RMMNet: Deep Memory Aided Extended Object Tracking via High-Order Markovian Modeling and State Decoupling

Zhixing Wang, Le Zheng, *Senior Member, IEEE,* Shi Yan, Ruud J. G. van Sloun, *Member, IEEE,* Nir Shlezinger, *Senior Member, IEEE,* and Yonina C. Eldar, *Fellow, IEEE*

***Abstract*—Grounded in Bayesian filtering (BF), random matrix-based extended object tracking (EOT) offers an efficient framework for the joint estimation of a target's kinematic state and extension. However, the performance of these methods is fundamentally constrained by the first-order Markov assumption, which often fails to capture the high-order Markovian dynamics prevalent in real-world scenarios. To overcome this limitation, we introduce RMMNet, a novel EOT method that integrates a deep memory-aided recursive neural network filter within a BF framework, enabling robust tracking of targets that exhibit high-order Markovian dynamics. Initially, we propose an approximate model under a high-order Markovian assumption and derive the implementation of its BF framework. Thereafter, Gaussian approximation and moment matching are employed to derive the analytical formulations for the proposed BF framework. Finally, based on the closed-form formulations, we design the architecture of RMMNet as an end-to-end trainable recursive neural network-based filter. Experimental results on both simulated and real-world data demonstrate that our method outperforms traditional EOT approaches and other state-of-the-art deep learning methods.**

***Index Terms*—Bayesian filtering, extended object tracking, random matrix, neural network.**

I. Introduction

With the improvement of sensor resolution, such as ultra-wideband radar, a target is resolved into multiple scatterers and occupies multiple resolution cells [1]. Therefore, it is possible to jointly estimate both the kinematic state and the extension of the target. This methodology, termed *extended object tracking (EOT)*, has seen rapid development over the past decade. It enhances the description of the target by incorporating its size, shape, and orientation, thus providing a more comprehensive characterization of the target. Consequently, EOT has found widespread applications in fields such as autonomous driving [2], [3], maritime surveillance [4], and aircraft formation tracking [5].

This work was supported in part by the National Natural Science Foundation of China No. 62388102 and No. 62401059; (*Corresponding author: Le Zheng*).

Zhixing Wang is with the School of Information and Electronics, Beijing Institute of Technology, Beijing 100081, China (e-mail: wangwangzx@bit.edu.cn). Le Zheng is with the Zhengzhou Research Institute, Beijing Institute of Technology, Zhengzhou 450000, China, and also with the School of Information and Electronics, Beijing Institute of Technology, Beijing 100081, China (e-mail: le.zheng.cn@gmail.com).

Shi Yan is with the School of Automation, Northwestern Polytechnical University, Xi'an 710072, China (e-mail: yanshi@mail.nwpu.edu.cn).

Ruud J. G. van Sloun is with the Department of Electrical Engineering, Eindhoven University of Technology, 5612 Eindhoven, The Netherlands (e-mail: r.j.g.v.sloun@tue.nl).

Nir Shlezinger is with the School of Electrical and Computer Engineering, Ben-Gurion University of the Negev, Beer Sheva 84105, Israel (e-mail: nirshl@bgu.ac.il).

Yonina C. Eldar is with the Department of Computer Science and Applied Mathematics, Weizmann Institute of Science, Rehovot 7610001, Israel (e-mail: yonina.eldar@weizmann.ac.il).

Mainstream EOT methods based on the Bayesian filtering (BF) framework can be categorized into three distinct types: 1) random matrix models (RMM); 2) random hypersurface models (RHM); and 3) multiplicative error models (MEM). The RMM-based approach was first introduced in [6]. The target's extension is represented by a symmetric positive definite (SPD) matrix, which can be equivalently described as an ellipse, and is assumed to obey an inverse Wishart distribution. Bayesian inference is then employed to predict and update the probability densities of the state and the extension jointly. The RHM-based method for EOT was first introduced in [7], which assumes the measurement source lies on a scaled version of the extension boundaries, to develop Gaussian estimators for EOT with ellipses and free-form star-convex shapes. To maintain the conjugate Gaussian distribution during the evolution process, MEM-based methods introduce a multiplicative noise term in the measurement process [8]. Recently, the MEM-RBPF [9] decouples variables within the extension (i.e., orientation and size). It marginalizes the orientation via a particle filter to achieve accurate estimation with reduced computational cost. Generally, the MEM-based approaches make each scatterer equivalent to the centroid position plus the scaled object size.

The aforementioned EOT online optimization methods are model-based (MB) methods, namely, they all rely on prior models. Such approaches can theoretically guarantee optimal filtering accuracy when the prior model accurately matches the environment. However, the introduction of prior models also introduces additional parameters, which can result in higher filtering errors in practical applications due to model mismatch or parameter mismatch [10]. Furthermore, different EOT methods focus solely on redefining the measurement process to achieve better filtering accuracy, while neglecting improvements in the evolution processes of both state and extension (e.g., prediction errors caused by the first-order Markov approximation in the evolution process). Additionally, there exists a coupled relationship between the target's state and extension in practical scenarios. To achieve recursive processing, the methods mentioned above typically decouple the target's state and extension roughly during calculations, which significantly limits the accuracy of the filters.

With the rapid development of deep learning in recent years, an increasing number of deep neural networks (DNNs) have been proposed for real-world tasks. These data-based (DB) methods are typically able to capture the rich features and patterns present in offline data, thereby superseding MB methods in various complex tasks, such as time series analysis [11]. Benefiting from the powerful capabilities of recurrent neural network (RNN) architectures for nonlinear features and the ability to extract temporal contextual information, naive RNN and their derivatives, such as gated recurrent units (GRU) [12] and long short-term memory (LSTM)

networks [13], have been widely utilized in target tracking [14]. However, these methods often function as incomprehensible "black boxes" to users, with their internal mechanisms being difficult to express analytically, leading to a lack of interpretability. Moreover, since pure DNNs do not incorporate prior models into the prediction process, they exhibit a strong dependency on the quantity of data. Even simple functions are challenging for DNNs to learn and understand when data is sparse.

