MITSUBISHI ELECTRIC RESEARCH LABORATORIES
https://www.merl.com

Multi-Band Wi-Fi Neural Dynamic Fusion

Kato, Sorachi; Wang, Pu; Koike-Akino, Toshiaki; Fujihashi, Takuya; Mansour, Hassan; Boufounos, Petros T.

TR2025-115 August 02, 2025

Abstract

Wi-Fi channel measurements across different bands, e.g., sub-7-GHz and 60-GHz bands, are asynchronous due to the uncoordinated nature of distinct standards protocols, e.g., 802.11ac/ax/be and 802.11ad/ay. Multi-band Wi-Fi fusion has been considered before on a frame-to-frame basis for simple classification tasks, which does not require fine-time-scale alignment. In contrast, this paper considers asynchronous sequence- to-sequence fusion between sub-7-GHz channel state information (CSI) and 60-GHz beam signal-to-noise-ratio (SNR)s for more challenging tasks, such as continuous coordinate estimation. To handle the timing disparity between asynchronous multi-band Wi-Fi channel measurements, this paper proposes a multi-band neural dynamic fusion (NDF) framework. This framework uses separate encoders to embed the multi-band Wi-Fi measurement sequences to separate initial latent conditions. Using a continuous-time ordinary differential equation (ODE) modeling, these initial latent conditions are propagated to the respective latent states of the multi-band channel measurements at the same time instances for a latent alignment and a post-ODE fusion, and at their original time instances for measurement reconstruction. We derive a customized loss function based on the variational evidence lower bound (ELBO) that balances between the multi-band measurement reconstruction and continuous co- ordinate estimation. We evaluate the NDF framework using an in-house multi-band Wi-Fi testbed and demonstrate substantial performance improvements over a comprehensive list of single- band and multi-band baseline methods.

IEEE Transactions on Wireless Communications 2025

Mitsubishi Electric Research Laboratories, Inc.
201 Broadway, Cambridge, Massachusetts 02139

Multi-Band Wi-Fi Neural Dynamic Fusion

Sorachi Kato, Pu (Perry) Wang, Toshiaki Koike-Akino, Takuya Fujihashi, Hassan Mansour, Petros Boufounos

***Abstract*—Wi-Fi channel measurements across different bands, e.g., sub-7-GHz and 60-GHz bands, are asynchronous due to the uncoordinated nature of distinct standards protocols, e.g., 802.11ac/ax/be and 802.11ad/ay. Multi-band Wi-Fi fusion has been considered before on a *frame-to-frame* basis for simple classification tasks, which does not require fine-time-scale alignment. In contrast, this paper considers asynchronous *sequence-to-sequence* fusion between sub-7-GHz channel state information (CSI) and 60-GHz beam signal-to-noise-ratio (SNR)s for more challenging tasks, such as continuous coordinate estimation. To handle the timing disparity between asynchronous multi-band Wi-Fi channel measurements, this paper proposes a multi-band neural dynamic fusion (NDF) framework. This framework uses separate encoders to embed the multi-band Wi-Fi measurement sequences to separate initial latent conditions. Using a continuous-time ordinary differential equation (ODE) modeling, these initial latent conditions are propagated to the respective latent states of the multi-band channel measurements at the same time instances for a latent alignment and a post-ODE fusion, and at their original time instances for measurement reconstruction. We derive a customized loss function based on the variational evidence lower bound (ELBO) that balances between the multi-band measurement reconstruction and continuous coordinate estimation. We evaluate the NDF framework using an in-house multi-band Wi-Fi testbed and demonstrate substantial performance improvements over a comprehensive list of single-band and multi-band baseline methods.**

***Index Terms*—WLAN sensing, 802.11bf, Wi-Fi sensing, ISAC, localization, multi-band fusion, and dynamic learning.**

I. Introduction

Wi-FI sensing, e.g., device localization and device-free human sensing, has received a great deal of attention in the past decade from both academia and industry. This trend has been manifested by the establishment of 802.11bf WLAN Sensing task group in 2020 to go beyond data transmission and meet industry demands for wireless sensing [2]–[5].

Existing Wi-Fi sensing is mainly based on coarse-grained received signal strength indicator (RSSI) and fine-grained channel state information (CSI) at sub-7-GHz bands [6]–[11]. At a high frame rate, CSI reflects intrinsic channel statistics in the form of channel frequency responses (CFR) over subcarrier frequencies (delay) and multiple transmitter-receiver antenna pairs (angle). At the same time, it may experience channel instability due to even small-scale environmental changes. On the other hand, mid-grained mmWave beam training measurements at 60 GHz, e.g., beam SNR, have shown better channel stability over time [12]–[17]. These beam SNR measurements originate from sector-level directional beam training, a mandatory step for mmWave Wi-Fi to compensate for large path loss and establish the link between access points (APs) and users. However, they suffer from low frame rates and irregular sample intervals due to the beam training overhead and follow-up association steps.

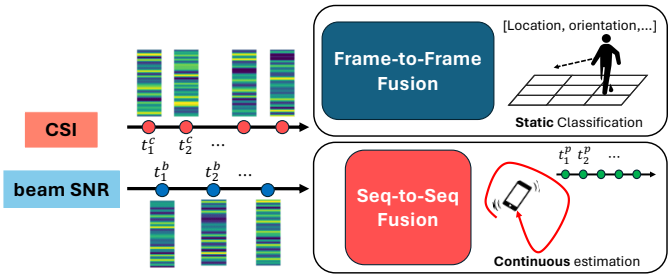


Fig. 1: Multi-band Wi-Fi fusion from *frame-to-frame* basis of [14] for classification (top) to asynchronous *sequence-to-sequence* basis for continuous-time regression (bottom).

Fusion-based approaches have been considered in the literature for robustness and higher accuracy. Heterogeneous sensor fusion was studied between Wi-Fi and other modalities, e.g., Wi-Fi and vision [18], Wi-Fi and ultra-wideband (UWB) [19]. Within Wi-Fi channel measurements, CSI and RSSI can be simply concatenated for joint feature extraction [20]. It is also possible to fuse the phase and amplitude of the fine-grained CSI for localization [21]. For multi-band Wi-Fi fusion, CSI splicing is a well-researched channel fusion technique that merges the phase waveforms of multiple channel frequencies. This fusion process compensates for hardware-induced distortion, enabling smooth connection between adjacent Wi-Fi channels and reproducing CSI measurements with significantly wider bandwidth, so it improves the resolution of angle-of-arrival (AoA)/time-of-flight (ToF) estimation [22]–[25]. However, these methods typically presume that narrowband CSIs are acquired from multiple channels of the same wireless standard that guarantee the same CSI format and are even guaranteed to be well-synchronized in time due to the rapid channel hopping. As a more challenging problem, we consider the integration of sub-7-GHz and mmWave channel measurements, in which the channel measurements for both bands are asynchronous in time and can have distinct modalities. There are few papers addressing the problem, and most proposals suggest that sub-7-GHz CSI is primarily used to support mmWave sensing by mitigating the beam training overhead or by selecting optimal mmWave APs whose coverage guarantees that they encompass the target device, which

Part of this paper was presented in ICASSP 2024 [1].

The work of S. Kato was done during his visit and internship at MERL. He was also supported by Japan Society for the Promotion of Science (JSPS) KAKENHI under Grant 23KJ1499.

PW, TK, HM, and PB are with Mitsubishi Electric Research Laboratories (MERL), Cambridge, MA 02139, USA.

SK and TF are with the Graduate School of Information Science and Technology, Osaka University, Suita, Osaka, Japan.

*Corresponding author: pwang@merl.com

is achieved by providing coarse-grained localization utilizing CSI [15], [26]. Our previous work in [14] considered simple classification tasks, e.g. pose classification (over 8 stationary poses), seat occupancy detection (8 stationary patterns), and fixed-grid localization. Despite being sampled at different time instances, both channel measurements can be simply combined on a *frame-to-frame* basis as these asynchronous samples correspond to the same stationary label (e.g. pose, occupancy, location) and their respective sampling time becomes irrelevant for the fusion task; see the top plot of Fig. 1 and notice the asynchronous time instances t_n^c and t_n^b, $n = 1, 2, \ldots$. [27] considers sequence-to-sequence multi-band Wi-Fi fusion by using fine-grained CSI (channel frequency responses - CFRs) from both sub-7-GHz and 60 GHz with the help of reflecting intelligent surface. Assuming similar frame rates of both CSIs, the attention mechanism and transformer architecture have been used to fuse these two Wi-Fi channel measurements using simulated datasets.

For more challenging tasks of continuous-time object trajectory estimation using asynchronous multi-band Wi-Fi measurements, several challenges need to be addressed. First, asynchronous channel measurements at different bands need to be aligned to estimate object trajectory. As illustrated in Fig. 1 (the bottom plot), there exists time disparity between CSI measurements at t_n^c and beam SNR measurements at t_n^b. In addition, there may exist time disparity between the input measurements at either t_n^c or t_n^b, and the desired trajectory estimates at t_n^p. Second, mmWave beam SNRs are sampled at a much lower frame rate than the CSI measurements. In an ideal scenario, beam SNRs can be obtained at a frame rate of 10 Hz for a typical beacon interval of 100 ms. However, multiple users need to contend the channel time for (uplink) beam training during each beacon interval, resulting in a lower frame rate. Third, contention-based channel access further results in irregularly sampled beam SNR measurements at AP for a given user.

To address the aforementioned challenges, we propose a multi-band Wi-Fi neural dynamic fusion (NDF) framework. This framework evolves from the stationary frame-to-frame basis in [14] (illustrated in the upper plot of Fig. 1) to a dynamic asynchronous *sequence-to-sequence* basis (the bottom plot of Fig. 1), thus supporting more challenging downstream tasks, e.g., regression in a continuous space and continuous-time object trajectory estimation. The proposed multi-band NDF substantially extends our previous work on a beam SNR-only framework of [16] and [17] to a neural network architecture comprising multiple encoders, latent dynamic learning modules, a post-ODE fusion module, and multiple decoders, as depicted in Fig. 2. Our main contributions are summarized below:

1) To the best of our knowledge, this is the first effort to address multi-band asynchronous fusion between sub-7-GHz CSI and 60-GHz beam SNR for trajectory estimation of moving objects.
2) We present a multi-encoder, multi-decoder NDF network in Fig. 2. It utilizes the two encoders that act as an initial latent condition estimator for the two distinct input sequences, employs an ordinary differential equation (ODE) modeling [28], [29] for latent dynamic learning and latent state alignment, and fuses these aligned latent states via the post-ODE fusion module.
3) We consider multiple fusion schemes such as multilayer perceptron (MLP) fusion, pairwise interaction fusion, and weighted importance fusion for the post-ODE fusion module.
4) We derive a loss function building upon the variational evidence lower bound (ELBO) between prior and approximate posterior distributions of the initial latent conditions as well as the likelihood of multiple decoder outputs. This ELBO-based loss function incorporates both unsupervised multi-band reconstruction loss and supervised coordinate estimation loss.
5) We build an automated data collection platform using commercial-of-the-shelf 802.11ac/ad-compliant Wi-Fi routers and a TurtleBot as a mobile user. This platform continuously gathers CSI at 5 GHz and beam SNR at 60 GHz from the TurtleBot, while simultaneously recording its ground truth positions.
6) We conduct a comprehensive ablation study on trajectory estimation performance, generalization capability, and interpretation using real-world experimental data.

The remainder of this paper is organized as follows. Section II introduces the problem formulation, followed by a brief review of existing multi-band Wi-Fi fusion solutions. Section III details the proposed multi-band NDF framework, with subsections dedicated to each module and the derivation of the loss function. Section V describes our in-house multi-band Wi-Fi data collection testbed and performance evaluation, followed by the conclusion in Section VI.

II. Problem Formulation and Existing Solutions

A. Problem Formulation

We formulate the trajectory estimation as a continuous regression problem with asynchronous CSI and beam SNR sequences. As illustrated in Fig. 2, in each time instance t_n^b, we collect a set of M_b beam SNR values $\mathbf{b}_n = [b_{n,1}, b_{n,2}, \cdots, b_{n,M_b}]^\top \in \mathbb{R}^{M_b \times 1}$, each corresponding to a beam training pattern. At time instance t_n^c, we collect a CSI measurement $\mathbf{C}_n \in \mathbb{C}^{N_{\mathrm{Tx}} \times N_{\mathrm{Rx}} \times N_{\mathrm{s}}}$ with the (i, j, k)-th element $C_n(i, j, k)$ given by the CFR from the transmitting antenna i, the receiving antenna j and the subcarrier k. For a time window size or sequence length ΔT_w, we group the N_b beam SNRs and the N_c CSI measurements as two input sequences. The problem of interest is to estimate the object trajectory $\mathbf{p}_n$ at N_p desired time instances t_n^p within the time window ΔT_w,

$$\{\mathbf{b}_n, t_n^b\}_{n=0}^{N_b}, \quad \{\mathbf{C}_n, t_n^c\}_{n=0}^{N_c} \rightarrow \{\mathbf{p}_n, t_n^p\}_{n=0}^{N_p}, \tag{1}$$

where $\mathbf{p}_n = [x_n, y_n]^\top$ consists of two-dimensional coordinates at t_n^p.

We follow standard practices to calibrate raw CSI measurements $\mathbf{C}_n$ due to the lack of synchronization between the Wi-Fi transmitter and the receiver [30]–[35]. Specifically, we use SpotFi [30] to remove linear phase offsets caused by

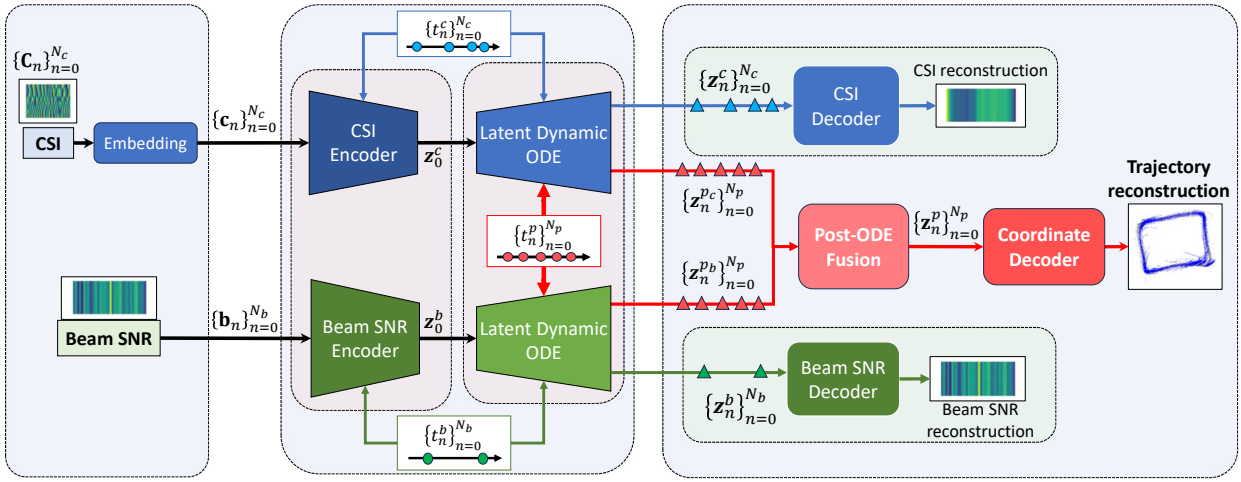


Fig. 2: The network architecture of the multi-band Wi-Fi neural dynamic fusion (NDF) for continuous-time object trajectory estimation. It comprises multiple encoders, latent dynamic learning modules, a post-ODE fusion module, and multiple decoders.

sampling time offset (STO) and apply antenna-wise conjugate multiplication [33] to minimize packet-to-packet phase fluctuation. Once the CSI measurements are calibrated, we employ a pretrained convolutional autoencoder (CAE) to compress each calibrated CSI measurement into a CSI embedding vector $\mathbf{c}_n \in \mathbb{R}^{M_c \times 1}$, where M_c is the dimension of the CSI embedding. More details on CSI calibration and embedding can be found in Appendix A.

