189	Var.: 0.350	Var.: 0.111	Var.: 0.495	Var.: 0.059	Var.: 0.061	Var.: 0.067	Var.: 0.082

Table 3 The communication latency under different numbers of FTM requests using Google Pixel UEs and Google Nest WiFi anchors.

Number of requested anchors	3	4	5	6	7	8
Latency (ms)	196.40	251.47	314.07	379.62	435.61	449.93

Table 4 The ranging performance of one Google Nest WiFi anchor under different numbers of UEs.

Number of UEs	LoS			NLoS		
	Mean	Standard deviation	Connection ratio (%)	Mean	Standard deviation	Connection ratio (%)
1	0.12	0.27	99.81	2.89	1.27	95.07
2	0.29	0.28	99.52	2.45	1.67	93.07
3	0.25	0.20	99.28	2.84	1.03	92.43
4	0.27	0.18	99.72	3.62	1.11	84.40
5	0.18	0.31	97.59	3.90	1.73	63.77

anchor-side response capacity under controllable device settings.

Communication latency under different numbers of requested anchors. We evaluate the communication latency when the mobile device sends FTM ranging requests to different numbers of anchors. The latency is defined as the time from sending the FTM ranging requests to receiving the feedback from the corresponding anchors. As shown in Table 3, requesting more anchors leads to a higher communication latency. For example, requesting all 8 anchors requires 449.93 ms, whereas requesting 3 anchors requires only 196.40 ms. Therefore, compared with the all-anchor setting, selecting 3 anchors reduces the FTM communication latency by approximately 56.35%. This saving is important for integrated sensing and communication scenarios, because the time saved from ranging can be used for communication tasks within the same time window.

Connection reliability under different anchor capacities. We evaluate the ranging performance and connection ratio when different numbers of UEs repeatedly send FTM requests to the same anchor under LoS and NLoS conditions. Specifically, the connection ratio is defined as the number of FTM responses that have valid ranging information divided by the total number of FTM requests within a half-hour measurement period. The mean and standard deviation of the ranging error are defined as the absolute difference between the measured anchor-UE ranging distance and the ground-truth anchor-UE distance in LoS and NLoS conditions, respectively.

As shown in Table 4, when the number of UEs is no larger than 4, the connection ratio remains relatively stable. However, when the number of UEs increases to 5, the connection ratio under NLoS conditions drops from 84.40% to 63.77%. This result shows that the connection reliability of an anchor improves when fewer UEs send FTM requests to it simultaneously, especially under challenging NLoS conditions. Since the proposed framework achieves comparable tracking performance to the all-anchor setting with fewer ranging requests, it can reduce anchor-side congestion, improve ranging reliability, and allow the same anchor infrastructure to serve more UEs. These results demonstrate that the proposed framework improves resource efficiency from both the UE-side latency perspective and the anchor-side service-capacity perspective.

6.4 On-device computational overhead

We further analyze the on-device computational overhead of the proposed framework in terms of model complexity, inference latency, and energy consumption. Specifically, the total number of trainable parameters of the proposed DSSM and RL-based anchor scheduling module is 67.69 K, and the computational cost for one inference is 544.66 K FLOPs. This indicates that the proposed model has a lightweight computational footprint during online deployment.

To evaluate the practical inference latency, we deploy the proposed framework on different commodity mobile devices, including Google Pixel 5, Pixel 6a, Pixel 6 Pro, Pixel 7 Pro, and Pixel tablet. The hardware specifications and measured computation time are summarized in Table 5. The proposed framework costs less than 200 ms of inference on all tested devices. For example, the inference latency is 143.49 ms on Google Pixel 7 Pro, and 172.39 ms on Google Pixel Tablet. These results show that the proposed framework can support real-time localization updates on commodity UEs.

We further evaluate the power consumption of the proposed framework. Specifically, we deploy the proposed framework on different types of UEs and measure the battery power during a fixed time. For example, we first charge the device to 80%, fix the screen brightness, and continuously send FTM requests and run the proposed framework for one hour. Then, we charge the same device back to 80% and keep the same screen brightness for one hour without sending FTM requests or running the proposed framework. The extra battery consumption is calculated by comparing these two cases. As shown in Table 5, the energy consumption of the proposed framework ranges

Table 5 The hardware specifications of different types of phones and the corresponding computation time and energy consumption.