Briefly, the aforementioned challenges in EOT can be summarized in the following two points: 1) Traditional MB EOT methods often suffer from performance degradation due to model mismatch in real-world high-order Markovian scenarios and errors caused by insufficient state-extension decoupling; 2) Purly DB methods lack physical priors, which can lead to inefficient convergence and inference, limiting their practical effectiveness. Inspired by the recent success of hybrid MB-DB algorithms for point object tracking, e.g., [15]–[20] (see also survey in [21]), we propose a deep learning-aided Bayesian framework for EOT. Our approach utilizes DNNs to augment a MB BF, following the principles of MB deep learning [22]. This synergy aims to combine the interpretability and structure of the Bayesian framework with the powerful representation learning capabilities of DNNs.

Specifically, the contributions of this work are as follows:

- We propose a memory mechanism to approximate a high-order Markovian evolution model as a first-order Markov model, while decoupling the complex relationship between the state and extension.
- Based on this approximate model, we develop a memory-aided BF framework for the joint estimation of memory, state, and extension.
- We derive closed-form analytical formulations for the proposed BF framework using Gaussian approximation and moment matching, enabling efficient recursive computation while providing accurate estimation.
- We introduce RMMNet, a deep memory-aided recursive neural network filter for EOT. Derived directly from our proposed BF framework, RMMNet provides interpretability while leveraging the learning capabilities of a DNN to implement the memory mechanism, which enables the model to capture high-order Markovian dynamics from historical trajectories.
- Through carefully designed end-to-end self-supervised training methods, the RMMNet outperforms various state-of-the-art (SOTA) MB EOT and purely DB methods in benchmark experiments.

This work focuses on capturing temporal dependencies in high-order Markovian scenarios while decoupling the kinematic state and extension. The rest of this paper is organized as follows. Section II presents the fundamental theory of RMM approaches and our novel model for high-order Markovian dynamics. Section III derives our EOT algorithm by first approximating the proposed model as a first-order Markov model, then proving its closed-form expressions, and introducing RMMNet along with its end-to-end training approach. Section IV presents detailed comparative experimental results. Finally, Section V concludes the paper.

Notation: throughout this paper, the superscript $(\cdot)^{\mathrm{T}}$ is the transpose operation; $\mathrm{tr}(\cdot)$ and $|\cdot|$ are the trace and determinant of a matrix, respectively; $\mathbb{E}_{(\cdot|\cdot)}[\cdot]$ is the conditional expectation; $(\cdot)_{k|k-1}$ and $(\cdot)_{k|k}$ are the prediction and posterior of the state and extension, respectively; $d(\cdot)$ is the dimension of a vector, or the dimension of a matrix in its flattened form.

II. Problem Formulation

Building upon the methodology proposed in [23], in this section we formulate the RMM-based EOT framework, which divides the tracking process into two distinct phases of prediction and update. We first review existing modeling, based on which we introduce our high-order Markovian representation.

A. Existing Dynamic Model

We consider a single-target tracking scenario within a two-dimensional plane. At time step k, the kinematic state of the target's centroid is a $2d\times 1$ random vector $\mathbf{x}_k$ (specifically, for d=3, the vector encompasses position, velocity, and acceleration). The target's extension is a $d\times d$ SPD matrix $\mathbf{X}_k$, which encapsulates its size, shape and orientation. The state evolution model is constructed as

$$\mathbf{x}_k = f_k(\mathbf{x}_{k-1}) + \mathbf{w}_k,\ \ \mathbf{w}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{D}_k \otimes \mathbf{X}_k), \tag{1}$$

where, $f_k(\cdot)$ represents the nominal (non)linear state-transition function; the process noise $\mathbf{w}_k$ assumed to obey an independent zero-mean Gaussian distribution with covariance of $\mathbf{D}_k\otimes\mathbf{X}_k$, where "$\otimes$" is the Kronecker product; $\mathbf{D}_k{=}\sigma_k^2\hat{\mathbf{D}}_k$ is the covariance matrix of the process noise in the one-dimensional model, where σ_k^2 is the variance of the velocity noise, $\hat{\mathbf{D}}_k{=}(1{-}e^{-2\tau/\theta})$ $\mathrm{diag}([0,0,1])$ is the parametric matrix introduced by [6], with τ stands for the temporal interval between sequential frames.

Notably, in Eq. (1), the covariance of the process noise $\mathbf{w}_k$ is modeled to reflect dependencies on the target's extension $\mathbf{X}_k$. This differs from standard Kalman filtering (KF), where the covariance of process noise is usually a constant SPD matrix that represents fixed system properties [24].

Furthermore, compared to the standard KF, the state evolution in Eq. (1) includes additional descriptions of the extension evolution process as

$$p(\mathbf{X}_k|\mathbf{X}_{k-1}) = \mathcal{W}(\mathbf{X}_k; \delta_k, \phi_k(\mathbf{X}_{k-1})), \tag{2}$$

where $\phi_k(\cdot)$ represents the nominal (non)linear extension-transition functions; $\mathcal{W}(\mathbf{Y}; a, \mathbf{C})$ stands for the density of the Wishart distribution for a SPD random matrix $\mathbf{Y}$. Typically, $\phi_k(\cdot){=}\mathbf{A}_k(\cdot)\mathbf{A}_k^{\mathrm{T}}$ [23], where $\mathbf{A}_k$ can be used to describe changes in the target's extension, such as the orientation (if $\mathbf{A}_k$ is a rotation matrix) or the size (if $\mathbf{A}_k{=}\lambda\mathbf{I}_d$ and λ is a scaler parameter); δ_k describes the uncertainty of the extension transition [6], when δ_k increases, the precision in delineating the transition process is correspondingly enhanced, and vice versa.