As illustrated in Fig. 2, the equivalent input sequences become $\mathbf{c}_n$ and $\mathbf{b}_n$, and the problem of interest reduces to

$$\{\mathbf{b}_n, t_n^b\}_{n=0}^{N_b}, \quad \{\mathbf{c}_n, t_n^c\}_{n=0}^{N_c} \to \{\mathbf{p}_n, t_n^p\}_{n=0}^{N_p}. \tag{2}$$

B. Existing Solutions

1) Frame-to-Frame Fusion: Given the time disparity between t_n^c and t_n^b, a simple way to combine the two input sequences is to align the CSI and beam SNR sequences at the input level. This can be accomplished using either linear or nearest-neighbor interpolation.

For the linear interpolation (**LinearInt**) scheme, for a given output time instance t_n^p, we first identify the intervals i_c and i_b in the CSI and beam SNR sequences, respectively, such that $t_{i_c}^c \le t_n^p \le t_{i_c+1}^c$ and $t_{i_b}^b \le t_n^p \le t_{i_b+1}^b$ and then interpolate the input sequence at t_n^p using the two input measurements at both ends of the identified interval

$$\begin{aligned}
\mathbf{b}_n^{\text{Lin}} &= \mathbf{b}_{i_b} + \frac{t_n^p - t_{i_b}^b}{t_{i_b+1}^b - t_{i_b}^b}(\mathbf{b}_{i_b+1} - \mathbf{b}_{i_b}), \\
\mathbf{c}_n^{\text{Lin}} &= \mathbf{c}_{i_c} + \frac{t_n^p - t_{i_c}^c}{t_{i_c+1}^c - t_{i_c}^c}(\mathbf{c}_{i_c+1} - \mathbf{c}_{i_c}),
\end{aligned} \tag{3}$$

where $n = 0, \cdots, N_p$. On the other hand, the nearest-neighbor interpolation (**NearestInt**) finds the input element from the two input sequences at the time instance that is closest to the desired output time instance t_n^p

$$\begin{aligned}
\mathbf{b}_n^{\text{Nea}} &= \begin{cases} \mathbf{b}_{i_b}, & \text{if } t_n^p - t_{i_b}^b \le t_{i_b+1}^b - t_n^p \\ \mathbf{b}_{i_b+1}, & \text{otherwise,} \end{cases} \\
\mathbf{c}_n^{\text{Nea}} &= \begin{cases} \mathbf{c}_{i_c}, & \text{if } t_n^p - t_{i_c}^c \le t_{i_c+1}^c - t_n^p \\ \mathbf{c}_{i_c+1}, & \text{otherwise.} \end{cases}
\end{aligned} \tag{4}$$

Given the aligned input sequences

$$\{\mathbf{b}_n^{\text{Lin/Nea}}, t_n^p\}_{n=0}^{N_p}, \quad \{\mathbf{c}_n^{\text{Lin/Nea}}, t_n^p\}_{n=0}^{N_p}, \tag{5}$$

we can fuse them in a frame-to-frame fashion and regress the fused sequence to the trajectory coordinate

$$\hat{\mathbf{p}}_n = \mathcal{M}(\mathcal{F}(\mathbf{c}_n^{\text{Lin/Nea}}, \mathbf{b}_n^{\text{Lin/Nea}})), \tag{6}$$

where $\mathcal{F}$ represents a fusion scheme, e.g., concatenation or other considered options in Section III-C, and $\mathcal{M}$ denotes a multi-layer perceptron (MLP) network. One may also try other interpolation schemes such as the spline and piecewise polynomial interpolation.

2) Sequence-to-Sequence Fusion: As opposed to the frame-to-frame fusion, one can use a recurrent neural network (RNN) to capture recurrently updated hidden features from the entire input sequence [7], [36], [37]. By fusing the hidden states corresponding to the CSI and beam SNR sequences, one can achieve what we refer to as the sequence-to-sequence fusion.

For an input sequence $\{\mathbf{s}_n\}_{n=0}^{N}$ ($\mathbf{s}_n$ can be either the CSI $\mathbf{c}_n$ or beam SNR $\mathbf{b}_n$ sequence), a standard RNN unit updates its hidden state $\mathbf{h}_{n-1}$ at time t_{n-1} to $\mathbf{h}_n$ at time t_n with the input measurement $\mathbf{s}_n$ at time instance t_n as

$$\mathbf{h}_n = \mathcal{R}(\tilde{\mathbf{h}}_n, \mathbf{s}_n; \boldsymbol{\theta}), \quad \tilde{\mathbf{h}}_n = \mathbf{h}_{n-1}, \tag{7}$$

where $\mathcal{R}$ is an Long Short-Term Memory (LSTM) [38] or Gated Recurrent Unit (GRU) [39] unit, and $\tilde{\mathbf{h}}_n$ is an auxiliary vector. In the standard RNN, it assumes that the sampling intervals $\Delta t_n = t_n - t_{n-1}$ are uniform, i.e., $\Delta t_1 = \cdots =$

Δt_N. Consequently, the auxiliary vector is simply given by the previous hidden state $\tilde{\mathbf{h}}_n = \mathbf{h}_{n-1}$. Refer to Appendix B for details on the update of the standard LSTM unit.

To address irregularly sampled sequences where $\Delta t_n \neq \Delta t_{n+1}$, we consider the following sequence-to-sequence baseline methods. One such method is **RNN-Decay** [29], which decays the previous hidden state exponentially with respect to the time interval before being fed into the RNN unit,

$$\mathbf{h}_n = \mathcal{R}(\tilde{\mathbf{h}}_n, \mathbf{s}_n; \boldsymbol{\theta}), \quad \tilde{\mathbf{h}}_n = \mathbf{h}_{n-1} e^{-\Delta t_n}, \tag{8}$$

where the auxiliary vector $\tilde{\mathbf{h}}_n$ accounts for the irregular sampling intervals. Another method is **RNN-Δ** [29], which accounts for the irregular sampling interval by augmenting the input

$$\mathbf{h}_n = \mathcal{R}(\tilde{\mathbf{h}}_n, \tilde{\mathbf{s}}_n; \boldsymbol{\theta}), \quad \tilde{\mathbf{h}}_n = \mathbf{h}_{n-1}, \quad \tilde{\mathbf{s}}_n = [\mathbf{s}_n^\top, \Delta t_n]^\top, \tag{9}$$

while keeping the auxiliary vector as the previous hidden state.

For a desired output time instance t_n^p, we first identify its immediate preceding time instances i_c and i_b in the CSI and beam SNR sequences as $t_{i_c}^c \leq t_n^p \leq t_{i_c+1}^c$ and $t_{i_b}^b \leq t_n^p \leq t_{i_b+1}^b$ with $\mathbf{h}_{i_c}^c$ and $\mathbf{h}_{i_b}^b$ previously updated using either (8) or (9). Then we propagate $\mathbf{h}_{i_c}^c$ and $\mathbf{h}_{i_b}^b$ to the output time instance t_n^p as

$$\begin{aligned} \mathbf{h}_n^{p_c} &\triangleq \mathbf{h}_{t_n^p}^c = \mathcal{R}(\mathbf{h}_{i_c}^c e^{-(t_n^p - t_{i_c}^c)}, \mathbf{0}; \boldsymbol{\theta}), \\ \mathbf{h}_n^{p_b} &\triangleq \mathbf{h}_{t_n^p}^b = \mathcal{R}(\mathbf{h}_{i_b}^b e^{-(t_n^p - t_{i_b}^b)}, \mathbf{0}; \boldsymbol{\theta}), \end{aligned} \tag{10}$$

for the RNN-Decay update and

$$\begin{aligned} \mathbf{h}_n^{p_c} &\triangleq \mathbf{h}_{t_n^p}^c = \mathcal{R}(\mathbf{h}_{i_c}^c, [\mathbf{0}^\top, (t_n^p - t_{i_c}^c)]^\top; \boldsymbol{\theta}), \\ \mathbf{h}_n^{p_b} &\triangleq \mathbf{h}_{t_n^p}^b = \mathcal{R}(\mathbf{h}_{i_b}^b, [\mathbf{0}^\top, (t_n^p - t_{i_b}^b)]^\top; \boldsymbol{\theta}), \end{aligned} \tag{11}$$

for the RNN-Δ update, where $n = 1, \cdots, N_p$, by setting the current input $\mathbf{s}_n = \mathbf{0}$ at t_n^p. We can then fuse the aligned hidden states $\mathbf{h}_n^{p_c}$ and $\mathbf{h}_n^{p_b}$ at t_n^p and regress the trajectory coordinate

$$\hat{\mathbf{p}}_n = \mathcal{M}(\mathcal{F}(\mathbf{h}_n^{p_c}, \mathbf{h}_n^{p_b})), \tag{12}$$

where $\mathcal{F}$ represents a fusion scheme and $\mathcal{M}$ is an MLP.

III. Multi-Band Neural Dynamic Fusion

In the following, we provide a detailed module-by-module explanation of the multi-band NDF framework illustrated in Fig. 2. This framework utilizes separate encoders to sequentially map the CSI embedding $\mathbf{c}_n$ and beam SNR $\mathbf{b}_n$ sequences (along with their respective time stamps t_n^c and t_n^b) into the latent space and estimate initial latent conditions, i.e., $\mathbf{z}_0^c$ and $\mathbf{z}_0^b$. In the latent dynamic learning module, both initial latent conditions are propagated using a learnable ODE model [28], [29], generating virtual latent states (i.e., $\mathbf{z}_n^{p_c}$ and $\mathbf{z}_n^{p_b}$) at the same time instances t_n^p for alignment. These aligned latent states are then fused in a post-ODE fashion before being fed into a coordinate decoder for trajectory estimation. Meanwhile, one can also utilize the same learnable ODE model to regress latent states (i.e., $\mathbf{z}_n^c$ and $\mathbf{z}_n^b$) at the input time instances t_n^c and t_n^b for the CSI and beam SNR. These regressed latent states can be fed into either the CSI or beam SNR decoder for waveform reconstruction.

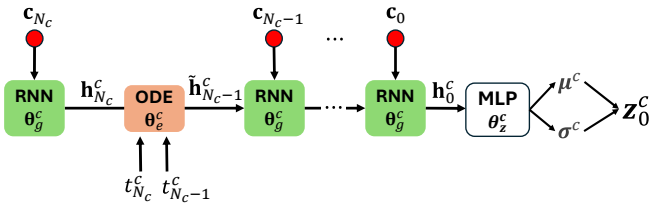


Fig. 3: CSI encoder module for initial condition corresponding to the latent trajectory of the input sequence. The beam SNR encoder is similarly defined.

A. Encoders: An Estimator for Initial Latent Conditions

The purpose of the encoder is to obtain the posterior distribution of an initial latent condition corresponding to an input sequence. We will first consider the CSI encoder.

We start by reversing the input sequence $\mathbf{c}_{N_c}, \cdots, \mathbf{c}_0$ from the last time instance $t_{N_c}^c$ towards the initial time instance t_0. Then, we map $\mathbf{c}_n$ into a hidden vector $\mathbf{h}_n^c \in \mathbb{R}^{H_c \times 1}$ with the help of an auxiliary vector $\tilde{\mathbf{h}}_n^c$

$$\mathbf{h}_n^c = \mathcal{R}(\tilde{\mathbf{h}}_n^c, \mathbf{c}_n; \boldsymbol{\theta}_g^c), \tag{13}$$

where $\mathcal{R}$ can be either GRU or LSTM unit with learnable parameters $\boldsymbol{\theta}_g^c$.

To handle the temporal irregularity $\Delta t_n^c = t_n^c - t_{n-1}^c \neq \Delta t_{n+1}^c$ of the input sequence, one can utilize a numerical ODE solver, e.g., the Euler or Runge-Kutta solvers, to propagate the hidden vector $\mathbf{h}_{n+1}^c$ at time t_{n+1}^c to the auxiliary vector $\tilde{\mathbf{h}}_n^c$ at time t_n^c in Fig. 3 [40]–[43]:

$$\begin{aligned} \tilde{\mathbf{h}}_n^c &= \mathcal{S}(\mathcal{O}_e, \mathbf{h}_{n+1}^c, (t_{n+1}^c, t_n^c); \boldsymbol{\theta}_e^c) \\ &= \mathbf{h}_{n+1}^c + \int_{\tau=t_{n+1}^c}^{t_n^c} \mathcal{O}_e(\mathbf{h}(\tau), \tau; \boldsymbol{\theta}_e^c) d\tau, \end{aligned} \tag{14}$$

where $\mathcal{O}_e$ is a learnable ODE function represented by a neural network with parameters $\boldsymbol{\theta}_e^c$.

By iterating between (13) and (14), we can propagate the hidden vector from $t_{N_c}^c$ to t_0 and output $\mathbf{h}_0^c$, which is used to estimate the initial condition $\mathbf{z}_0^c$ in the latent space and approximate its distribution by a Gaussian distribution with mean $\boldsymbol{\mu}^c$ and variance $(\boldsymbol{\sigma}^c)^2$

$$q_{\theta_c}(\mathbf{z}_0^c | \mathbf{c}_{N_c}, \cdots, \mathbf{c}_0) = q_{\theta_c}(\mathbf{z}_0^c | \mathbf{h}_0^c) = \mathcal{N}(\boldsymbol{\mu}^c, (\boldsymbol{\sigma}^c)^2), \tag{15}$$

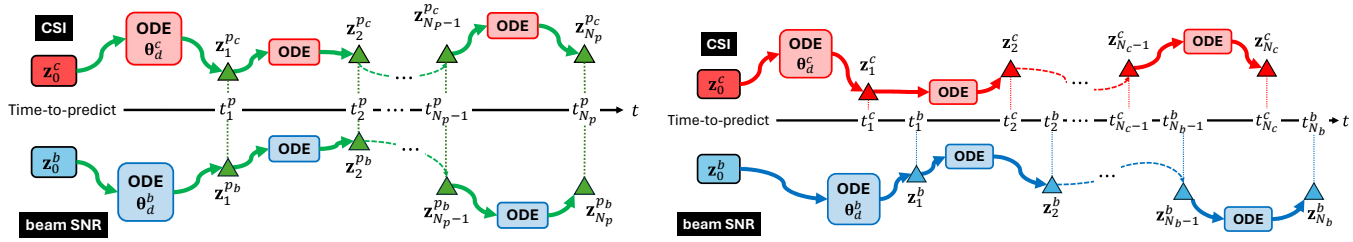
Following the variational autoencoder framework [44], we infer $\boldsymbol{\mu}^c$ and $\boldsymbol{\sigma}^c$ from $\mathbf{h}_0$ as

$$\boldsymbol{\mu}^c, \boldsymbol{\sigma}^c = \mathcal{M}(\mathbf{h}_0^c; \boldsymbol{\theta}_z^c), \tag{16}$$

with $\mathcal{M}$ denoting an MLP network with parameters $\boldsymbol{\theta}_z^c$. Since the initial latent condition $\mathbf{z}_0^c$ is stochastic, we sample it as

$$\mathbf{z}_0^c = \boldsymbol{\mu}^c + \boldsymbol{\sigma}^c \odot \boldsymbol{\epsilon}^c, \ \boldsymbol{\epsilon}^c \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_{L_c}), \tag{17}$$

where $\boldsymbol{\epsilon}^c$ is a standard Gaussian sample of dimension L_c and $\odot$ represents the Hadamard product.