	Google Pixel 5	Google Pixel 6a	Google Pixel 6 Pro	Google Pixel 7 Pro	Google Pixel tablet
Processor	Snapdragon 765G (7 nm)	Google tensor (5 nm)	Google tensor (5 nm)	Google tensor G2 (5 nm)	Google tensor G2 (5 nm)
Memory	8 GB LPDDR4	6 GB LPDDR5	12 GB LPDDR5	12 GB LPDDR5	8 GB LPDDR5
Proposed framework	164.65 ms	152.20 ms	145.53 ms	143.49 ms	172.39 ms
Energy consumption per hour	244.8 mAh	220.5 mAh	200.12 mAh	150 mAh	210.6 mAh

from 150 to 244.8 mAh per hour across the tested devices. These results demonstrate that the proposed framework introduces acceptable on-device computational and energy overhead, and is feasible for real-world deployment on resource-constrained UEs.

7 Conclusion

This paper studied a resource-efficient indoor tracking system for the large-scale deployment of anchor-based networks, where bandwidth and energy are constrained, preventing the activation of all anchors at each time step. Specifically, we proposed a self-supervised joint tracking and anchor scheduling framework that integrates a physics-guided deep state space model with a GRU-based sequence module to capture target dynamics and uncertainty directly from range measurements without requiring ground truth trajectories. To reduce sensing and communication overhead, we further developed an actor-critic reinforcement learning scheduler that selects a resource-budgeted subset of informative anchors for future ranging. The scheduler is guided by a self-supervised reward derived from the reconstruction likelihood of the learned tracking module, thereby encouraging actions that are most consistent with the learned dynamics and measurement statistics. Experiments on a real-world UWB dataset demonstrated that the proposed approach achieves robust tracking performance while significantly reducing anchor activations compared with classical filtering and supervised learning-based baselines, validating the effectiveness of jointly learning latent dynamics and proactive anchor selection. Nevertheless, under severe or persistent NLoS conditions affecting most available anchors, the number of reliable ranging links may become insufficient. In this case, the scheduler may be unable to identify an informative and reliable anchor subset, leading to degraded tracking performance. Future work will investigate online adaptation and explicit NLoS detection to further improve robustness under such challenging conditions.

References

1 Yang Y, Chen M, Blankenship Y, et al. Positioning using wireless networks: applications, recent progress, and future challenges. IEEE J Sel Areas Commun, 2024, 42: 2149–2178
2 3GPP. Study on artificial intelligence (AI)/machine learning (ML) for NR air interface (Release 18). TR 38.843 V18.0.0., 2023
3 Farahsari P S, Farahzadi A, Rezazadeh J, et al. A survey on indoor positioning systems for IoT-based applications. IEEE Int Things J, 2022, 9: 7680–7699
4 Zhu X, Qu W, Qiu T, et al. Indoor intelligent fingerprint-based localization: principles, approaches and challenges. IEEE Commun Surv Tut, 2020, 22: 2634–2657
5 Kwon H, Hegde C, Kiarashi Y, et al. A feasibility study on indoor localization and multiperson tracking using sparsely distributed camera network with edge computing. J Ind Sea Pos Nav, 2023, 1: 187–198
6 Zeng Q, Tao X, Yu H, et al. An indoor 2-D LiDAR SLAM and localization method based on artificial landmark assistance. IEEE Sens J, 2023, 24: 3681–3692
7 Bai S, Wen W, Shi C. 3DIO: low-drift 3-D deep-inertial odometry for indoor localization using an IMU. IEEE Int Things J, 2024, 12: 12711–12722
8 Ali R, Liu R, Nayyar A, et al. Tightly coupling fusion of UWB ranging and IMU pedestrian dead reckoning for indoor localization. IEEE Access, 2021, 9: 164206
9 Yu H, She C, Hu Y, et al. Floor-plan-aided indoor localization: zero-shot learning framework, data sets, and prototype. IEEE J Sel Areas Commun, 2024, 42: 2472–2486
10 Wang G, Cheng P, Li S, et al. Self-supervised deep state space model for enhanced indoor tracking. In: Proceedings of IEEE International Conference on Communications (ICC), 2025. 692–697
11 Zhuang Y, Zhang C, Huai J, et al. Bluetooth localization technology: principles, applications, and future trends. IEEE Int Things J, 2022, 9: 23506–23524
12 Hu C, Huang P, Wang W. Tightly coupled visual-inertial-UWB indoor localization system with multiple position-unknown anchors. IEEE Robot Autom Lett, 2023, 9: 351–358
13 Ma C, Wu B, Poslad S, et al. Wi-Fi RTT ranging performance characterization and positioning system design. IEEE Trans Mobile Comput, 2020, 21: 740–756
14 Che F, Ahmed Q Z, Fontaine J, et al. Feature-based generalized Gaussian distribution method for NLoS detection in ultra-wideband (UWB) indoor positioning system. IEEE Sens J, 2022, 22: 18726–18739
15 Pan W, Yang Y, Chen M, et al. Fusing Bluetooth with pedestrian dead reckoning: a floor plan-assisted positioning approach. 2025. ArXiv:2504.09905
16 Wang G, Li S, Cheng P, et al. ToF-based NLoS indoor tracking with adaptive ranging error mitigation. IEEE Trans Signal Process, 2024, 72: 4855–4870
17 Yu H, She C, Hu Y, et al. Meta-learning based indoor localization: a hierarchical framework. IEEE Trans Veh Technol, 2024, 74: 6697–6702