B. Existing Measurement Model

We use the notation $\mathbf{Z}_k = \{\mathbf{z}_k^i\}_{i=1}^{n_k}$ to represent that the target has yielded n_k measurement vectors $\mathbf{z}_k^i$ during a single measurement. When employing observational sensors (such as radar), $\mathbf{z}_k^i$ stands for the scatterers generated by the target, which predominantly encapsulates the locational attributes of the target. The measurement process is characterized as

$$\mathbf{z}_k^i = h_k(\mathbf{x}_k) + \mathbf{v}_k^i,\ \ \mathbf{v}_k^i \sim \mathcal{N}(\mathbf{0}, \mathbf{B}_k\mathbf{X}_k\mathbf{B}_k^{\mathrm{T}}), \tag{3}$$

where $h_k(\cdot)$ represents the nominal (non)linear measurement function; $\mathbf{v}_k^i$ denotes the measurement noise corresponding to the i-th scatter, which is independent of $\mathbf{v}_k^r$ $(i, r = 1, \ldots, n_k, i \neq r)$, and obeys a zero-mean Gaussian distribution with covariance of $\mathbf{B}_k\mathbf{X}_k\mathbf{B}_k^{\mathrm{T}}$; $\mathbf{B}_k$ is equivalent to the observation distortion matrix as described in [23], where the distortions in the observation can originate from the geometric relationship between the target and the sensor, making $\mathbf{B}_k$ a function of $\mathbf{x}_k$.

The above measurement model assumes that all scatterers are generated by the centroid of the target and describes the dispersion of the point clouds through the covariance of the measurement noise. Additionally, the measurement error inherent to the sensor itself can be equivalently represented as the noise term in Eq. (3) [25]. When $n_k = 1$, the measurement process with an extended form reduces to the form used in point object tracking tasks, as employed by the standard KF. In this case, the covariance of the measurement noise is typically assumed to take a form $\mathbf{R}_k$.

In [6], it is assumed that the number of scatterers n_k is independent of both $\mathbf{x}_k$ and $\mathbf{X}_k$. Consequently, the likelihood of obtaining the measurement $\mathbf{Z}_k$, given the state, the extension, and the number of scatterers, takes the form of a product of n_k Gaussian distributions, namely,

$$p(\mathbf{Z}_k|n_k, \mathbf{x}_k, \mathbf{X}_k) = \prod_{i=1}^{n_k} \mathcal{N}(\mathbf{z}_k^i; h_k(\mathbf{x}_k), \mathbf{B}_k\mathbf{X}_k\mathbf{B}_k^{\mathrm{T}})$$
$$\propto \mathcal{N}(\widetilde{\mathbf{z}}_k; h_k(\mathbf{x}_k), \frac{\mathbf{B}_k\mathbf{X}_k\mathbf{B}_k^{\mathrm{T}}}{n_k}) \mathcal{W}(\widetilde{\mathbf{Z}}_k; n_k - 1, \mathbf{B}_k\mathbf{X}_k\mathbf{B}_k^{\mathrm{T}}), \quad (4)$$

where

$$\widetilde{\mathbf{z}}_k = \frac{1}{n_k}\sum_{i=1}^{n_k} \mathbf{z}_k^i, \quad \widetilde{\mathbf{Z}}_k = \sum_{i=1}^{n_k} (\mathbf{z}_k^i - \widetilde{\mathbf{z}}_k)(\mathbf{z}_k^i - \widetilde{\mathbf{z}}_k)^{\mathrm{T}}. \quad (5)$$

C. Flaw of RMM-Based EOT

The state and extension evolution models in Eqs. (1) and (2) are both based on a first-order Markov assumption, considering the transition between consecutive time steps. This is equivalent to the prevalent assumption within the state space model (SSM) that the contribution of distant historical frames to the current state is negligible compared to that of the immediately preceding frame [26]. The introduction of this hypothesis simplifies the analysis of the dynamic evolution process and allows it to be computed recursively and efficiently. These advantages have led to the widespread application of the first-order Markov assumption in RMM-based EOT methods.

However, the first-order Markov assumption can introduce significant errors because its core premise is often violated in practice. The assumption dictates that all historical information is discarded except for the immediately preceding state and extension. In reality, the motion of objects such as vehicles and civilian aircraft often exhibits high-order Markovian dynamics; their movement is typically smooth and structured, which creates temporal correlations that beyond a single time step. Ignoring these temporal dependencies leads to the loss of critical information, which can result in substantial biases in the estimation of both state and extension.

And according to the hypotheses outlined in [6], the target's state $\mathbf{x}_k$ exhibits an explicit dependence on the extension $\mathbf{X}_k$; that is, the evolution of the centroid state is influenced by the morphology of the extension as defined in Eq. (1), while the evolution of the extension is independent of the target's state. This assumption appears to be counterintuitive, as in practical EOT tasks, the state and extension often exhibit intricate interdependencies (see Fig. 1). The deviation causes the state to exert an influence on the evolution of the extension. Consequently, the previously assumed independence between state and extension is no longer applicable.

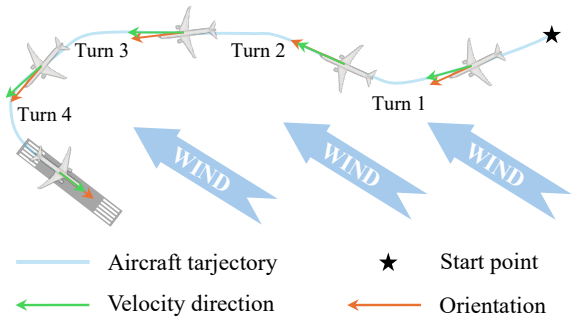


Fig. 1: Illustration of high-order Markovian characteristics and the coupling between kinematic state and extension in a real-world civilian aircraft. Temporal correlations often exist across multiple turning actions. For example, during the final turn, the aircraft's current state is influenced by earlier turns, since previous actions affect orientation and require the pilot to compensate accordingly to align with the runway. Additionally, to counteract crosswinds, the aircraft may perform a sideslip, where the velocity direction deviates from the orientation.

Besides, the measurement model for EOT, as defined in Eq. (3), is only an approximation to the measurement process. As such, it can present substantial discrepancies with the actual measurement processes in real-world tasks. For instance, improperly calibrated sensors can introduce unquantifiable biases in the measurements. In addition, in the measurement process of the real world, the geometric relationship between the target's position, orientation angle, and the sensor often affects the measurement equation. Consequently, the measurement matrix $h_k(\cdot)$ in Eq. (3), must be expressed as a function of $\mathbf{x}_k$ and $\mathbf{X}_k$. This factor is frequently neglected in conventional modeling approaches.