(a) Unrolling latent dynamic ODEs for latent state alignment between CSI and beam SNR at t_n^p.

(b) Unrolling latent dynamic ODEs for latent state recovery CSI at t_n^c (top) and beam SNR at t_n^b (bottom).

Fig. 4: Unrolling latent dynamic ODEs for latent state alignment (a) and latent state recovery (b).

Similarly, we can repeat the process from Eq.(13) to (17) to approximate the posterior distribution and obtain initial latent condition corresponding to the beam SNR sequence

$$q_{\theta_b}(\mathbf{z}_0^b|\mathbf{b}_{N_b},\cdots,\mathbf{b}_0)=\mathcal{N}(\boldsymbol{\mu}^b,(\boldsymbol{\sigma}^b)^2), \tag{18}$$

$$\mathbf{z}_0^b=\boldsymbol{\mu}^b+\boldsymbol{\sigma}^b\odot\boldsymbol{\epsilon}^b,\quad \boldsymbol{\epsilon}^b\sim\mathcal{N}(\mathbf{0},\mathbf{I}_{L_b}), \tag{19}$$

where L_b is the dimension of $\mathbf{z}_0^b$.

B. Latent Dynamic Learning Modules: Alignment in Latent Space

Now, given the initial latent condition $\mathbf{z}_0$ for the input sequence, we employ a unified continuous-time ODE function $\mathcal{O}_d$, modeled by a neural network with parameters θ_d, to unroll the latent dynamics at any query time instance t_n^q. Depending on the query time instance, we have the following three cases.

1) $t_n^q=t_n^p$ for latent state alignment: We first look into the case that the query time t_n^q is the same as the time instance for coordinate estimation t_n^p. This is also the case where we can use the same time instance to align the latent states between the CSI and beam SNR measurements. Specifically, we directly populate the initial latent condition to the latent state at the same query time instance for both CSI and beam SNR [40]–[43]

$$\mathbf{z}_n^{p_c}\triangleq\mathbf{z}_{t_n^p}^c=\mathbf{z}_0^c+\int_{t_0}^{t_n^p}\mathcal{O}_d(\mathbf{z}_t^c,t;\boldsymbol{\theta}_d^c)dt, \tag{20}$$

$$\mathbf{z}_n^{p_b}\triangleq\mathbf{z}_{t_n^p}^b=\mathbf{z}_0^b+\int_{t_0}^{t_n^p}\mathcal{O}_d(\mathbf{z}_t^b,t;\boldsymbol{\theta}_d^b)dt, \tag{21}$$

where $\mathbf{z}_0^c$ and $\mathbf{z}_0^b$ are, respectively, the initial latent conditions for the CSI and beam SNR.

In practice, we incrementally align the latent states at one query time instance at a time and then use the aligned latent states to calculate the next latent states at the next query time t_{n+1}^q. This is illustrated in Fig. 4 (a), where we resort to align the latent states at the first time instance t_1^p for the respective initial conditions $\mathbf{z}_0^c$ and $\mathbf{z}_0^b$,

$$\mathbf{z}_1^{p_c}=\mathbf{z}_0^c+\int_{t_0}^{t_1^p}\mathcal{O}_d(\mathbf{z}_t^c,t;\boldsymbol{\theta}_d^c)dt,$$

$$\mathbf{z}_1^{p_b}=\mathbf{z}_0^b+\int_{t_0}^{t_1^p}\mathcal{O}_d(\mathbf{z}_t^b,t;\boldsymbol{\theta}_d^b)dt,$$

and then

$$\mathbf{z}_2^{p_c}=\mathbf{z}_1^{p_c}+\int_{t_1^p}^{t_2^p}\mathcal{O}_d(\mathbf{z}_t^c,t;\boldsymbol{\theta}_d^c)dt,$$

$$\mathbf{z}_2^{p_b}=\mathbf{z}_1^{p_b}+\int_{t_1^p}^{t_2^p}\mathcal{O}_d(\mathbf{z}_t^b,t;\boldsymbol{\theta}_d^b)dt,$$

where the neural networks to represent latent ODE functions for $\mathbf{z}_t^c$ and $\mathbf{z}_t^b$ are parameterized by $\boldsymbol{\theta}_d^c$ and $\boldsymbol{\theta}_d^b$, respectively.

2) $t_n^q=t_n^c$ for latent state recovery of CSI: Similarly to the above case, we can recover the latent states of the CSI measurements at their original time instances t_n^c

$$\mathbf{z}_n^c\triangleq\mathbf{z}_{t_n^c}^c=\mathbf{z}_0^c+\int_{t_0}^{t_n^c}\mathcal{O}_d(\mathbf{z}_t^c,t;\boldsymbol{\theta}_d^c)dt. \tag{22}$$

3) $t_n^q=t_n^b$ for latent state recovery of beam SNR: The last case is to recover the latent states of the beam SNR measurements at their original time instances t_n^b

$$\mathbf{z}_n^b\triangleq\mathbf{z}_{t_n^b}^b=\mathbf{z}_0^b+\int_{t_0}^{t_n^b}\mathcal{O}_d(\mathbf{z}_t^b,t;\boldsymbol{\theta}_d^b)dt. \tag{23}$$

Note that we set that $t_0\le\min(t_0^c,t_0^b,t_0^p)$. In other words, the time instance for the initial latent conditions are prior to the time instance of the first measurement, either CSI or beam SNR, even before the start of the time window ΔT. More details of setting t_0 can be found in Sec. V.

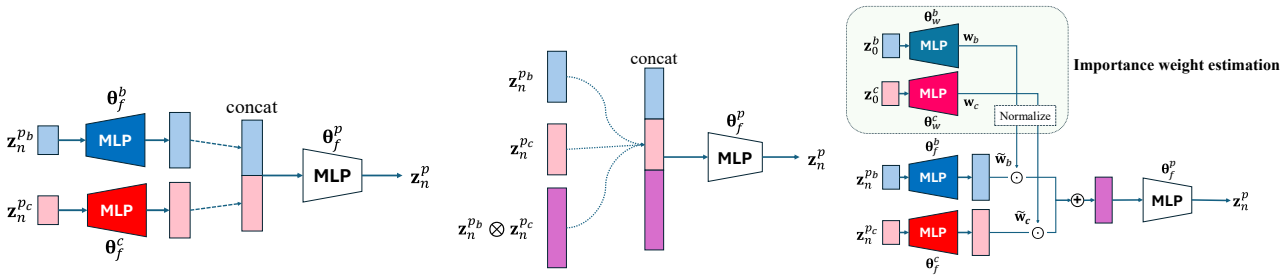
C. Post-ODE Latent Fusion

Once the latent states between CSI and beam SNR measurements are aligned at $\{t_n^p\}_{n=1}^{N_p}$ in (20) and (21), one can fuse them together as $\mathbf{z}_n^p\in\mathbb{R}^{L_p}$ of dimension L_p to support the subsequent coordinate estimate. In the following, we consider three multi-band fusion schemes.

1) MLP Fusion: As shown in Fig. 5 (a), the MLP fusion scheme starts by lifting the aligned latent states $\mathbf{z}_n^{p_c}$ and $\mathbf{z}_n^{p_b}$ to a higher dimension of L_f via separate MLP networks and then projecting the concatenated latent state to a fused latent state $\mathbf{z}_n^p$ as

$$\mathbf{z}_n^p=\mathcal{M}\left(\mathcal{M}(\mathbf{z}_n^{p_b};\boldsymbol{\theta}_f^b)\oplus\mathcal{M}(\mathbf{z}_n^{p_c};\boldsymbol{\theta}_f^c);\boldsymbol{\theta}_f^p\right), \tag{24}$$

where $\boldsymbol{\theta}_f^b$, $\boldsymbol{\theta}_f^c$, and $\boldsymbol{\theta}_f^p$ are learnable parameters for the three MLP networks, and $\oplus$ denotes the vector concatenation, and $\mathbf{z}_n^p$ is of dimension L_p.



(a) MLP Fusion (b) Pairwise Interaction (PI) Fusion (c) Weighted Importance (WI) Fusion

Fig. 5: Schemes for the post-ODE fusion.

2) Pairwise Interaction Fusion: We combine the two aligned latent states, i.e., $\mathbf{z}_n^{p_c}$ and $\mathbf{z}_n^{p_b}$, along with their pairwise interaction $\mathbf{z}_n^{p_b} \otimes \mathbf{z}_n^{p_c} \in \mathbb{R}^{L_b L_c \times 1}$, and feed the expanded multi-band latent states to an MLP for fusion,

$$\mathbf{z}_n^p = \mathcal{M}\left(\mathbf{z}_n^{p_b} \oplus \mathbf{z}_n^{p_c} \oplus (\mathbf{z}_n^{p_b} \otimes \mathbf{z}_n^{p_c}); \boldsymbol{\theta}_f^p\right), \tag{25}$$

where $\otimes$ denotes the Kronecker product, and $\boldsymbol{\theta}_f^p$ represent the MLP parameters. The Kronecker term $\mathbf{z}_n^{p_b} \otimes \mathbf{z}_n^{p_c}$ accounts for cross-modal nonlinearity by expanding the dimension from L_b or L_c to $L_b L_c$ and including all possible element-wise multiplications between the two latent states. This is illustrated in Fig. 5 (b).

3) Weighted Importance Fusion: We also consider a weighted fusion between the two aligned latent states with their respective importance estimated directly from their initial latent conditions. This is illustrated in Fig. 5 (c). Specifically, we first convert the initial latent states $\mathbf{z}_0^b, \mathbf{z}_0^c$ to importance weight vectors of the same dimension L_f:

$$\mathbf{w}_b = \mathcal{M}(\mathbf{z}_0^b; \boldsymbol{\theta}_w^b) \in \mathbb{R}^{L_f}, \mathbf{w}_c = \mathcal{M}(\mathbf{z}_0^c; \boldsymbol{\theta}_w^c) \in \mathbb{R}^{L_f}, \tag{26}$$

where $\boldsymbol{\theta}_w^b$ and $\boldsymbol{\theta}_w^b$ are learnable parameters. Then, we apply the softmax on $[\mathbf{w}_b, \mathbf{w}_c] \in \mathbb{R}^{L_f \times 2}$ over each row such that

$$[\tilde{\mathbf{w}}_b, \tilde{\mathbf{w}}_c] \stackrel{\triangle}{=} \sigma([\mathbf{w}_b, \mathbf{w}_c]) \longrightarrow \tilde{\mathbf{w}}_b + \tilde{\mathbf{w}}_c = \mathbf{1}_{L_f} \tag{27}$$

where $\sigma(\cdot)$ denotes the softmax for importance weight normalization and $\mathbf{1}_{L_f}$ is the all-one vector of dimension L_f. Meanwhile, we lift the aligned latent states $\mathbf{z}_n^{p_c}$ and $\mathbf{z}_n^{p_c}$ to a space of dimension L_f and fuse them by weighting corresponding normalized importance weights as

$$\mathbf{z}_n^p = \mathcal{M}\left([\mathcal{M}(\mathbf{z}_n^{p_b}; \boldsymbol{\theta}_f^b) \odot \tilde{\mathbf{w}}_b] + [\mathcal{M}(\mathbf{z}_n^{p_c}; \boldsymbol{\theta}_f^c) \odot \tilde{\mathbf{w}}_c]; \boldsymbol{\theta}_f^p\right), \tag{28}$$

where $\boldsymbol{\theta}_f^b$, $\boldsymbol{\theta}_f^c$, and $\boldsymbol{\theta}_f^p$ are MLP parameters.

D. Decoders for Trajectory Estimation and Input Sequence Reconstruction

In Fig. 2, the NDF consists of three decoders: one for estimating trajectory coordinates and the other two for reconstructing the CSI embedding and beam SNR sequences.

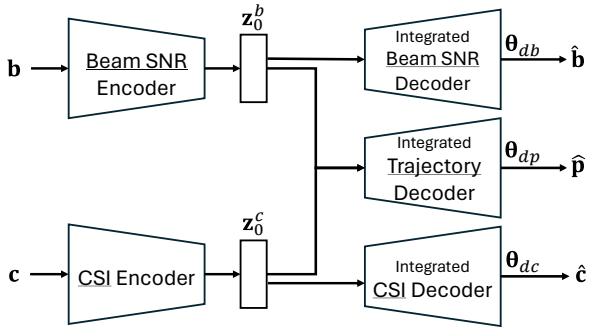


Fig. 6: An equivalent multi-encoder, multi-decoder NDF structure with multiple integrated decoders directly connecting the two initial latent conditions ($\mathbf{z}_0^c$ and $\mathbf{z}_0^b$) to estimated coordinates $\hat{\mathbf{p}}_n$ and reconstructed input sequences $\hat{\mathbf{b}}_n$ and $\hat{\mathbf{c}}_n$.

1) Trajectory Decoding: Given the fused latent state $\mathbf{z}_n^p$ at desired time instances t_n^p, we simply employ an MLP network $\mathcal{M}$ parameterized by $\boldsymbol{\theta}_p$ as a coordinate estimation decoder for estimating trajectory coordinates at t_n^p

$$\hat{\mathbf{p}}_n = \mathcal{M}(\mathbf{z}_n^p; \boldsymbol{\theta}_p), \quad n = 0, \cdots, N_p, \tag{29}$$

where $\hat{\mathbf{p}}_n = [\hat{x}_n, \hat{y}_n]^T$ is the coordinate estimate at t_n^p and $\boldsymbol{\theta}_p$ is shared over all time instances of t_n^p.

By combining the latent dynamic learning modules of (20) and (21), the post-ODE fusion module (either (24), (25) or (28)), and the above trajectory decoder of (29), we establish the **Integrated Trajectory Decoder** $\mathcal{P}(\cdot)$. This integrated decoder can be considered to directly take the two initial latent conditions $\mathbf{z}_0^c$ and $\mathbf{z}_0^b$ and output the coordinate estimate $\hat{\mathbf{p}}_n$

$$\hat{\mathbf{p}}_n = \mathcal{P}(\mathbf{z}_0^c, \mathbf{z}_0^b, t_0, t_n^p; \boldsymbol{\theta}_{dp}), \tag{30}$$

where $\boldsymbol{\theta}_{dp} = \{\boldsymbol{\theta}_d^c, \boldsymbol{\theta}_d^b, \boldsymbol{\theta}_f, \boldsymbol{\theta}_p\}$ with $\boldsymbol{\theta}_f$ encompassing all learnable parameters in the post-ODE fusion model. For instance, $\boldsymbol{\theta}_f = \{\boldsymbol{\theta}_f^c, \boldsymbol{\theta}_f^b, \boldsymbol{\theta}_f^p\}$ for the MLP fusion, while $\boldsymbol{\theta}_f = \{\boldsymbol{\theta}_w^c, \boldsymbol{\theta}_w^b, \boldsymbol{\theta}_f^c, \boldsymbol{\theta}_f^b, \boldsymbol{\theta}_f^p\}$ for the weighted importance fusion. As illustrated in Fig. 6, this integrated decoder structure directly links the initial latent conditions to the coordinate output and simplifies the derivation of the ELBO-based loss function in the next section. We hereafter group the estimated trajectory coordinates as $\hat{\mathbf{p}} = \{\hat{\mathbf{p}}_n\}_{n=0}^{N_p}$.