18 Albaidhani A, Morell A, Vicario J L. Anchor selection for UWB indoor positioning. Trans Emerging Tel Tech, 2019, 30: e3598
19 Zhao Y, Li Z, Hao B, et al. Sensor selection for TDOA-based localization in wireless sensor networks with non-line-of-sight condition. IEEE Trans Veh Technol, 2019, 68: 9935–9950
20 Fan C, Li L, Zhao M M, et al. Communication and energy-constrained neighbor selection for distributed cooperative localization. IEEE Trans Wireless Commun, 2022, 22: 4158–4172
21 Xie J, Wang W, Liu X, et al. Identification of NLOS based on soft decision method. IEEE Wireless Commun Lett, 2023, 12: 703–707
22 Roy P, Chowdhury C. A survey of machine learning techniques for indoor localization and navigation systems. J Intell Robot Syst, 2021, 101: 63
23 Gómez-Vega C A, Win M Z, Conti A. Machine learning based node selection for UWB network localization. In: Proceedings of IEEE Military Communications Conference (MILCOM), 2023. 735–740
24 Hajiakhondi-Meybodi Z, Mohammadi A, Hou M, et al. DQLEL: deep Q-learning for energy-optimized LoS/NLoS UWB node selection. IEEE Trans Signal Process, 2022, 70: 2532–2547
25 Kim D H, Park J, Ko Y B, et al. Deep Q-network based UWB anchor and strategy selection for accurate pedestrian localization in vehicular environments. In: Proceedings of IEEE International Conference on Machine Learning for Communication and Networking (ICMLCN), 2025. 1–6
26 Girin L, Leglaive S, Bie X, et al. Dynamical variational autoencoders: a comprehensive review. Found Trends(r) Machine Learn, 2022, 15: 1–175
27 Wang G, Gu Z, Li S, et al. LocDreamer: world model-based learning for joint indoor tracking and anchor scheduling. In: Proceedings of AAAI Workshop on Machine Learning for Wireless Communication and Networks (ML4Wireless), 2026
28 Gedon D, Wahlström N, Schön T B, et al. Deep state space models for nonlinear system identification. IFAC-PapersOnLine, 2021, 54: 481–486
29 Rong Li X, Jilkov V P. Survey of maneuvering targettracking. Part I: dynamic models. IEEE Trans Aerosp Electron Syst, 2003, 39: 1333–1364
30 Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need. In: Proceedings of Advances in Neural Information Processing Systems, 2017. 5998–6008
31 Wang G, Cheng P, Li S, et al. Label-free range-based indoor tracking with physics-guided deep state space model. IEEE Trans Signal Process, 2026, 74: 1174–1189
32 Bregar K. Indoor UWB positioning and position tracking data set. Sci Data, 2023, 10: 744
33 Diederik P K, Jimmy B. Adam: a method for stochastic optimization. 2014. arXiv:1412.6980
34 Albaidhani A, Alsudani A. Anchor selection by geometric dilution of precision for an indoor positioning system using ultra-wide band technology. IET Wireless Sens Syst, 2021, 11: 22–31
35 IEEE. IEEE Standard for Information Technology–Telecommunications and Information Exchange between Systems Local and Metropolitan Area Networks–Specific Requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Ame. IEEE Std 802.11-2016 (Revision of IEEE Std 802.11-2012), 2016. 1–3534
36 Android Developers. Wi-Fi location: ranging with RTT. 2026. https://developer.android.com/develop/connectivity/wifi/wifi-rtt