D. Proposed Model

Inspired by [27], we develop a respective state and extension evolution model with high-order Markovian dynamics to remedy the estimation errors introduced by the first-order Markov assumption. This model couples the state and extension to better capture the complexities of real-world evolution processes. The specific equations are formulated as

State evolution:

$$\mathbf{x}_k = f_k^{\text{general}}(\mathbf{x}_{k-1}, ..., \mathbf{x}_1, \mathbf{X}_{k-1}, ..., \mathbf{X}_1) + \mathbf{w}_k, \quad (6)$$

Extension evolution:

$$p(\mathbf{X}_k|\mathbf{x}_{k-1}, ..., \mathbf{x}_1, \mathbf{X}_{k-1}, ..., \mathbf{X}_1) =$$
$$\mathcal{W}(\mathbf{X}_k; \delta_k, \phi_k^{\text{general}}(\mathbf{x}_{k-1}, ..., \mathbf{x}_1, \mathbf{X}_{k-1}, ..., \mathbf{X}_1)), \quad (7)$$

where $f_k^{\text{general}}(\cdot)$ and $\phi_k^{\text{general}}(\cdot)$ denote the generalized state and extension evolution function, respectively. These functions

encapsulate the entirety of the historical state and extension data into their respective evolution frameworks, thereby providing a thorough adaptation to real-world evolution models. However, this comprehensive modeling results in increased dimensionality, which complicates the modelling of their probability density function (PDF). It is noteworthy that when Eqs. (6) and (7) are reduced to include only $\mathbf{x}_k$ and $\mathbf{X}_k$, respectively, they simplify to the single frame forms delineated in Eqs. (1) and (2).

Remark 1: Eqs (6) *and* (7) *represent the general form of high-order Markovian dynamics. Instead of explicitly defining a fixed order, these formulations encompass all historical information from* 1 *to* $k-1$*. This design enables the model to learn the necessary amount of historical dependencies directly from data when the exact order is unknown.*

Similarly, the general measurement model for extended targets can be defined as

$$\mathbf{z}_k^i = h_k^{\text{general}}(\mathbf{x}_k, \mathbf{X}_k) + \mathbf{v}_k^i, \tag{8}$$

where $h_k^{\text{general}}(\cdot)$ is the generalized measurement function.

While the model above describes EOT more accurately by overcoming the limitations of first-order Markov assumption, its high dimensionality makes real-time tracking difficult. Specifically, it is challenging to perform recursion through conventional methods within this high-order Markovian framework.

To address the aforementioned challenges, we propose a novel hybrid approach for the joint estimation of a target's kinematic state and extension in high-order Markovian scenarios. Inspired by [28], our method effectively combines physical model priors with a data-driven component trained on offline data, which achieves precise target estimation while maintaining the computational efficiency of a recursive framework.

III. Deep Memory Aided Bayesian Filter for EOT

This section elucidates the design process of the framework in three sequential steps. First, we approximating the high-order Markovian problem as a relaxed first-order Markovian model, which is achieved by introducing terms to compensate for evolution and measurement errors. Upon this relaxed model, we devise a BF framework that jointly integrates the target's state, extension, and memory. The final implementation of this framework is our proposed *RMMNet* model, which uses a deep memory DNN to learn the analytically intractable components of the filter, thereby enabling a tractable, iterative solution.

A. Relaxed Model

1) First-Order Approximated Model: The high-order Markovian evolution processes in Eqs. (6) and (7) are characterized by high dimensional complexity, thereby rendering them infeasible for recursive computation. By introducing evolution functions $f_k(\cdot)$ and $\phi_k(\cdot)$, which exhibit first-order Markovian properties into the aforementioned models, evolution equations described by mismatch terms are derived in Eqs. (9) and (10), where $\mathbf{\Delta}_k^f$ and $\mathbf{\Delta}_k^\phi$ stand for the model mismatch errors in the state and extension evolution processes, respectively. And the process noise $\mathbf{w}_k$ obey a zero-mean Gaussian distribution with covariance matrix of $\mathbf{Q}_k$.

Similarly, by introducing the nominal measurement function $h_k(\cdot)$, the measurement equation incorporating the measurement mismatch error can be defined as

$$\mathbf{z}_k^i = \underbrace{h_k^{\text{general}}(\mathbf{x}_k, \mathbf{X}_k) - h_k(\mathbf{x}_k)}_{\mathbf{\Delta}_k^h} + h_k(\mathbf{x}_k) + \mathbf{v}_k^i, \tag{11}$$

where $\mathbf{\Delta}_k^h$ stands for the model mismatch errors in the measurement.

In Eq. (11), the variability among scatterers within a frame depends solely on the measurement noise $\mathbf{v}_k^i$ and is independent of $\mathbf{\Delta}_k^h$. Therefore, the model mismatch error in the measurement remains constant over the n_k observations.

The transition functions in Eqs. (9) and (10) are based on a first-order Markov approximation. Consequently, the resulting model mismatch errors become a function of the entire history of states and extensions, not just the previous time step. This cumulative dependency makes it difficult to represent these errors with a simple analytical expression. To create a tractable model, we therefore express the evolution mismatch errors, $\mathbf{\Delta}_k^f$ and $\mathbf{\Delta}_k^\phi$, using a hierarchical nested function:

$$\mathbf{c}_k = g_k^\Delta(\mathbf{x}_{k-1}, ..., \mathbf{x}_1, \mathbf{X}_{k-1}, ..., \mathbf{X}_1), \tag{12}$$

$$\mathbf{\Delta}_k^f = f_k^\Delta(\mathbf{c}_k),\ \mathbf{\Delta}_k^\phi = \phi_k^\Delta(\mathbf{c}_k). \tag{13}$$

Note that $\mathbf{c}_k$ denotes a memory term, encapsulating the state and extension patterns from time steps 1 to $k-1$, thus exhibiting high-order Markovian properties; the nonlinear function $g_k^\Delta(\cdot)$ is utilized to map high-dimensional historical features into the latent space, whereas $f_k^\Delta(\cdot)$ and $\phi_k^\Delta(\cdot)$ signify the nonlinear state and extension output functions, respectively, which map the latent memory information to the expressible state and extension evolution model mismatch terms.