2) CSI Decoding: Given the CSI latent states $\mathbf{z}_n^c$ of (22) at their original time instances t_n^c, we employ another MLP decoder with parameters $\boldsymbol{\theta}_c$ to project $\mathbf{z}_n^c$ back to the CSI embedding sequence as

$$\hat{\mathbf{c}}_n = \mathcal{M}(\mathbf{z}_n^c; \boldsymbol{\theta}_c), \quad n = 0, \cdots, N_c, \tag{31}$$

where $\boldsymbol{\theta}_c$ is shared over all time instances of t_n^c.

Similar to the integrated trajectory decoder $\mathcal{P}(\cdot)$, we combine the latent dynamic learning (20) and the above CSI decoder (31) and establish the **Integrated CSI Decoder** $\mathcal{C}(\cdot)$

$$\hat{\mathbf{c}}_n = \mathcal{C}(\mathbf{z}_0^c, t_0, t_n^c; \boldsymbol{\theta}_{dc}), \tag{32}$$

where $\boldsymbol{\theta}_{dc} = \{\boldsymbol{\theta}_d^c, \boldsymbol{\theta}_c\}$. Equivalently, the integrated CSI decoder takes the initial latent condition $\mathbf{z}_0^c$ corresponding to the CSI input sequence and reconstructs the CSI embedding sequence as $\hat{\mathbf{c}}_n$ in t_n^c. We also group the estimated CSI embedding sequence as $\hat{\mathbf{c}} = \{\hat{\mathbf{c}}_n\}_{n=0}^{N_c}$.

3) Beam SNR Decoding: The last decoder is to project the latent states $\mathbf{z}_n^b$ of (23) at t_n^b back to the beam SNR sequence as

$$\hat{\mathbf{b}}_n = \mathcal{M}(\mathbf{z}_n^b; \boldsymbol{\theta}_b), \quad n = 0, \cdots, N_b, \tag{33}$$

where $\boldsymbol{\theta}_b$ is shared over all time instances of t_n^b.

Combining the latent dynamic learning (21) and the above beam SNR decoder (33), we establish the **Integrated Beam SNR Decoder** $\mathcal{B}(\cdot)$ as

$$\hat{\mathbf{b}}_n = \mathcal{B}(\mathbf{z}_0^b, t_0, t_n^b; \boldsymbol{\theta}_{db}), \tag{34}$$

where $\boldsymbol{\theta}_{db} = \{\boldsymbol{\theta}_d^b, \boldsymbol{\theta}_b\}$. We also group the estimated beam SNR as $\hat{\mathbf{b}} = \{\hat{\mathbf{b}}_n\}_{n=0}^{N_b}$. We introduce CSI/beam SNR reconstruction to make latent dynamics ODEs strongly conditioned by the dynamics of multi-band wireless propagation. If the network is trained with a single decoder for coordinate regression, the latent dynamic ODE and the coordinate decoder can independently learn sets of parameters that are not subject to the constraints of the wireless dynamics.

Remark: The utilization of ODEs in both encoders and latent dynamic ODEs offers advantages beyond merely managing asynchronous-sampled signals but also facilitates the alignment of two distinct wireless modalities while simultaneously accounting for the temporal evolution of their dynamics. In our end-to-end structure, the continuous dynamics of CSI and beam SNR are encoded into the latent distributions using distinct ODE encoders. We impose a constraint on both encoders to project input trajectories to the initial latent state following a normal distribution, thereby projecting the initial latent states for CSI and beam SNR to the unified latent space. Modeling ODEs as neural networks enables the encoders to acquire more appropriate parameters to satisfy the constraint due to the flexibility of ODEs that can express the evolution of a state between consecutive time steps in a complex manner. Additionally, we obtain aligned initial latent states at the time preceding the initial samples in both sequences, which helps the latent dynamic ODEs to regress latent trajectories starting from same starting time. This is a notable advantage of our NDF framework over conventional RNN and RNN-variant methods in which the hidden states are consistent or merely decay along time with relatively simple functions until the next step.

E. Complexity Analysis

The time complexity of our NDF is estimated by considering the number of components in each module in the architecture. First, the time complexity of the CSI encoder depends on the number of hidden units in recurrent layers H_e and the number of time steps in the input CSI sequence T_c and is approximated as $\mathcal{O}(H_e^2 T_c)$. Similarly, the complexity of the beam SNR encoder is approximated as $\mathcal{O}(H_e^2 T_b)$ where T_b is the number of time steps in the input beam SNR sequence. For the latent dynamic ODE of CSI, the complexity is determined by the queued time steps for CSI reconstruction T_c and for trajectory estimation T_p and is approximated as $\mathcal{O}(T_c + T_p)$. Similarly, the complexity of the latent dynamic ODE of beam SNR is approximated as $\mathcal{O}(T_b + T_p)$. Finally, the complexity of the CSI decoder, beam SNR decoder, and coordinate decoder are approximated as $\mathcal{O}(T_c(L+1)C)$, $\mathcal{O}(T_b(L+1)B)$, and $\mathcal{O}(T_p(L+1))$, respectively, where L is the dimension of latent variables and C, B are the dimension of reconstructed CSI and beam SNR. Therefore, the total time complexity of our NDF is the sum of those of the components, as $\mathcal{O}((H_e^2(T_b+T_c))+(T_b+T_c+T_p)+((L+1)(T_cC+T_bB+T_p)))$.

IV. ELBO-Based Loss Function

In the following, we derive an ELBO-based loss function that accounts for the multi-encoder, multi-decoder NDF architecture. By grouping $\mathbf{b} = \{\mathbf{b}_n\}_{n=0}^{N_b}$, $\mathbf{c} = \{\mathbf{c}_n\}_{n=0}^{N_c}$, and $\mathbf{p} = \{\mathbf{p}_n\}_{n=0}^{N_p}$ as illustrated in Fig. 6, The modified ELBO can be expressed as [44]

$$\begin{aligned}
\text{ELBO} =& \mathbb{E}_{q(\mathbf{z}_0^b, \mathbf{z}_0^c|\mathbf{b},\mathbf{c})}[\log p(\mathbf{p}|\mathbf{z}_0^b, \mathbf{z}_0^c)] \\
&+ \lambda_1 \mathbb{E}_{q(\mathbf{z}_0^b|\mathbf{b})}[\log p(\mathbf{b}|\mathbf{z}_0^b)] + \lambda_2 \mathbb{E}_{q(\mathbf{z}_0^c|\mathbf{c})}[\log p(\mathbf{c}|\mathbf{z}_0^c)] \\
&- \lambda_3 D_{\mathrm{KL}}[q(\mathbf{z}_0^b, \mathbf{z}_0^c|\mathbf{b}, \mathbf{c})||p(\mathbf{z}_0^b, \mathbf{z}_0^c)] \\
\overset{(a)}{\approx}& \frac{1}{V}\sum_{v=1}^{V} \log p(\mathbf{p}|\mathbf{z}_0^{b(v)}\mathbf{z}_0^{c(v)}) \\
&+ \frac{\lambda_1}{V}\sum_{v=1}^{V} \log p(\mathbf{b}|\mathbf{z}_0^{b(v)}) + \frac{\lambda_2}{V}\sum_{v=1}^{V} \log p(\mathbf{c}|\mathbf{z}_0^{c(v)}) \\
&- \lambda_3 D_{\mathrm{KL}}[q(\mathbf{z}_0^b, \mathbf{z}_0^c|\mathbf{b}, \mathbf{c})||p(\mathbf{z}_0^b, \mathbf{z}_0^c)],
\end{aligned} \tag{35}$$

where $q(\mathbf{z}_0^c|\mathbf{c})$ and $q(\mathbf{z}_0^b|\mathbf{b})$ are the approximate posterior distributions defined in (15) and (18), respectively, the joint posterior distribution of $\mathbf{z}_0^c$ and $\mathbf{z}_0^b$ can be factorized as

$$q(\mathbf{z}_0^b, \mathbf{z}_0^c|\mathbf{b}, \mathbf{c}) = q(\mathbf{z}_0^c|\mathbf{c})q(\mathbf{z}_0^b|\mathbf{b}), \tag{36}$$

due to the independence assumption between the two input sequences and the use of separate encoders, $\{\lambda_i\}_{i=1}^3$ are regularization weights, $p(\mathbf{z}_0^b, \mathbf{z}_0^c)$ are the joint prior of $\mathbf{z}_0^c$ and $\mathbf{z}_0^b$ that can be also factorized as

$$p(\mathbf{z}_0^b, \mathbf{z}_0^c) = p(\mathbf{z}_0^b)p(\mathbf{z}_0^c), \tag{37}$$

with $p(\mathbf{z}_0^b) \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_{L_b})$ and $p(\mathbf{z}_0^c) \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_{L_c})$, and $p(\mathbf{p}|\mathbf{z}_0^b, \mathbf{z}_0^c)$, $p(\mathbf{c}|\mathbf{z}_0^c)$ and $p(\mathbf{b}|\mathbf{z}_0^b)$ denote the output likelihood functions of the three integrated (trajectory/CSI/beam SNR)

decoders in Fig. 6. In the above equation, (a) holds as we replace the posterior mean by its sample mean over V samples of the two initial latent conditions $\mathbf{z}_0^c$ and $\mathbf{z}_0^b$ according to (17) and (19), respectively, with V independent realizations of $\boldsymbol{\epsilon}^c$ and $\boldsymbol{\epsilon}^b$. In practice, the number of initial latent conditions is set to $V = 1$ as one can average over the independent realizations within the minibatch samples.

For the KL divergence term $D_{\mathrm{KL}}[q(\mathbf{z}_0^b, \mathbf{z}_0^c|\mathbf{b}, \mathbf{c})||p(\mathbf{z}_0^b, \mathbf{z}_0^c)]$, we invoke the independent condition between the posterior distributions of $\mathbf{z}_0^b$ and $\mathbf{z}_0^c$ given the input sequences $\mathbf{b}$ and $\mathbf{c}$ and between the prior distributions of $\mathbf{z}_0^b$ and $\mathbf{z}_0^c$

$$\begin{aligned} &D_{\mathrm{KL}}[q(\mathbf{z}_0^b, \mathbf{z}_0^c|\mathbf{b}, \mathbf{c})||p(\mathbf{z}_0^b, \mathbf{z}_0^c)] \\ &\overset{(a)}{=} D_{\mathrm{KL}}[q(\mathbf{z}_0^b|\mathbf{b})q(\mathbf{z}_0^c|\mathbf{c})||p(\mathbf{z}_0^b)p(\mathbf{z}_0^c)] \\ &\overset{(b)}{=} D_{\mathrm{KL}}[q(\mathbf{z}_0^b|\mathbf{b})||p(\mathbf{z}_0^b)] + D_{\mathrm{KL}}[q(\mathbf{z}_0^c|\mathbf{c})||p(\mathbf{z}_0^c)], \end{aligned} \tag{38}$$

where (a) holds due to the factorization in (36) and (37), and (b) can be derived using (61) in Appendix C. Then it is straightforward to show that

$$\begin{aligned} D_{\mathrm{KL}}[q(\mathbf{z}_0^c|\mathbf{c})||p(\mathbf{z}_0^c)] &= D_{\mathrm{KL}}[\mathcal{N}(\boldsymbol{\mu}^c, \boldsymbol{\sigma}^c)||\mathcal{N}(\mathbf{0}, \mathbf{I}_{L_c})] \\ &= \frac{1}{2}\sum_{l=1}^{L_c}\left((\mu_l^c)^2 + (\sigma_l^c)^2 - 1 - \log(\sigma_l^c)^2\right), \end{aligned} \tag{39}$$

$$\begin{aligned} D_{\mathrm{KL}}[q(\mathbf{z}_0^b|\mathbf{b})||p(\mathbf{z}_0^b)] &= D_{\mathrm{KL}}[\mathcal{N}(\boldsymbol{\mu}^b, \boldsymbol{\sigma}^b)||\mathcal{N}(\mathbf{0}, \mathbf{I}_{L_b})] \\ &= \frac{1}{2}\sum_{l=1}^{L_b}\left((\mu_l^b)^2 + (\sigma_l^b)^2 - 1 - \log(\sigma_l^b)^2\right), \end{aligned} \tag{40}$$

where $\mu_l^{b/c}$ and $\sigma_l^{b/c}$ are the l-th element of $\boldsymbol{\mu}^{b/c}$ and $\boldsymbol{\sigma}^{b/c}$, respectively.

For output log-likelihood functions, we start with the integrated trajectory decoder $\mathcal{P}(\cdot)$ that takes the two initial latent conditions and estimates the trajectory coordinates at t_n^p,

$$\begin{aligned} \log p(\mathbf{p}|\mathbf{z}_0^b, \mathbf{z}_0^c) &= \log p(\mathbf{p}_0, \mathbf{p}_1, \cdots, \mathbf{p}_{N_p}|\mathbf{z}_0^b, \mathbf{z}_0^c) \\ &\overset{(a)}{\approx} \sum_{n=1}^{N_p} \log p(\mathbf{p}_n|\mathbf{z}_0^b, \mathbf{z}_0^c), \end{aligned} \tag{41}$$

where the approximation (a) holds as we invoke an independent assumption over the sequential coordinate outputs over the time instance n, primarily for the sake of closed-form expression of the joint likelihood. We can assume that each element in $\mathbf{p}_n = [x_n, y_n]^\top$ follows a Laplace distribution:

$$\begin{aligned} p(x_n|\mathbf{z}_0^b, \mathbf{z}_0^c) &= \frac{1}{2b_p}\exp\left(-\frac{|x_n - \hat{x}_n|}{b_p}\right), \\ p(y_n|\mathbf{z}_0^b, \mathbf{z}_0^c) &= \frac{1}{2b_p}\exp\left(-\frac{|y_n - \hat{y}_n|}{b_p}\right), \end{aligned} \tag{42}$$

where $b_p \in \mathbb{R}$ is a scaling parameter and $\hat{\mathbf{p}}_n = [\hat{x}_n, \hat{y}_n]^\top = \mathcal{P}(\mathbf{z}_0^c, \mathbf{z}_0^b, t_0, t_n^p; \boldsymbol{\theta}_{dp})$ is the estimated trajectory coordinate at t_n^p. As a result, we can show that

$$\log p(\mathbf{p}_n|\mathbf{z}_0^b, \mathbf{z}_0^c) \propto -\frac{\|\mathbf{p}_n - \hat{\mathbf{p}}_n\|_1}{b_p}, \tag{43}$$

where $\|\cdot\|_1$ denotes the ℓ_1 norm. Assuming $b_p = 1$ and plugging the above equation back to (41), the output log-likelihood function of the integrated trajectory decoder is given as

$$\log p(\mathbf{p}|\mathbf{z}_0^b, \mathbf{z}_0^c) \propto -\sum_{n=1}^{N_p}\|\mathbf{p}_n - \hat{\mathbf{p}}_n\|_1. \tag{44}$$

It is seen that maximizing this log-likelihood is equivalent to minimizing mean absolute error (MAE) between ground truth and estimated trajectory coordinates. We can follow Eq. (41) to (44) for the output log-likelihood functions of the integrated beam SNR and CSI decoders,

$$\log p(\mathbf{b}|\mathbf{z}_0^b) \propto -\sum_{n=1}^{N_b}\|\mathbf{b}_n - \hat{\mathbf{b}}_n\|_1, \tag{45}$$

$$\log p(\mathbf{c}|\mathbf{z}_0^c) \propto -\sum_{n=1}^{N_c}\|\mathbf{c}_n - \hat{\mathbf{c}}_n\|_1. \tag{46}$$

Combining the KL divergence term and the output log-likelihood functions of the integrated decoders, the modified ELBO (35) reduces to the following loss function

$$\begin{aligned} \mathcal{L} = &\sum_{n=1}^{N_p}\|\mathbf{p}_n - \hat{\mathbf{p}}_n\|_1 + \lambda_1\sum_{n=1}^{N_b}\|\mathbf{b}_n - \hat{\mathbf{b}}_n\|_1 + \lambda_2\sum_{n=1}^{N_c}\|\mathbf{c}_n - \hat{\mathbf{c}}_n\|_1 \\ &+\lambda_3\sum_{l=1}^{L_b}\left((\mu_l^b)^2 + (\sigma_l^b)^2 - 1 - \log(\sigma_l^b)^2\right) \\ &+\lambda_4\sum_{l=1}^{L_c}\left((\mu_l^c)^2 + (\sigma_l^c)^2 - 1 - \log(\sigma_l^c)^2\right) \end{aligned} \tag{47}$$

where we relax the regularization weight λ_3 for the joint KL term to different regularization weights λ_3 and λ_4 for individual KL terms of beam SNR and CSI, respectively.