The feature embedding function $g_k^\Delta(\cdot)$ in Eq. (12) lacks a concrete physical interpretation. Following the approach in [28], we postulate that $g_k^\Delta(\cdot)$ adheres to a fixed pattern, signifying its temporal invariance, i.e., $g_k^\Delta(\cdot) \Leftrightarrow g^\Delta(\cdot)$. Therefore, the memory update process can be described in a heuristic manner as follows:

$$\begin{aligned}\mathbf{c}_k &= \underbrace{g^c(g^c(g^c}_{k \text{ times}}(\cdots), \mathbf{x}_{k-2}, \mathbf{X}_{k-2}), \mathbf{x}_{k-1}, \mathbf{X}_{k-1}) \\ &= g^c(\mathbf{c}_{k-1}, \mathbf{x}_{k-1}, \mathbf{X}_{k-1}).\end{aligned} \tag{14}$$

Remark 2: Actually, $\mathbf{c}_k$ *is a latent feature vector with a much higher dimensionality than the kinematic and extension states. It is updated at each time step via a complex and analytically intractable function* $g^c(\cdot)$*, which integrates the previous memory along with previous kinematic and extension states. The high-dimensional design of* $\mathbf{c}_k$ *enables it to compress historical information, enabling* $\mathbf{x}_k$ *and* $\mathbf{X}_k$ *to evolve with high-order Markovian dynamics while the update process remains Markovian in form.*

Similarly, the error term $\mathbf{\Delta}_k^h$ arising from measurement model mismatches can be mapped through the nonlinear measurement output function $h^\Delta(\cdot)$, namely,

$$\mathbf{\Delta}_k^h = h^\Delta(\mathbf{x}_k, \mathbf{X}_k). \tag{15}$$

By transforming the aforementioned problem, we have demonstrated that the joint state-extension evolution model,

characterized by high-order Markovian properties, can be approximated to an iterative first-order Markov model by incorporating a memory mechanism. Nevertheless, the complex coupling among memory, state, and extension in Eq. (14) complicates the memory update process, making it difficult to articulate through concise analytical expressions.

2) Model Identification: Inspired by data-driven approaches for system identification [29], [30], we propose approximating the memory term by learning from offline data. Given that target motion generally exhibits analogous state evolution, extension evolution, and measurement, deep learning methods suitable for time series prediction can be employed to effectively distill common patterns from offline data. This approach can supplant the estimation process of $\mathbf{c}_k$, thereby improving estimation accuracy during the evolution process.

Let $\mathcal{D}$ denote the offline historical data, encompassing time-varying target states, true extents, and corresponding measurement sequences, i.e.,

$$\mathcal{D} = \{\bar{\mathbf{x}}^j_{1:K}, \overline{\mathbf{X}}^j_{1:K}, \mathbf{Z}^j_{1:K}\}_{j=1}^{J}, \tag{16}$$

where $\bar{\mathbf{x}}_{1:K}$, $\overline{\mathbf{X}}_{1:K}$, and $\mathbf{Z}_{1:K}$ denote the ground truth sequence of target's states, extension, and corresponding measurements, respectively; K represents the length of each sequence, while J indicates the number of sequence in the offline dataset. Such an offline dataset can be generated through straightforward simulations, for instance, by collecting the motion trajectories of a vehicle and the temporal variations in its extension.

With the introduction of offline data, the problem of estimating model mismatch errors is transformed into the task of learning the mapping functions $f_k^{\Delta}(\cdot)$, $\phi_k^{\Delta}(\cdot)$, and $h_k^{\Delta}(\cdot)$ from the offline dataset $\mathcal{D}$. However, in contrast to traditional Bayesian methods that naively estimate only the state and extension, the incorporation of memory and model mismatch errors complicates the evolution process, thus necessitating recursive expressions with analytical forms.

B. Bayesian Filtering Framework Design

We first establish a general Bayesian framework to describe the propagation of the probability density through its standard prediction and update steps. In Theorem 1, we extend the conventional RMM by incorporating a memory term $\mathbf{c}_k$ and offline data $\mathcal{D}$ to compensate for state-extension-memory mismatch errors. While this theoretical framework is comprehensive, the resulting integrals are analytically intractable, making direct application difficult. To overcome this, Theorem 2 derives a practical, closed-form solution. By employing a Gaussian assumption and a moment matching approach that extends the work in [28], we obtain analytical expressions for the filter. These expressions are directly applicable to the system model defined by Eqs. (6) - (8).

Theorem 1. *(Joint state-extension-memory Bayesian filtering framework) For the state, extension and memory evolution in Eqs. (9) - (15), given the previous measurements $\mathbf{Z}_{1:k-1}$, the offline dataset $\mathcal{D}$, and the previous joint posterior density $p(\mathbf{x}_{k-1}, \mathbf{X}_{k-1}, \mathbf{c}_{k-1}|\mathbf{Z}_{1:k-1}, \mathcal{D})$, the joint state-extension-memory density prediction is given by*

$$p(\mathbf{x}_k, \mathbf{X}_k, \mathbf{c}_k|\mathbf{Z}_{1:k-1}, \mathcal{D}) = \iiint P_k^1 p(\mathbf{x}_{k-1}, \mathbf{X}_{k-1}, \mathbf{c}_{k-1}|\mathbf{Z}_{1:k-1}, \mathcal{D}) \mathrm{d}\mathbf{x}_{k-1}\mathrm{d}\mathbf{X}_{k-1}\mathrm{d}\mathbf{c}_{k-1}, \tag{17}$$

with

$$\begin{aligned} P_k^1 = \iint & p(\mathbf{x}_k|\boldsymbol{\Delta}_k^f, \mathbf{x}_{k-1}, \mathbf{X}_k) p(\boldsymbol{\Delta}_k^f|\mathbf{c}_k, \mathcal{D}) \\ & \times p(\mathbf{X}_k|\boldsymbol{\Delta}_k^\phi, \mathbf{X}_{k-1}) p(\boldsymbol{\Delta}_k^\phi|\mathbf{c}_k, \mathcal{D}) \\ & \times p(\mathbf{c}_k|\mathbf{x}_{k-1}, \mathbf{X}_{k-1}, \mathbf{c}_{k-1}, \mathcal{D}) \mathrm{d}\boldsymbol{\Delta}_k^f \mathrm{d}\boldsymbol{\Delta}_k^\phi. \end{aligned} \tag{18}$$

The joint state-extension-memory density update is

$$\begin{aligned} p(\mathbf{x}_k, \mathbf{X}_k, \mathbf{c}_k|\mathbf{Z}_{1:k}, \mathcal{D}) \propto & \, p(\mathbf{x}_k, \mathbf{X}_k, \mathbf{c}_k|\mathbf{Z}_{1:k-1}, \mathcal{D}) \\ & \times \int p(\mathbf{Z}_k|\boldsymbol{\Delta}_k^h, \mathbf{x}_k, \mathbf{X}_k) p(\boldsymbol{\Delta}_k^h|\mathbf{x}_k, \mathbf{X}_k, \mathcal{D}) \mathrm{d}\boldsymbol{\Delta}_k^h. \end{aligned} \tag{19}$$

While the aforementioned BF framework provides a complete iterative process, deriving a closed-form analytical solution is challenging. This difficulty arises because the key probability densities for prediction and update are difficult to model explicitly, including the state evolution mismatch density $p(\boldsymbol{\Delta}_k^f|\mathbf{c}_k, \mathcal{D})$, the extension evolution mismatch density $p(\boldsymbol{\Delta}_k^\phi|\mathbf{c}_k, \mathcal{D})$, and the observation mismatch density $p(\boldsymbol{\Delta}_k^h|\mathbf{x}_k, \mathbf{X}_k, \mathcal{D})$.

Fortunately, the information inherent in rich offline datasets encapsulates the characteristics of these PDFs. Consequently, designing appropriate DNN modules may allow for the learning of the mapping of these unknown distributions from offline data. Once the forms of these PDFs have been ascertained, the analytical implementation of the filtering can be derived independently.

Before designing the specific DNN modules, it is imperative to derive the closed-form solution of the Bayesian filtering framework as stated in Theorem 1. The filtering process of the proposed framework has various implementation methods, such as Monte Carlo sampling [31] and Gaussian approximation [32]. Since Monte Carlo sampling typically optimizes the estimated values of variables in an online manner, it is often challenging to meet the low-latency requirements of real-time systems. In contrast, expressions derived from Gaussian approximation generally exhibit analytical properties, making them computationally efficient and thus consuming fewer system resources.

For this reason, we employ Gaussian approximation with moment matching to derive the filtering expressions. The following outlines the assumptions necessary to apply the Gaussian approximation.

$$\mathbf{x}_k = \underbrace{f_k^{\text{general}}(\mathbf{x}_{k-1}, ..., \mathbf{x}_1, \mathbf{X}_{k-1}, ..., \mathbf{X}_1) - f_k(\mathbf{x}_{k-1})}_{\boldsymbol{\Delta}_k^f} + f_k(\mathbf{x}_{k-1}) + \mathbf{w}_k, \tag{9}$$

$$p(\mathbf{X}_k|\mathbf{x}_{k-1}, ..., \mathbf{x}_1, \mathbf{X}_{k-1}, ..., \mathbf{X}_1) = \mathcal{W}(\mathbf{X}_k; \delta_k, \underbrace{\phi_k^{\text{general}}(\mathbf{x}_{k-1}, ..., \mathbf{x}_1, \mathbf{X}_{k-1}, ..., \mathbf{X}_1) - \phi_k(\mathbf{X}_{k-1})}_{\boldsymbol{\Delta}_k^\phi} + \phi_k(\mathbf{X}_{k-1})). \tag{10}$$

Assumption 1. At time k, the posterior of the kinematic state $\mathbf{x}_k$ is assumed to follow a multivariate Gaussian distribution with mean $\mathbf{x}_{k|k}$ and covariance $\mathbf{P}_{k|k}$ according to [33]. While the posterior of the extension $\mathbf{X}_k$ is assumed to follow an inverse Wishart distribution according to [6] with scalar parameter $v_{k|k}$ and matrix parameter $\mathbf{X}_{k|k}$:

$$p(\mathbf{x}_k|\mathbf{Z}_{1:k},\mathcal{D}) = \mathcal{N}(\mathbf{x}_k;\mathbf{x}_{k|k},\mathbf{P}_{k|k}), \tag{20}$$

$$p(\mathbf{X}_k|\mathbf{Z}_{1:k},\mathcal{D}) = \mathcal{IW}(\mathbf{x}_k;v_{k|k},\mathbf{X}_{k|k}). \tag{21}$$

Assumption 2. The probability densities of the compensation terms $\mathbf{\Delta}_k^f$ and $\mathbf{\Delta}_k^h$, as estimated by the network, are assumed to follow multivariate Gaussian distributions:

$$p(\mathbf{\Delta}_k^f|\mathbf{c}_k,\mathcal{D}) = \mathcal{N}(\mathbf{\Delta}_k^f;\hat{\mathbf{\Delta}}_k^f,\mathbf{P}_k^f), \tag{22}$$

$$p(\mathbf{\Delta}_k^h|\mathbf{x}_k,\mathbf{X}_k,\mathcal{D}) = \mathcal{N}(\mathbf{\Delta}_k^h;\hat{\mathbf{\Delta}}_k^h,\mathbf{P}_k^h). \tag{23}$$