V. Performance Evaluation

A. In-House Testbed and Data Collection

We upgrade our previous in-house testbed in [16] and [17] from collecting single-band beam SNRs to simultaneously gathering both 5-GHz CSI and 60-GHz beam SNRs. As shown in Fig. 7, we mount two routers on a TurtleBot as a mobile user: one router is the 802.11ac-compliant ASUS RT-AC86U device to collect 80-MHz CSI data at 5 GHz and the other 802.11ad-compliant TP-Link Talon AD7200 router for 60-GHz beam SNR. The mobile user TurtleBot moves along predefined rectangular trajectories (denoted by red dot lines in the right plot of Fig. 7) in a large conference room. Positioned at the lower left corner of the rectangular trajectory, another pair of identical routers act as a multi-band AP.

To enable data collection on these commercial-off-the-shelf routers, we replace the original firmware with open-source ones [45], [46] and follow the methods of [47] and [45] to extract the beam SNR and CSI from the commercial routers. From the four antennas (three external and one internal) of the ASUS router, we are able to extract $N_{\mathrm{Tx}} \times N_{\mathrm{Rx}} = 4 \times 2$ spatial streams of CSI over $N_{\mathrm{s}} = 234$ subcarriers, excluding null subcarriers. Each raw CSI frame $\mathbf{C}_n \in \mathbb{C}^{4\times 2\times 234}$ is calibrated and compressed in the CSI embedding input $\mathbf{c}_n \in \mathbb{R}^{36\times 1}$ with $M_c = 36$, as described in Appendix A. On the other hand, the TP-Link router employs an analog phase array of 32

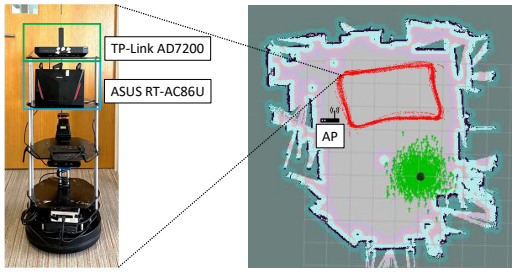


Fig. 7: In-house multi-band Wi-Fi testbed: a TurtleBot with a TP-Link router for 60-GHz beam SNR and an ASUS router for 5-GHz CSI (left); and the data collection floorplan with the AP location and trajectories (right).

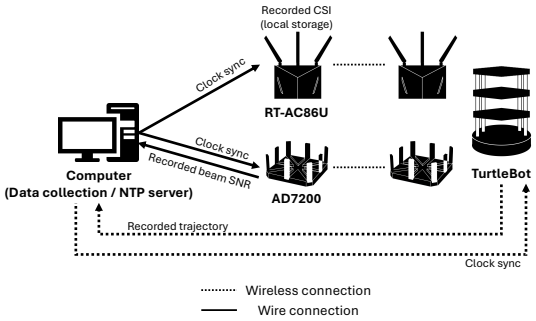


Fig. 8: The device connection and data flow of the testbed.

antenna elements and sequentially scans $M_b = 36$ predefined directional beam patterns, leading to $\mathbf{b}_n \in \mathbb{R}^{36\times 1}$.

Our testbed is also equipped with a LiDAR and a wheel encoder to self-localize over a predefined map. The self-localized coordinates, recorded at a frame rate of 10 frames per second (fps), are then used as ground-truth labels $\mathbf{p}_n$ for trajectory estimation. The system clocks of all networked devices, including routers and TurtleBot, are precisely synchronized using the Network Time Protocol (NTP) with a central desktop acting as the NTP server. The desktop controls and aligns the clocks of all other devices on the network connection, ensuring that the timestamps across the network refer to the same clock. Consequently, we obtained $43,237$ frames for CSI and coordinate labels, and $9,590$ frames for beam SNRs. All parameters and values related to our evaluation and the dataset are listed in Table I.

B. Implementation

We set $\Delta T_w = 5$ seconds to group all collected CSI frames, beam SNRs, and coordinate labels into sequences. Fig. 9 shows the histograms of the number of CSI (top) and beam SNR (bottom) frames over non-overlapping sequences of 5 seconds. It reveals that most CSI sequences have $20-40$ frames over a period of 5 seconds, yielding an average frame rate of $4-8$ fps. In comparison, the beam SNR sequences contain much less number of frames over 5 seconds, yielding an average frame rate of 1 fps. For each sequence, all timestamps $\{t_n^b\}_{n=0}^{N_b}$, $\{t_n^c\}_{n=0}^{N_c}$, and $\{t_n^p\}_{n=0}^{N_p}$ are normalized into $[0,1]$ by dividing the relative timestamps by 5 seconds.

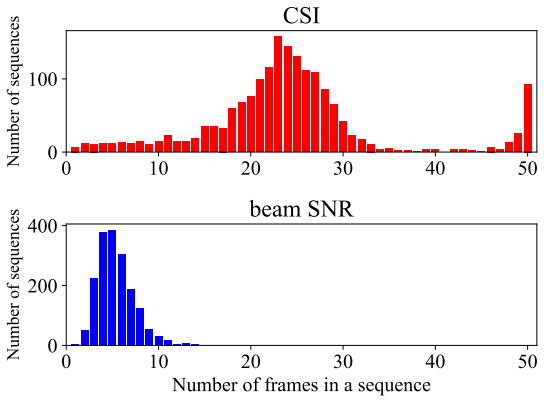


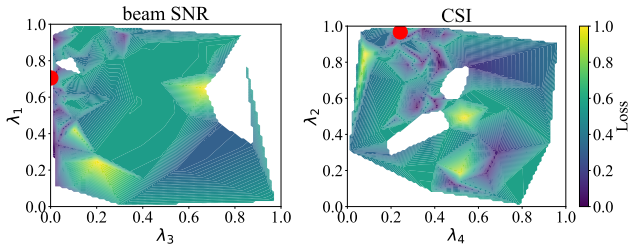
Fig. 9: The histogram of the number of CSI (top) and beam SNR (bottom) over non-overlapping sequences.

	Parameter	Notation	Value
Device	**CSI**		
	- Wireless Standard	-	IEEE 802.11ac
	- Center Frequency	-	5 GHz
	- Bandwidth	-	80 MHz
	- Tx antenna	N_{Tx}	4
	- Rx antenna	N_{Rx}	2
	- Subcarriers	N_s	234
	beam SNR		
	- Wireless Standard	-	IEEE 802.11ad
	- Center Frequency	-	60 GHz
	- Bandwidth	-	2.16 GHz
	- Tx antenna	-	32
	- Rx antenna	-	32
	- Beam sectors	N_b	36
Dataset	**Total path length**	-	2178 m
	# of samples		
	- CSI	-	43,237
	- beam SNR	-	9,590
	Sampling rate		
	- CSI	-	7.44 Hz
	- beam SNR	-	1.71 Hz
Training	**CSI embedding dimension**	M_c	36
	GRU hidden dimension	H_b, H_c	20
	Latent dimension	L_b, L_c, L_p	20
	Fusion latent dimension	L_f	128
	ODE solver		
	- Encoder	$\mathcal{O}_e$	Euler solver
	- Latent dynamic ODE	$\mathcal{O}_d$	Dopri5 solver
	Loss coefficients	$\lambda_1, \lambda_2, \lambda_3, \lambda_4$	0.7, 1.0, 0.0010, 0.25
	Epochs	-	250
	Batch size	-	32
	Optimizer	-	Adamax
	Learning rate	-	4e-3
	LR scheduler	-	OneCycle

TABLE I: A list of parameters in experiments.

We consider three data splits for performance evaluation:

1) **Random Split**: We group the frames into $1,778$ non-overlapping sequences of 5 seconds (with a step size of 5 seconds) and randomly divide these non-overlapping sequences into train, validation, and test sets with a ratio of $80:10:10$. The results in the following subsections are based on this data split.
2) **Temporal Split**: We divide all frames into training and test sets strictly according to their chronological order. Specifically, we group the first collected $s\%$ of frames into the training set, and the remaining frames into the test set. In other words, all test frames represent future data that was not seen during the training phase. This sequential split is used to evaluate the temporal generalization performance in Section V-G with different values of s. With a sequence length of 5 seconds, we group the training and test frames into sequences with



(a) beam SNR (b) CSI

Fig. 10: The search for regularization parameters in the ELBO-based loss function of (47).

step sizes of 1 and, respectively, 5 seconds.

3) **Coordinate Split**: We also divide all frames into training and test sets according to their ground truth coordinates. Specifically, we keep frames from a particular area (e.g., a corner) in the test set, completely unseen from the training set. This split is used to evaluate the generalization performance at unseen coordinates in Section V-H, which is referred to as the spatial generalization.

We use an autoencoder with 3 1D convolutional layers and 3 MLP layers for the pretraining discussed in Appendix A to obtain the CSI embedding vector $\mathbf{c}_n$. Both beam SNR and CSI embedding sequences are normalized to $[0, 1]$. For the encoder, we use the GRU unit with a hidden dimension of $H_b = H_c = 20$ for the beam SNR and CSI input sequences. We set $L_b = L_c = 20$ for the dimensions of the initial latent condition and unrolled latent states $\mathbf{z}_n^b$ and $\mathbf{z}_n^c$, $n = 0, 1, \cdots, N_b/N_c$. We lift the aligned latent state to a space of dimension $L_f = 128$ before projecting it back to the fused latent space of $L_p = 20$. We employ the Euler and Dopri5 ODE solvers for encoding and latent dynamic ODE, respectively. The decoders for trajectory (29), CSI (31) and beam SNR (33) share the same MLP architecture of three MLP layers.

The set of regularization parameters is chosen by performing a hyperparameter search in the interval of $[0, 1]$ using Optuna [48]. It is based on the validation loss transition of coordinate estimation within 125 epochs and 100 trials are executed. Fig. 10 illustrates the loss function as a function of regularization parameters (λ_1, λ_3) for beam SNR and (λ_2, λ_4) for CSI, where red dots denote the values of hyperparameter pairs achieving the smallest validation loss over 125 epochs or the smallest intermediate loss if terminated in an earlier epoch. As a result, we set the regularization parameters as $\lambda_1 = 0.7, \lambda_2 = 1.0, \lambda_3 = 0.0010, \lambda_4 = 0.25$ in the ELBO-based loss function of (47). To train the NDF network, we set the minibatch size to 32 and the maximum number of epochs is 250, and we save the model achieving the best validation loss while training. We used the Adamax optimizer with the maximum learning rate of $4e-3$ with the OneCycle learning rate scheduling for fast convergence [49].

C. Evaluation Metrics

As the localization error, we use the Euclidean distance from the estimated coordinate $\hat{\mathbf{p}} = [\hat{x}, \hat{y}]$ and the ground truth $\mathbf{p} = [x, y]$, which is calculated for each estimation and averaged across all N estimations, as

$$L_{\text{error}} = \frac{1}{N}\sum_{i=1}^{N} \|\mathbf{p} - \hat{\mathbf{p}}\|. \tag{48}$$

Furthermore, we compute the cumulative distribution function (CDF) for localization error to effectively demonstrate and compare the error distributions in various methods. The CDF for the localization error $F_e(t)$ is calculated as

$$F_e(t) = \frac{1}{N}\sum_{i=1}^{N} \mathbb{I}(e_i \le t), \tag{49}$$

where t is the error threshold, e_i is a localization error of i-th estimation point as $e_i = \|\mathbf{p}_i - \hat{\mathbf{p}}_i\|$, and $\mathbb{I}$ is the indicator function defined as

$$\mathbb{I}(e_i \le t) = \begin{cases} 1, & e_i \le t, \\ 0, & \text{otherwise.} \end{cases} \tag{50}$$

D. Comparison to Baseline Methods

For performance comparison, we consider a comprehensive list of baseline methods

- **Single-band (CSI)** methods: Solving trajectory regression written as $\{\mathbf{c}_n, t_n^c\}_{n=0}^{N_c} \longrightarrow \{\mathbf{p}_n, t_n^p\}_{n=0}^{N_p}$ with 1) Linear interpolation (LinearInt) of (3); 2) Nearest interpolation (NearestInt) of (4); 3) RNN-Decay of (8); 4) RNN-Δ of (9); 5) DDND of [16], [17].
- **Single-band (beam SNR)** methods: Solving trajectory regression written as $\{\mathbf{b}_n, t_n^b\}_{n=0}^{N_b} \longrightarrow \{\mathbf{p}_n, t_n^p\}_{n=0}^{N_p}$ with 1) Linear interpolation (LinearInt) of (3); 2) Nearest interpolation (NearestInt) of (4); 3) RNN-Decay of (8); 4) RNN-Δ of (9); 5) DDND of [16], [17].
- **Frame-to-Frame Fusion** methods: 1) LinearInt fusion of (3) and (6); 2) NearestInt fusion of (4) and (6).
- **Sequence-to-Sequence Fusion** methods: 1) RNN-Decay fusion of (8), (10) and (12); 2) RNN-Δ fusion of (9), (11) and (12).

Table II summarizes the trajectory estimation performance of all baseline methods and the proposed NDF method under the random sequence split. By comparing the mean, median, and the 90th percentile of the localization error in the unit of meters, it is seen that, for a given method, e.g., the linear interpolation or the RNN-Decay, the multi-band fusion improves the localization performance from either the CSI-only or the beam SNR-only methods. Comparison between the interpolation (i.e., linear and nearest) and RNN methods (i.e., RNN-Decay and RNN-Δ) shows that the RNN-based methods can significantly improve the CSI-only performance and contribute to the overall improvement using both CSI and beam SNR. If we narrow down to the last column of Table II, it is clear that, by properly aligning the latent states using the latent dynamic ODE, the NDF can further reduce the location error from the best multi-band baseline (i.e., RNN-Decay) to a mean localization error of 14.8 cm. Fig. 11 highlights the cumulative distribution functions (CDFs) of the localization error from the multi-band methods and the NDF.

TABLE II: Localization errors (m) for all single-band and multi-band baseline and the proposed NDF methods.

	Single-band						Multi-band		
	CSI			beam SNR					
	Mean	Median	CDF@0.9	Mean	Median	CDF@0.9	Mean	Median	CDF@0.9
LinearInt	1.79	1.89	2.91	0.932	0.726	1.77	0.764	0.506	1.76
NearestInt	1.82	1.92	2.87	1.00	0.715	2.09	0.839	0.553	2.01
RNN-Decay	1.02	0.739	2.27	1.03	0.545	2.64	0.685	0.432	1.69
RNN-Δ	1.03	0.733	2.39	0.994	0.499	2.70	0.506	0.215	1.42
DDND	0.975	0.588	2.45	0.390	0.191	0.859	-	-	-
NDF (ours)	-	-	-	-	-	-	**0.263**	**0.148**	**0.611**

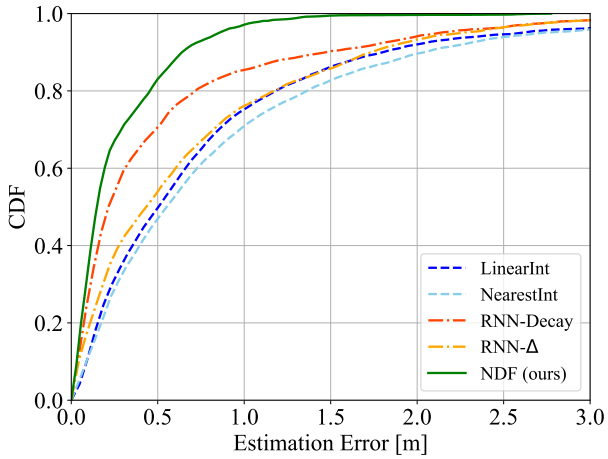


Fig. 11: Cumulative distribution function (CDF) of localization errors for the baseline and NDF methods.