To implement Assumption 2, we use DNNs to learn the required mismatch statistics from data. Specifically, the first- and second-order moments of the model mismatch terms are regressed using linear layers. For the state evolution mismatch, the mean $\hat{\mathbf{\Delta}}_k^f$ and covariance $\mathbf{P}_k^f$ are obtained by mapping the memory vector $\mathbf{c}_k$ through two linear layers, $\Upsilon_{\boldsymbol{\psi}^f}(\cdot)$ and $\Upsilon_{\boldsymbol{\psi}^{\mathbf{P}^f}}(\cdot)$, respectively:

$$\Upsilon_{\boldsymbol{\psi}^f}:\mathbb{R}^{d(\mathbf{c}_k)}\mapsto\mathbb{R}^{d(\hat{\mathbf{\Delta}}_k^f)},\quad \Upsilon_{\boldsymbol{\psi}^{\mathbf{P}^f}}:\mathbb{R}^{d(\mathbf{c}_k)}\mapsto\mathbb{R}^{d(\mathbf{P}^f)}. \tag{24}$$

Similarly, the first moment of extension evolution mismatch $\mathbf{P}_k^\phi$ is regressed by linear layer parameterized by $\boldsymbol{\psi}^{\mathbf{P}_k^\phi}$, namely, $\Upsilon_{\boldsymbol{\psi}^f}:\mathbb{R}^{d(\mathbf{c}_k)}\mapsto\mathbb{R}^{d(\mathbf{P}_k^\phi)}$. For the measurement mismatch, the mean $\hat{\mathbf{\Delta}}_k^h$ and covariance $\mathbf{P}_k^h$ are derived from linear layers $\Upsilon_{\boldsymbol{\psi}^h}(\cdot)$ and $\Upsilon_{\boldsymbol{\psi}^{\mathbf{P}^h}}(\cdot)$ that take the concatenated state and extension vectors as input. All mapping parameters $\boldsymbol{\psi}^{(\cdot)}$ are learned from the offline dataset. During the online estimation phase, these parameters are fixed, and no offline data is required as input.

We have substituted all probability densities from the assumptions above into Theorem 1, utilizing moment matching to derive an analytical expression for the filtering, as presented in Theorem 2. Notably, we have circumvented the complex integration process in Theorem 1 to obtain these expressions, which possess favorable recursive properties, making them applicable to practical filtering processes.

Theorem 2. *(Implementation of the joint estimation for state and extension) Under assumptions 1 and 2, the expressions used for expectation and covariance prediction of kinematic state are*

$$\mathbf{x}_{k|k-1} = f_k(\mathbf{x}_{k-1|k-1}) + \underbrace{\hat{\mathbf{\Delta}}_k^f}_{\text{learned}}, \tag{25}$$

$$\mathbf{P}_{k|k-1} = \mathbf{F}_k\mathbf{P}_{k-1|k-1}\mathbf{F}_k^{\mathrm{T}} + \mathbf{Q}_k + \underbrace{\mathbf{P}_k^f}_{\text{learned}}, \tag{26}$$

and the expressions update are

$$\mathbf{x}_{k|k} = \mathbf{x}_{k|k-1} + \mathbf{P}_{k|k-1}^{xz}(\mathbf{P}_{k|k-1}^{zz})^{-1}(\widetilde{\mathbf{z}}_k - \mathbf{z}_{k|k-1}), \tag{27}$$

$$\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1} - \mathbf{P}_{k|k-1}^{xz}(\mathbf{P}_{k|k-1}^{zz})^{-1}(\mathbf{P}_{k|k-1}^{xz})^{\mathrm{T}}, \tag{28}$$

where

$$\mathbf{z}_{k|k-1} = h_k(\mathbf{x}_{k|k-1}) + \underbrace{\hat{\mathbf{\Delta}}_k^h}_{\text{learned}}, \tag{29}$$

$$\mathbf{P}_{k|k-1}^{xz} = \mathbf{P}_{k|k-1}\mathbf{H}_k^{\mathrm{T}}, \tag{30}$$

$$\mathbf{P}_{k|k-1}^{zz} = \mathbf{H}_k\mathbf{P}_{k|k-1}\mathbf{H}_k^{\mathrm{T}} + \frac{|\mathbf{B}_k|^{d/2}}{n_k} + \underbrace{\mathbf{P}_k^h}_{\text{learned}}. \tag{31}$$

The prediction process for the extension and its uncertainty are expressed as

$$v_{k|k-1} = \frac{2\delta_k(\lambda_{k-1}+1)(\lambda_{k-1}-1)(\lambda_{k-1}-2)}{\lambda_{k-1}^2(\lambda_{k-1}+\delta_k)} + 2d + 4, \tag{32}$$

$$\mathbf{X}_{k|k-1} = \frac{v_{k|k-1}-2d-2}{\lambda_{k-1}}(\delta_k\mathbf{A}_k\mathbf{X}_{k-1|k-1}\mathbf{A}_k^{\mathrm{T}} + \underbrace{\mathbf{P}_k^\phi}_{\text{learned}}), \tag{33}$$

and the update process for the update are expressed as

$$v_{k|k} = v_{k|k-1} + n_k, \tag{34}$$

$$\mathbf{X}_{k|k} = \mathbf{P}_{k|k-1}^{zz}(\widetilde{\mathbf{z}}_k - \mathbf{z}_{k|k-1})(\widetilde{\mathbf{z}}_k - \mathbf{z}_{k|k-1})^{\mathrm{T}} + \mathbf{X}_{k|k-1} + \mathbf{B}_k^{-1}\widetilde{\mathbf{Z}}_k\mathbf{B}_k^{-\mathrm{T}}, \tag{35}$$

where

$$\lambda_{k-1} = v_{k-1|k-1} - 2d - 2. \tag{36}$$

Proof. See Appendix.

Adopting the Gaussian approximation above, we derive a formally iterative expression for the filter. However, the compensation terms derived from memory in the expression (variables underlined and annotated with "learned") are challenging to construct. To overcome this limitation, we next design a DNN based on the proposed BF architecture to learn the mapping of errors induced by model mismatches from offline data, thereby endowing the analytical expression of the filter with practical iterability.

C. RMMNet Design

The joint state-extension-memory BF shares some similar iterative forms with RNNs, as both are implemented via a step-by-step recursive method over time. Drawing inspiration from the recurrent recursion structure of RNNs, RMMNet utilizes LSTM as its backbone to capture and preserve memory information within sequences in the memory update block. The overall structure of RMMNet, shown in Fig. 2, contains DNN modules for estimating the first- and second-order moments of model mismatch errors.