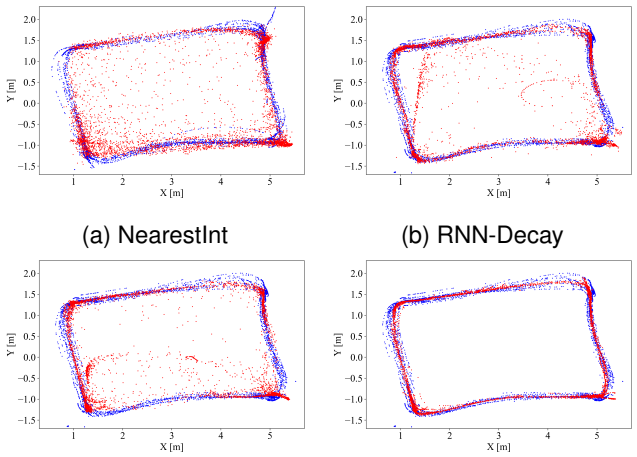


Fig. 12: Visualization of estimated trajectories (red) and ground truth (blue) for selected multi-band baseline methods and the proposed NDF.

To qualitatively compare the baseline and proposed methods, we overlap the estimated trajectories (in red dots) with the ground truth coordinates (in dim blue dots) in Fig. 12 for selected multi-band fusion baseline methods (nearest interpolation, RNN-Decay, RNN-Δ) and the proposed NDF method. The improvement from frame-to-frame fusion (nearest interpolation) to sequence-to-sequence fusion (RNN-Decay, RNN-Δ) is noticeable, as there are less localization errors in the center of the rectangular area. The outliers in the trajectories arise because the baselines fail to accurately model the temporal evolution of the trajectory, leading to outputs where the estimated coordinates for the next time step can be far from the coordinates at the current time step. Our NDF addresses this issue through its architectural design, which comprises two pairs of ODE modules capable of modeling the temporal evolution of wireless propagation and device locations, as well as their correlation. The estimation of a sequence of latent variables corresponding to the time steps at which we wish to estimate a trajectory is fully conditioned on the learned derivatives and is highly probable to draw a practically natural transition from one location to the next. Referring to Fig. 12, our NDF shows the best results by significantly reducing outliers and forcing the trajectory estimates along the rectangular track.

E. Impact of Sequence Length ΔT_w

In the following, we investigate the impact of sequence length ΔT_w on the trajectory estimation performance. Given the frame rate of about 5 Hz for CSI and about 1 Hz for beam SNR, the number of effective samples is proportional to the sequence length ΔT_w. For a given sequence length ΔT_w, we follow the random split protocol to segment the raw data into non-overlapping ΔT_w-sec sequences with $\Delta T_w = \{2, 8\}$ seconds.

Table III lists the trajectory estimation errors in terms of mean, median, and the 90-th percentile of the CDF for three choices of ΔT_w. Overall, it confirms that, the longer the sequence, the better the trajectory estimation performance. In the case of $\Delta T_w = 2$ seconds, there might not be sufficient beam SNR samples for latent dynamic learning as the frame rate is limited to about 1 Hz. The choice of $\Delta T_w = 8$ seconds appears to give lower median and CDF@0.9 localization errors while keeping the mean error close to that of $\Delta T_w = 5$ seconds.

F. Impact of Fusion Scheme

In the following, we examine the impact of the three fusion schemes in Sec. III-C on trajectory estimation accuracy, using the random split protocol of non-overlapping 5-sec input

TABLE III: Impact of sequence length ΔT_w.

Sequence length	Mean	Median	CDF@0.9
2 seconds	0.481	0.233	1.20
5 seconds	**0.263**	0.148	0.611
8 seconds	0.270	**0.136**	**0.547**

TABLE IV: The impact of the three fusion schemes in Sec. III-C on the trajectory estimation error (m).

Fusion scheme	Mean	Median	CDF@0.9
MLP	**0.263**	**0.148**	**0.611**
Pairwise Interaction	0.397	0.230	0.903
Weighted Importance	0.287	0.164	0.618

sequences. As shown in Table IV, the MLP fusion scheme delivers the best results in terms of mean, median, and 90th-percentile CDF, with the weighted importance fusion scheme showing nearly identical performance. Moreover, the three fusion schemes outperform all multi-band baseline methods listed in Table II. This seems to imply that once the latent states between the beam SNR and CSI are aligned, the choice of fusion scheme has only a marginal impact on the final localization performance.

The training process of our NDF, constrained by the loss of signal reconstruction in Equation (47), enables the latent dynamic ODE module to effectively acquire a set of parameters that accurately represents the wireless dynamics of both bands. The latent dynamic ODE then regresses two sequences of latent variables in the queued timestamps, initialized by $\mathbf{z}_0^b$ and $\mathbf{z}_0^c$ for beam SNR and CSI, respectively. We assume that this well-trained latent dynamic ODE can regress sequences primarily conditioned by the initial state of each band while incorporating multi-band information, and this would be the reason the choice of fusion scheme did not make significant difference on performance, and even achieved reasonable performance with relatively simple fusion scheme like pairwise interaction.

G. Generalization under Temporal Split

Under **temporal split**, sequences in the training and test sets are distinctly separated in the temporal domain such that they do not intertwine. This separation allows us to effectively assess the generalization capability on future Wi-Fi measurements. We consider various training data ratios of $s\% = 20\%, 40\%, 60\%, 80\%$ corresponding to, respectively, $(20:80, 40:60, 60:40, 80:20)$ training-test data split ratios. For better use of training data when the training data size is small, we segment the training data into 5-sec overlapping sequences using a step size of 1 second.

Table V shows the localization errors for various choices of s. It is seen that a temporal training-test split ratio of $60:40$ provides the best localization performance in all metrics evaluated. The NDF with a temporal training-test split ratio of $80:20$ results in less accurate performance compared to the random training-test split ratio of $80:10:10$ in Table II. For example, the mean localization error increases from 26.3 cm under the random split to 50.1 cm under the temporal split. Such degradation is anticipated due to the temporal fluctuation in Wi-Fi measurements, which may stem from channel instability over time. Compared to baselines, our NDF also works better for all selected split ratios s in general.

H. Generalization under Coordinate Split

Under **coordinate split**, we can test the generalization capability on test data collected from unseen positions. In this evaluation, we use Fréchet distance as the evaluation metric to highlight the extent to which the baselines and our NDF can replicate the shape and position of the unseen corner. Fréchet distance is a measure of similarity between curves considering the location and ordering of the points along the curves. Fréchet distance $F(A, B)$ for two normalized curves $A = \{a_t\}_{t=0}^1, B = \{b_t\}_{t=0}^1$ is defined as

$$F(A,B) = \inf_{f:[0,1]\to[0,1]} \max_{t\in[0,1]} \|A(t) - B(f(t))\|, \qquad (51)$$

where f is an arbitrary reparametrization function. Table VI shows the localization errors of our NDF and baselines for the unseen regions, and Fig. 13 illustrates the estimated trajectories (in red dots) alongside the ground truth trajectories in the upper right corner, with training trajectories plotted in blue. As shown in Fig. 13 (a) and (b), the frame-to-frame fusion baseline methods (LinearInt and NearestInt) fail to leverage latent dynamics, resulting in scattered estimated trajectories. In contrast, the sequence-to-sequence baseline methods, specifically RNN-Decay fusion and RNN-Δ fusion in Fig. 13 (c) and (d), demonstrate improved alignment between the estimated red trajectories and the training blue trajectories. Fig. 13 (e) presents the NDF results, clearly showing that the estimated trajectories effectively complement the training trajectories, closely mirroring the ground truth rectangular trajectories.

I. Training on a more complex trajectory

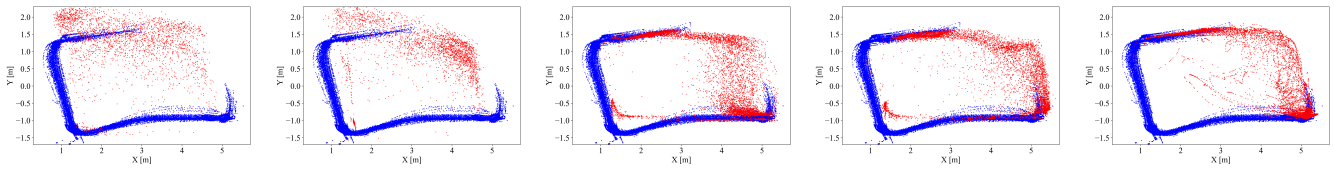
In this evaluation, we train the baseline and NDF models using multi-band Wi-Fi data collected from a checkerboard trajectory of Fig. 14a and then evaluate their localization performance using multi-band Wi-Fi data collected from a cross trajectory of Fig. 14b.

Fig. 15 shows the localization performance from two baseline (NearestInt and RNN-Decay) methods and the proposed NDF framework using the multi-band Wi-Fi data, where blue dots denote the ground truth location of the cross trajectory, while red dots denote estimated locations using the multi-band Wi-Fi data.

- **NearestInt**: a frame-to-frame fusion baseline, produces most of the position estimates around the center of the trajectory and fails to capture the cross trajectory.
- **RNN-Decay**: a sequence-to-sequence fusion baseline, shows improved performance than NearestInt, following more along the cross trajectory.
- **NDF**: on the other hand, demonstrates more clustered estimates around the cross trajectory, especially around the corners and the left side of the trajectory. We zoom in one corner for a better visualization.

TABLE V: Localization error (m) under temporal split as a function of training data ratio $s\%$ for multi-band cases.

	20%			**40%**			**60%**			**80%**		
	Mean	Median	CDF@0.9	Mean	Median	CDF@0.9	Mean	Median	CDF@0.9	Mean	Median	CDF@0.9
LinearInt	1.36	0.933	3.42	1.40	0.856	4.44	0.968	0.749	1.99	1.00	0.759	2.13
NearestInt	1.39	0.989	3.24	1.40	0.881	4.37	0.981	0.763	2.01	1.01	0.760	2.19
RNN-Decay	0.743	0.266	2.38	0.578	0.232	**1.65**	0.562	0.214	1.62	0.639	0.238	1.99
RNN-Δ	0.671	0.316	**1.88**	0.738	0.284	2.27	0.580	0.233	1.72	0.734	0.332	2.07
NDF (ours)	**0.647**	**0.250**	2.07	0.749	**0.183**	3.81	**0.454**	**0.169**	**0.981**	**0.501**	**0.183**	**1.17**

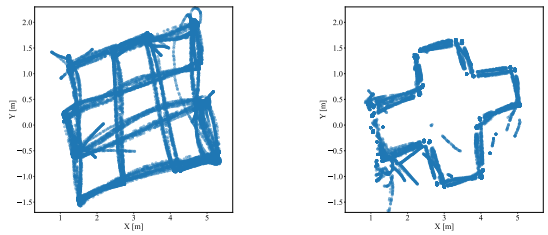


(a) LinearInt fusion (b) NearestInt fusion (c) RNN-Decay fusion (d) RNN-Δ fusion (e) NDF

Fig. 13: Generalization capability to unseen locations (upper right corner) is illustrated with training trajectories in blue. Ideally, the estimated trajectories in red should complement the blue training trajectories to form a complete rectangular path.

TABLE VI: Fréchet distance for unseen locations.

	Mean	Median	CDF@0.9
LinearInt	3.14	3.23	3.90
NearestInt	3.12	3.29	3.73
RNN-Decay	3.60	3.58	3.96
RNN-Δ	3.36	3.36	3.74
NDF (ours)	**2.55**	**2.57**	**2.97**



(a) Checkerboard trajectory (b) Cross trajectory

Fig. 14: **Two complex trajectories**: a checkerboard trajectory (a) and a cross trajectory (b) to evaluate the generalization capabilities of the considered baseline methods and the proposed NDF framework.

Table VII shows the Fréchet distance between curves with every 100 points in ground truth and estimations for the selected baselines and our NDF. The NDF achieves lower mean and median distance than other baselines, so thus we also qualitatively prove that our NDF reproduces reasonable shape of the trajectory better.

TABLE VII: Fréchet distance between curves with every 100 points in ground truth and estimations.

	Mean	Median	CDF@0.9
NearestInt	2.07	2.05	**2.67**
RNN-Decay	2.20	2.16	2.93
NDF (ours)	**1.92**	**1.90**	2.72

J. Latent Space Visualization

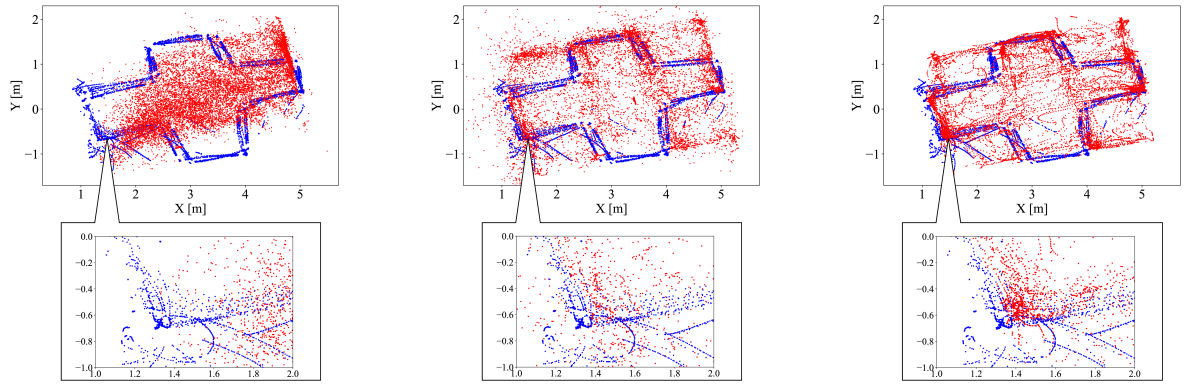
As shown in Fig. 16 (a), we identify 8 local regions along the trajectory. We gather all data frames with their corresponding ground truth locations from the same region and map them into latent states using selected baseline methods and the proposed NDF method. These high-dimensional latent states are then visualized by projecting them onto a 2D plane using t-distributed Stochastic Neighbor Embedding (t-SNE).

Fig. 16 (b) presents the t-SNE visualization results for the single-band baseline (beam SNR-based DDND), the RNN-Decay fusion baseline, and the proposed NDF method. The sequence-to-sequence RNN-Decay fusion baseline exhibits much clearer separation compared to the single-band beam SNR-based DDND, highlighting the advantages of utilizing multi-band Wi-Fi channel measurements in our experiment. Nonetheless, it is seen that the latent states of regions 2 and 6 overlap within the upper right cluster.

In contrast, Fig. 16b demonstrates that our NDF learns a compact and well-separated representation in the latent space, with denser latent distributions for each region. These results further suggest that the low-dimensional latent space effectively preserves the trajectory geometry within the NDF framework. Notably, each latent cluster is spatially connected to the edge of its adjacent cluster, creating a continuous latent space that seamlessly transitions from one region to another.

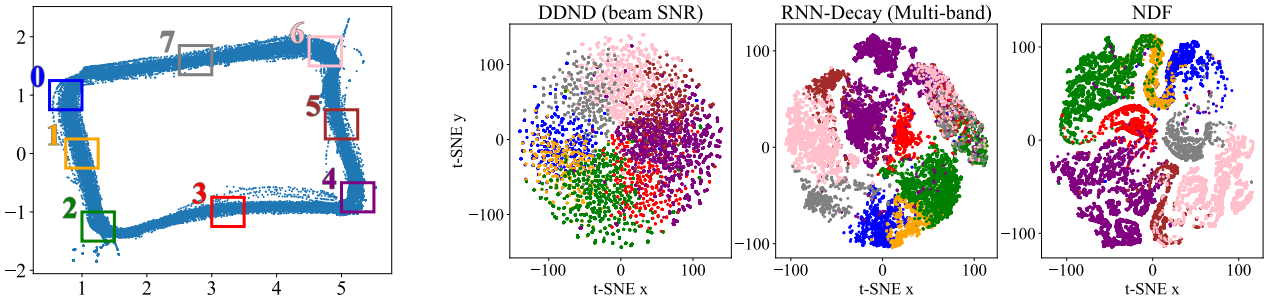
VI. Conclusion

In this paper, we introduce the NDF framework, which utilizes asynchronous multi-band Wi-Fi channel measurements to estimate trajectories in a continuous-time manner. This is achieved through a multiple-encoder, multiple-decoder architecture that aligns latent states across different input sequences



(a) NearestInt (b) RNN-Decay (c) NDF (ours)

Fig. 15: **Training on Checkerboard, Testing on Cross Trajectory**. Blue dots denote the ground truth location of the cross trajectory, while red dots denote estimated locations from two baseline methods (a) and (b) and the proposed NDF framework (c) using the multi-band Wi-Fi data.



(a) Selected local regions for latent space inspection.

(b) Latent states from selected regions projected on 2D plane by t-SNE.

Fig. 16: Visualization of learned latent states from selected local regions, with matching colors between the local regions and their corresponding latent states.

and fuses them for trajectory estimation. Latent state alignment is facilitated by a learnable ODE model and the initial latent conditions from the encoders. Evaluated with real-world multi-band Wi-Fi data, the NDF framework demonstrates significant performance enhancements compared to a comprehensive set of single-band and multi-band baseline methods: our NDF achieves lower localization error in average of about 26 cm than baselines with errors exceeding 50 cm.

Our NDF framework can be extended to other types of downstream tasks, particularly those continuous tracking a subject's behavior over time, such as activity recognition, gait-based identification, vital sign monitoring, and behavior analysis.

Appendix A
CSI Calibration and Embedding

The extracted CSI suffers from both magnitude and phase offsets, including carrier frequency offset (CFO), sample time offset (STO) [30]–[35]. We choose the SpotFi [30] calibration method to eliminate the linear phase offset caused by STO and CSI conjugate multiplication to cancel out packet-wise random phase offset and improve the stability of the waveform. The calibration is performed packet-wise and antenna-wise at the receiving side. For each receiving antenna, we first unwrap the CSI phase, then we obtain the best linear fit of the unwrapped phase as

$$[\hat{\tau}_{i,n}, \hat{\beta}_{i,n}] = \arg\min_{\tau,\beta} \sum_{j,k=1}^{N_{\mathrm{R}x}, N_{\mathrm{s}}} (\psi_n(i,j,k) - 2\pi f_\delta (k-1)\tau + \beta)^2, \tag{52}$$

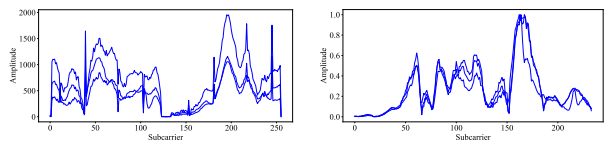
where $\psi_n(i,j,k) = \angle C_n(i,j,k)$ is the unwrapped phase of the n-th packet from transmitting antenna i and receiving antenna j at subcarrier k, and f_δ is the frequency spacing between two adjacent subcarriers. We obtain the calibrated phase by subtracting phase offset as

$$\hat{\psi}_n(i,j,k) = \psi_n(i,j,k) - 2\pi f_\delta (k-1)\hat{\tau}_{i,n}. \tag{53}$$

We further perform CSI conjugate multiplication across receiving antennas to remove random phase fluctuation over packets [33]. This leads to the calibrated CSI element

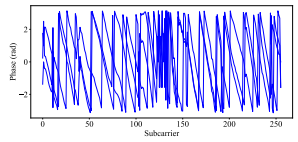
$$\tilde{C}_n(i,j,k) = C_n(i,j,k) C_n^*(i,j+1,k), \tag{54}$$

where $j = 1, \cdots, N_{\mathrm{R}x} - 1$, and $*$ represents the complex conjugate. Grouping all calibrated CSI elements $\tilde{C}_n(i,j,k)$ over transmitting antenna i, receiving antenna j, and subcarrier k, the calibrated CSI tensor is given by $\tilde{\mathbf{C}}_n \in \mathbb{C}^{N_{\mathrm{T}x} \times (N_{\mathrm{R}x}-1) \times N_{\mathrm{s}}}$. We present below additional plots of the CSI measurements before calibration (Fig. 17a and 17c) and

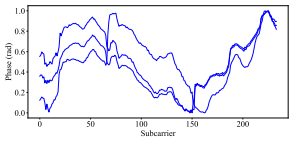


(a) Amp. before calibration

(b) Amp. after calibration



(c) Phase before calibration



(d) Phase after calibration

Fig. 17: Visualization of CSI measurements before and after calibration.

after calibration (Fig. 17b and 17d). Each plot includes three lines representing three consecutive CSI frames over time. The plots demonstrate that the calibration process effectively sanitizes the measurements, resulting in significantly smoother and more consistent waveforms along the subcarrier axis over time.

To balance between the two input (CSI and beam SNR) sequences, we employ a pretrained convolutional autoencoder (CAE) to compressed CSI tensor $\tilde{\mathbf{C}}_n$ into an embedding vector $\mathbf{c}_n \in \mathbb{R}^{M_c \times 1}$ with the following steps:

- **Complex-to-Real Conversion**: we convert the complex-valued $\tilde{\mathbf{C}}_n$ into a real-valued matrix $\mathbf{C}_n^f \in \mathbb{R}^{N_{\mathrm{Tx}}(N_{\mathrm{Rx}}-1)N_s \times 4}$. This is achieved by by splitting each element into four parts: real, imaginary, phase, and magnitude. Each of these parts is then vectorized into a 1D vector and putting these four vectors together yields $\mathbf{C}_n^f$.
- **Embedding from Autoencoder**: The real-valued matrix $\mathbf{C}_n^f$ is fed to the CAE as $\mathbf{c}_n = \mathcal{E}_{\boldsymbol{\theta}_{\mathrm{AE}}^e}(\mathbf{C}_n^f)$ and $\hat{\mathbf{C}}_n^f = \mathcal{D}_{\boldsymbol{\theta}_{\mathrm{AE}}^d}(\mathbf{c}_n)$, where $\mathbf{c}_n$ is the CSI embedding vector, $\mathcal{E}_{\boldsymbol{\theta}_{\mathrm{AE}}^e}$ and $\mathcal{D}_{\boldsymbol{\theta}_{\mathrm{AE}}^d}$ represent the encoder and decoder of CAE, respectively, and $\hat{\mathbf{C}}_n^f$ is the reconstructed real-valued CSI matrix at the decoder output. The CAE is pretrained by minimizing the reconstruction error between $\mathbf{C}_n^f$ and $\hat{\mathbf{C}}_n^f$.

Appendix B
LSTM Update Step

Given the measurement $\mathbf{s}_n$ at time step n and the auxiliary variable $\tilde{\mathbf{h}}_n$, one can use a standard LSTM unit to update the latent variable $\mathbf{h}_n = \mathcal{R}(\tilde{\mathbf{h}}_n, \mathbf{s}_n; \boldsymbol{\theta}), \quad n = 0, 1, \cdots, N$, where $\mathcal{R}(\cdot, \cdot|\boldsymbol{\theta})$ is implemented with the following process (with abuse of notation)

$$\tilde{\mathbf{c}}_n = \tanh\left(\mathbf{W}_{rc}\mathbf{s}_n + \mathbf{W}_{hc}\tilde{\mathbf{h}}_n + \mathbf{b}_c\right), \tag{55}$$

$$\mathbf{f}_n = \sigma\left(\mathbf{W}_{rf}\mathbf{s}_n + \mathbf{W}_{hf}\tilde{\mathbf{h}}_n + \mathbf{b}_f\right), \tag{56}$$

$$\mathbf{i}_n = \sigma\left(\mathbf{W}_{ri}\mathbf{s}_n + \mathbf{W}_{hi}\tilde{\mathbf{h}}_n + \mathbf{b}_i\right). \tag{57}$$

The above process consists of three *gates*:

- a memory gate of (55) uses the tanh function to combine the auxiliary hidden state $\tilde{\mathbf{h}}_n$ and the current input $\mathbf{s}_n$ into a value range of $(-1, 1)$.
- a forget gate of (56) also acts on $(\tilde{\mathbf{h}}_n, \mathbf{s}_n)$ but compresses the value into $(0, 1)$ with the sigmoid function $\sigma(\cdot)$ to determine how much old memory should be retained.
- an input gate of (57) compresses $(\tilde{\mathbf{h}}_n, \mathbf{s}_n)$ into another value between 0 and 1 and decides how much information we should take from the new input $\mathbf{s}_n$,

along with weight matrices $\mathbf{W}_{rc/rf/ri/hc/hf/hi}$ and bias terms $\mathbf{b}_{c/f/i}$. Then new hidden state $\mathbf{h}_n$ is updated as $\mathbf{h}_n = \tanh(\hat{\mathbf{c}}_n) \odot \mathbf{o}_n$, where the new memory variable $\hat{\mathbf{c}}_n$ updates its "old" memory $\hat{\mathbf{c}}_{n-1}$ passing through the "current" forget gate output $\mathbf{f}_n$ and adds new memory cell $\tilde{\mathbf{c}}_n$ weighted by the "current" input gate output $\mathbf{i}_n$: $\hat{\mathbf{c}}_n = \mathbf{f}_n \odot \hat{\mathbf{c}}_{n-1} + \mathbf{i}_n \odot \tilde{\mathbf{c}}_n$, and the output gate $\mathbf{o}_n$ is computed as

$$\mathbf{o}_n = \sigma\left(\mathbf{W}_{ro}\mathbf{s}_n + \mathbf{W}_{ho}\tilde{\mathbf{h}}_n + \mathbf{W}_{co} \odot \hat{\mathbf{c}}_n + \mathbf{b}_o\right). \tag{58}$$

It is seen that the parameters $\boldsymbol{\theta}$ in the LSTM update step is given as $\boldsymbol{\theta} = \{\mathbf{W}_{rc/rf/ri/hc/hf/hi/ro/ho/co}, \mathbf{b}_{c/f/i/o}\}$.

Appendix C
General Derivation of ELBO

Evidence lower bound, or ELBO, is a lower bound on the log-likelihood of observed data. We first express the log-likelihood of the input data $\mathbf{x}$ as

$$\begin{aligned}\log p(\mathbf{x}) &= \log p(\mathbf{x}) \int q(\mathbf{z}|\mathbf{x}) d\mathbf{z} \\ &= \int q(\mathbf{z}|\mathbf{x}) \left(\log \frac{q(\mathbf{z}|\mathbf{x})}{p(\mathbf{z}|\mathbf{x})} + \log \frac{p(\mathbf{x}, \mathbf{z})}{q(\mathbf{z}|\mathbf{x})}\right) d\mathbf{z} \\ &= D_{\mathrm{KL}}[q(\mathbf{z}|\mathbf{x})||p(\mathbf{z}|\mathbf{x})] + \int q(\mathbf{z}|\mathbf{x}) \log \frac{p(\mathbf{x}, \mathbf{z})}{q(\mathbf{z}|\mathbf{x})} d\mathbf{z},\end{aligned} \tag{59}$$

where $D_{\mathrm{KL}}[\cdot||\cdot]$ is the Kullback–Leibler (KL) divergence between two given distributions. Given that $D_{\mathrm{KL}}[q(\mathbf{z}|\mathbf{x})||p(\mathbf{z}|\mathbf{x})] \geq 0$, (59) follows as [44]

$$\begin{aligned}\log p(\mathbf{x}) &\geq \int q(\mathbf{z}|\mathbf{x}) \log q(\mathbf{x}|\mathbf{z}) d\mathbf{z} + \int q(\mathbf{z}|\mathbf{x}) \log \frac{p(\mathbf{z})}{q(\mathbf{z}|\mathbf{x})} d\mathbf{z} \\ &= \mathbb{E}_{q(\mathbf{z}|\mathbf{x})}\left[\log q(\mathbf{x}|\mathbf{z})\right] - D_{\mathrm{KL}}[q(\mathbf{z}|\mathbf{x})||p(\mathbf{z})].\end{aligned} \tag{60}$$

We extend the above lower bound to the case where two inputs $\mathbf{x}$ and $\mathbf{y}$ with corresponding $\mathbf{z}_x$ and $\mathbf{z}_y$ as

$$\mathbb{E}_{q(\mathbf{z}_x, \mathbf{z}_y|\mathbf{x}, \mathbf{y})}\left[\log q(\mathbf{x}, \mathbf{y}|\mathbf{z}_x, \mathbf{z}_y)\right] - D_{\mathrm{KL}}[q(\mathbf{z}_x, \mathbf{z}_y|\mathbf{x}, \mathbf{y})||p(\mathbf{z}_x, \mathbf{z}_y)].$$

The KL divergence term can be decomposed to the sum of two KL divergence terms by using the independent assumptions between $\mathbf{x}$ and $\mathbf{y}$ and between $\mathbf{z}_x$ and $\mathbf{z}_y$,

$$\begin{aligned}&D_{\mathrm{KL}}[q(\mathbf{z}_x, \mathbf{z}_y|\mathbf{x}, \mathbf{y})||p(\mathbf{z}_x, \mathbf{z}_y)] \\ &= \int\int q(\mathbf{z}_x|\mathbf{x}) q(\mathbf{z}_y|\mathbf{y}) \left(\log \frac{q(\mathbf{z}_x|\mathbf{x})}{p(\mathbf{z}_x)} + \log \frac{q(\mathbf{z}_y|\mathbf{y})}{p(\mathbf{z}_y)}\right) d\mathbf{z}_x d\mathbf{z}_y \\ &= \underbrace{\int q(\mathbf{z}_y|\mathbf{y}) d\mathbf{z}_y}_{=1} \int q(\mathbf{z}_x|\mathbf{x}) \log \frac{q(\mathbf{z}_x|\mathbf{x})}{p(\mathbf{z}_x)} d\mathbf{z}_x \\ &\quad + \underbrace{\int q(\mathbf{z}_x|\mathbf{x}) d\mathbf{z}_x}_{=1} \int q(\mathbf{z}_y|\mathbf{y}) \log \frac{q(\mathbf{z}_y|\mathbf{y})}{p(\mathbf{z}_y)} d\mathbf{z}_y \\ &= D_{\mathrm{KL}}[q(\mathbf{z}_x|\mathbf{x})||p(\mathbf{z}_x)] + D_{\mathrm{KL}}[q(\mathbf{z}_y|\mathbf{y})||p(\mathbf{z}_y)].\end{aligned} \tag{61}$$

REFERENCES

[1] Sorachi Kato et al., "Object trajectory estimation with multi-band Wi-Fi neural dynamic fusion," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2024, pp. 13261–13265.

[2] Francesca Meneghello et al., "Toward integrated sensing and communications in IEEE 802.11bf Wi-Fi networks," *IEEE Communications Magazine*, vol. 61, no. 7, pp. 128–133, 2023.

[3] Steve Blandino et al., "IEEE 802.11bf DMG sensing: Enabling high-resolution mmWave Wi-Fi sensing," *IEEE Open Journal of Vehicular Technology*, vol. 4, pp. 342–355, 2023.

[4] Cheng Chen et al., "Wi-Fi sensing based on IEEE 802.11bf," *IEEE Communications Magazine*, vol. 61, no. 1, pp. 121–127, 2023.

[5] Rui Du et al., "An overview on IEEE 802.11bf: WLAN sensing," *arXiv:2207.04859*, 2022.

[6] M. Youssef and A. Agrawala, "The Horus location determination system," *Wirel. Netw.*, vol. 14, no. 3, pp. 357–374, June 2008.

[7] Minh Tu Hoang et al., "Recurrent neural networks for accurate RSSI indoor localization," *IEEE Internet of Things Journal*, vol. 6, no. 6, pp. 10639–10651, Dec 2019.

[8] Hao Chen et al., "ConFi: Convolutional neural networks based indoor Wi-Fi localization using channel state information," *IEEE Access*, vol. 5, pp. 18066–18074, 2017.

[9] J. Ding and Y. Wang, "WiFi CSI-based human activity recognition using deep recurrent neural network," *IEEE Access*, vol. 7, pp. 174257–174269, 2019.

[10] Yan Li, Jie Yang, Shang-Ling Shih, Wan-Ting Shih, Chao-Kai Wen, and Shi Jin, "Efficient IoT devices localization through WI-fi CSI feature fusion and anomaly detection," *IEEE Internet Things J.*, vol. PP, no. 99, pp. 1–17, 2024.

[11] Roshan Ayyalasomayajula, Aditya Arun, Chenfeng Wu, Sanatan Sharma, Abhishek Rajkumar Sethi, Deepak Vasisht, and Dinesh Bharadia, "Deep learning based wireless localization for indoor navigation," in *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking*, Apr. 2020, number Article 17 in MobiCom '20, pp. 1–14.

[12] Toshiaki Koike-Akino et al., "Fingerprinting-based indoor localization with commercial mmwave WiFi: A deep learning approach," *IEEE Access*, vol. 8, pp. 84879–84892, 2020.

[13] Dolores Garcia et al., "POLAR: Passive object localization with IEEE 802.11ad using phased antenna arrays," in *IEEE Conference on Computer Communications*, July 2020, pp. 1838–1847.

[14] Jianyuan Yu et al., "Multi-band Wi-Fi sensing with matched feature granularity," *IEEE Internet of Things Journal*, vol. 9, no. 23, pp. 23810–23825, 2022.

[15] Alejandro Blanco et al., "Augmenting mmwave localization accuracy through sub-6 GHz on off-the-shelf devices," in *MobiSys*, 2022, pp. 477–490.

[16] Cristian J. Vaca-Rubio et al., "mmwave Wi-Fi trajectory estimation with continuous-time neural dynamic learning," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2023, pp. 1–5.

[17] Cristian J. Vaca-Rubio et al., "Object trajectory estimation with continuous-time neural dynamic learning of millimeter-wave Wi-Fi," *IEEE Journal of Selected Topics in Signal Processing*, vol. 17, 2024.

[18] Lang Deng et al., "GaitFi: Robust device-free human identification via WiFi and vision multimodal learning," *IEEE Internet of Things Journal*, vol. 10, no. 1, pp. 625–636, Jan. 2023.

[19] Jianyuan Yu et al., "Multi-modal recurrent fusion for indoor localization," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 5083–5087.

[20] Afaz Uddin Ahmed et al., "Multi-radio data fusion for indoor localization using Bluetooth and WiFi," in *PECCS*, 2019, pp. 13–24.

[21] Xiaochao Dang et al., "A novel passive indoor localization method by fusion CSI amplitude and phase information," *Sensors*, vol. 19, no. 4, pp. 875, 2019.

[22] Y. Xie, Z. Li, and M. Li, "Precise power delay profiling with commodity WiFi," in *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*, 2015, pp. 53–64.

[23] Jie Xiong, Karthikeyan Sundaresan, and Kyle Jamieson, "ToneTrack: Leveraging frequency-agile radios for time-based indoor wireless localization," in *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*. Sept. 2015, ACM.

[24] D. Vasisht, S. Kumar, and D. Katabi, "Decimeter-level localization with a single WiFi access point," in *13th USENIX Symposium on Networked Systems Design and Implementation (NSDI 16)*, Santa Clara, CA, Mar. 2016, pp. 165–178.

[25] Hongzi Zhu, Yiwei Zhuo, Qinghao Liu, and Shan Chang, "π-splicer: Perceiving accurate CSI phases with commodity WiFi devices," *IEEE Trans. Mob. Comput.*, vol. 17, no. 9, pp. 2155–2165, Sept. 2018.

[26] Nebojsa Maletic, Vladica Sark, Jesus Gutierrez, and Eckhard Grass, "Device localization using mmWave ranging with sub-6-assisted angle of arrival estimation," in *IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB)*. June 2018, IEEE.

[27] Jingzhi Hu, Dusit Niyato, and Jun Luo, "Cross-domain learning framework for tracking users in RIS-aided multi-band ISAC systems with sparse labeled data," *IEEE Journal on Selected Areas in Communications*, vol. 42, no. 10, pp. 2754–2768, 2024.

[28] R. Chen et al., "Neural ordinary differential equations," *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 31, 2018.

[29] Yulia Rubanova et al., "Latent ordinary differential equations for irregularly-sampled time series," in *Advances in Neural Information Processing Systems*, 2019, vol. 32, pp. 5320–5330.

[30] Manikanta Kotaru et al., "SpotFi: Decimeter level localization using WiFi," in *ACM Conference on Special Interest Group on Data Communication*, Aug 2015, pp. 269–282.

[31] Xiaonan Guo et al., "WiFi-enabled smart human dynamics monitoring," in *ACM Conference on Embedded Network Sensor Systems*, 2017, pp. 1–13.

[32] Xuyu Wang et al., "PhaseBeat: Exploiting CSI phase data for vital sign monitoring with commodity WiFi devices," in *ICDCS*, June 2017, pp. 1230–1239.

[33] Youwei Zeng et al., "FullBreathe: Full human respiration detection exploiting complementarity of CSI phase and amplitude of WiFi signals," *IMWUT*, vol. 2, no. 3, pp. 1–19, Sept. 2018.

[34] Daqing Zhang et al., "Practical issues and challenges in CSI-based integrated sensing and communication," in *IEEE International Conference on Communications Workshops*, May 2022, pp. 836–841.

[35] Dongheng Zhang et al., "Calibrating phase offsets for commodity WiFi," *IEEE Systems Journal*, vol. 14, no. 1, pp. 661–664, Mar. 2020.

[36] Haotai Sun et al., "WiFi based fingerprinting positioning based on seq2seq model," *Sensors*, vol. 20, no. 13, pp. 3767, 2020.

[37] Jianyuan Yu et al., "Centimeter-level indoor localization using channel state information with recurrent neural networks," in *IEEE/ION Position, Location and Navigation Symposium (PLANS)*, 2020, pp. 1317–1323.

[38] S Hochreiter and J Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[39] Kyunghyun Cho et al., "Learning phrase representations using RNN Encoder-Decoder for statistical machine translation," *arXiv:1406.1078*, June 2014.

[40] Jacob Kelly et al., "Learning differential equations that are easy to solve," *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, pp. 4370–4380, 2020.

[41] Chris Finlay et al., "How to train your neural ODE: the world of Jacobian and kinetic regularization," in *International Conference on Machine Learning (ICML)*, 2020, pp. 3154–3164.

[42] Aiqing Zhu et al., "On numerical integration in neural ordinary differential equations," in *International Conference on Machine Learning (ICML)*, 2022, pp. 27527–27547.

[43] Ho Huu Nghia Nguyen et al., "Improving neural ordinary differential equations with Nesterov's accelerated gradient method," *NeurIPS*, vol. 35, pp. 7712–7726, 2022.

[44] Diederik P Kingma and Max Welling, "Auto-Encoding variational Bayes," *arXiv preprint arXiv:1312.6114*, 2014.

[45] F. Gringoli et al., "Free your CSI: A channel state information extraction platform for modern Wi-Fi chipsets," in *WiNTECH*, 2019, pp. 21–28.

[46] D. Steinmetzer, D. Wegemer, and M. Hollick, "Talon tools: The framework for practical IEEE 802.11ad research," in *Available: https://seemoo.de/talon-tools/*, 2018.

[47] Guillermo Bielsa et al., "Indoor localization using commercial off-the-shelf 60 GHz access points," in *IEEE Conference on Computer Communications*, 2018, pp. 2384–2392.

[48] Takuya Akiba et al., "Optuna: A next-generation hyperparameter optimization framework," in *SIGKDD*, 2019, p. 2623–2631.

[49] Leslie N Smith and Nicholay Topin, "Super-convergence: Very fast training of neural networks using large learning rates," in *Artificial intelligence and machine learning for multi-domain operations applications*, 2019, vol. 11006, pp. 369–386.

Sorachi Kato received B.E. and M.E. degrees from Osaka University, Japan, in 2021 and 2023, respectively. He is currently pursuing his Ph.D. degree in the Graduate School of Information Science and Technology, Osaka University. He is a research fellow (DC1) of Japan Society for the Promotion of Science from 2023. From 2023 to 2024, he was an intern at Mitsubishi Electric Research Labs. (MERL) working with the computational sensing team. His research interests are in the areas of RF sensing and deep neural signal processing.

Pu (Perry) Wang (Senior Member, IEEE) received the Ph.D. degree in Electrical Engineering from the Stevens Institute of Technology, Hoboken, NJ, USA, in 2011.

He is a Senior Principal Research Scientist at the Mitsubishi Electric Research Laboratories (MERL), Cambridge, MA, where he was an intern in the summer of 2010. Before rejoining MERL in 2016, he was a Research Scientist at Schlumberger-Doll Research, Cambridge, MA, where he contributed to the development and commercialization of logging-while-drilling Acoustics/NMR products. His research focuses on radar perception, wireless sensing, signal processing, Bayesian inference, deep learning, and their industrial applications.

Dr. Wang received the IEEE Jack Neubauer Memorial Award from the IEEE Vehicular Technology Society in 2013 and was recognized as a Distinguished Speaker by the Society of Petrophysicists and Well Log Analysts (SPWLA) in 2017. He served as an Associate Editor (2018-2022) and later a Senior Area Editor (SAE) (2022-Present) for *IEEE Signal Processing Letters* and was a Guest Editor for *IEEE Signal Processing Magazine*, *IEEE Journal of Selected Topics in Signal Processing*, *IEEE Communications Standards Magazine*, and *IEEE Sensors Journal*. He is an active member of the IEEE SPS Technical Committees on Signal Processing Theory and Methods (SPTM), Machine Learning for Signal Processing (MLSP), and Applied Signal Processing Systems (ASPS) and serves as a Voting Member of the IEEE 802 Standards Association. He has also organized special sessions and satellite workshops at ICASSP and SAM and delivered tutorials at the IEEE Radar Conference and GLOBECOM.

Toshiaki Koike-Akino (Senior Member, IEEE) received the Ph.D. degree from Kyoto University, Kyoto, Japan, in 2005. During 2006-2010, he was a Postdoctoral Researcher at Harvard University, Cambridge, MA, USA, and he is currently Distinguished Research Scientist at MERL, Cambridge, MA, USA. He was the recipient of the 2008 Ericsson Young Scientist Award, the IEEE GLOBECOM'08 Best Paper Award, the 24th TELECOM System Technology Encouragement Award, and the IEEE GLOBECOM'09 Best Paper Award. He is a Fellow of Optica (formerly OSA).

Takuya Fujihashi received B.E. and M.S. degree from Shizuoka University, Japan, in 2012 and 2013, respectively. In 2016, he received a Ph.D. degree from the Graduate School of Information Science and Technology, Osaka University, Japan. He is currently an assistant professor at the Graduate School of Information Science and Technology, Osaka University since April 2019. He was a research fellow (PD) of the Japan Society for the Promotion of Science in 2016. From 2014 to 2016, he was a research fellow (DC1) of the Japan Society for the Promotion of Science. From 2014 to 2015, he was an intern at Mitsubishi Electric Research Labs. (MERL) working with the Electronics and Communications group. His research interests are in the area of video compression and communication, with a focus on immersive video coding and streaming.

Hassan Mansour (Senior Member, IEEE) received the B.E. degree in computer and communications engineering from the American University of Beirut, Beirut, Lebanon, in 2003, and the M.A.Sc. degree in electrical and computer engineering and the Ph.D. degree in electrical and computer engineering from The University of British Columbia, Vancouver, BC, Canada, in 2005 and 2009, respectively. Between January 2010 and January 2013, he was a Postdoctoral Research Fellow with the Department of Computer Science, the Mathematics Department, and the Department of Earth, Ocean, and Atmospheric Sciences, The University of British Columbia. He is a Senior Principal Research Scientist and Computational Sensing Team Leader with Mitsubishi Electric Research Laboratories, Cambridge, MA, USA. His research interests are in inverse problems, machine learning, compressed sensing, sparse signal reconstruction, image enhancement, and scalable video compression and transmission. His current research is focused on the design of efficient acquisition schemes and reconstruction algorithms for natural images, radar sensing, video analytics, and seismic imaging. Dr. Mansour is a member of the IEEE Signal Processing Society. He has also served on the Computational Imaging Technical Committee and the Sensor Array and Multichannel Technical Committee. He was an Associate Editor for the IEEE Transactions on Signal Processing between 2018 and 2022. He is currently an Associate Editor for the IEEE Transactions on Computational Imaging.

Petros T. Boufounos (Fellow, IEEE) is a Distinguished Research Scientist, a Deputy Director and the Computational Sensing Senior Team Leader at Mitsubishi Electric Research Laboratories (MERL). Dr. Boufounos completed his undergraduate and graduate studies at MIT. He received the S.B. degree in Economics in 2000, the S.B. and M.Eng. degrees in Electrical Engineering and Computer Science (EECS) in 2002, and the Sc.D. degree in EECS in 2006. Between September 2006 and December 2008, he was a postdoctoral associate with the Digital Signal Processing Group at Rice University. Dr. Boufounos joined MERL in January 2009, where he has been heading the Computational Sensing Team since 2016.

Dr. Boufounos' immediate research focus includes signal acquisition and processing, computational sensing, inverse problems, quantization, and data representations. He is also interested in how signal acquisition interacts with other fields that use sensing extensively, such as machine learning, robotics, and dynamical system theory. He has over 40 patents granted and more than 10 pending, and more that 100 peer reviewed journal and conference publications in these topics. Dr. Boufounos was the general co-chair of the ICASSP 2023 organizing committee and is currently a regional director-at-large in the IEEE Signal Processing Society's Board of Governors. He has also served as an Area Editor and a Senior Area Editor for the IEEE Signal Processing Letters, an AE for IEEE Transactions on Computational Imaging, and as a member of the SigPort editorial board and the IEEE Signal Processing Society Theory and Methods technical committee. Dr. Boufounos is an IEEE Fellow and an IEEE SPS Distinguished Lecturer for 2019-2020.