The architecture is composed of three core blocks. Specifically, the Memory Update Block (MUB) updates the memory vector $\mathbf{c}_k$ using the LSTM cell. It includes a cell state $\mathcal{C}_k$ for storing long-term contextual information and a hidden state $\mathcal{H}_k$. The hidden state first integrates posterior information from the previous time step $k-1$ and then interacts with the cell state to distill refined memory features. The Joint Prediction Block (JPB) uses this updated memory to generate compensations for the state and extension evolution according to Eqs. (25), (26) and (33). It employs three independent, two-layer fully connected networks to extract relevant information from the memory features to generate compensations. And the Joint Update Block (JUB) compensates for measurement model mismatch according to Eq. (31). It uses a linear layer to process the incoming scatterers, capturing the geometric relationship between the target and sensor to produce a correction for measurement.

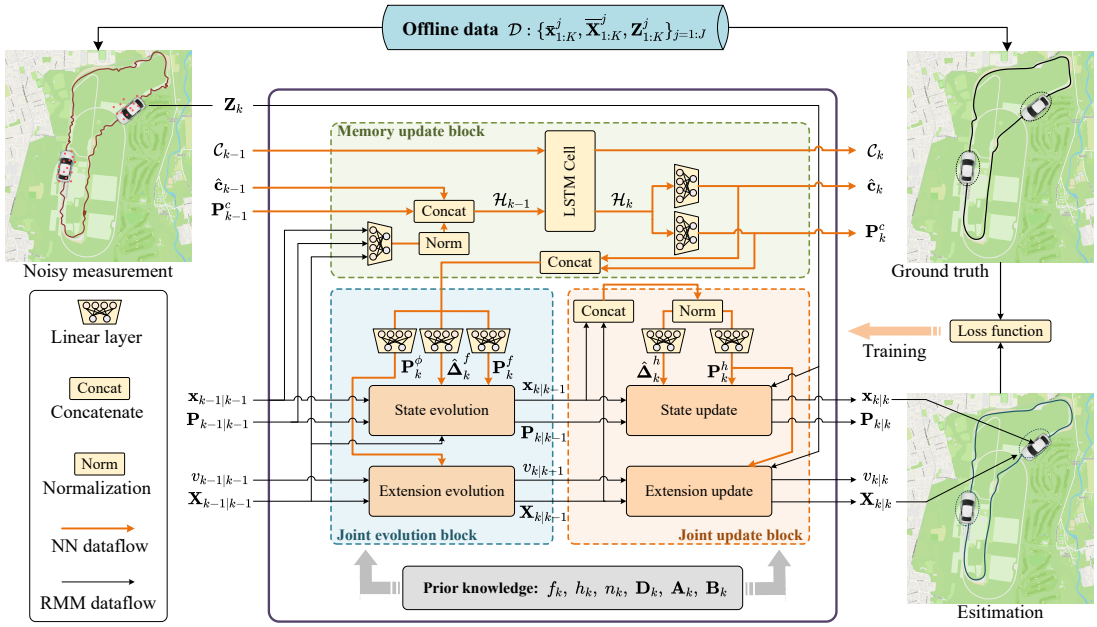


Fig. 2: The overall structure of RMMNet. "State evolution", "State update", "Extension evolution", and "Extension update" represent the formulas in Theorem 2 that exclude the error mismatch terms, derived from the basic RMM. The flow of variables is depicted by black lines. The network parameters along the orange lines (used for generate compensation terms in Theorem 2) are learned via training.

RMMNet integrates prior state and extension information through its filtering equations, while its neural network learns from offline data to compensate for errors induced by this prior information. This hybrid architecture is strictly grounded in BF theory, a foundation that allows the network modules within each block to remain simple and efficient, avoiding the need for overly complex designs.

Compared to the naive application of RNNs and LSTMs in target tracking, RMMNet is derived from BF theory, where DNNs are utilized to compensate for model mismatch errors within the filtering process. From the perspective of DNNs, this approach incorporates prior knowledge of the actual physical process, thereby reducing the network's reliance on large datasets. Additionally, the incorporation of second-order moments, representing uncertainty within the BF framework, endows the predictions of RMMNet with greater interpretability.

D. Algorithm Training

RMMNet adopts a self-supervised training approach, learning the high-order Markovian property inherent in offline dataset and the error information introduced due to the coupling between the state and extension in an end-to-end manner, as shown in Fig. 2. For the variables in the proposed BF framework that lack closed-form expression, obtaining their true labels is challenging, making it impossible to directly learn the corresponding mapping functions. Therefore, we compute the joint error between the model's estimated posteriors $\mathbf{x}_{k|k}$ and $\mathbf{X}_{k|k}$ with the ground truth in the offline dataset, and then use the backpropagation method to update the model's weight parameters. Notably, the iterative formulas with compensation for the error terms are implemented through a Gaussian approximation without random sampling, meaning that the closed-form of the proposed BF framework is differentiable. This enables the gradient of the error to propagate to the compensation terms we aim to learn, allowing the model to optimize the parameters. This learning method has been demonstrated as a viable paradigm in [28].

The stochastic gradient descent method is used for training RMMNet. We use the minimum mean square error (MMSE) combined with additional L2 regularization as the loss function, which is consistent with the strategy commonly employed in sequential regression methods. For the offline dataset $\mathcal{D}$ given in Eq. (16), during the t-th training iteration, we denote the parameters in RMMNet as $\boldsymbol{\Theta}_t$. The prediction error for the input of the j-th sequence can be calculated as follows:

$$\ell_j(\boldsymbol{\Theta}_t)=\frac{1}{2K}\sum_{k=1}^{K}(\parallel \mathbf{x}^j_{k|k}(\mathbf{Z}^j_k;\boldsymbol{\Theta}_t)-\bar{\mathbf{x}}^j_k\parallel^2 + \parallel \mathbf{X}^j_{k|k}(\mathbf{Z}^j_k;\boldsymbol{\Theta}_t)-\overline{\mathbf{X}}^j_k\parallel^2)+\gamma\parallel\boldsymbol{\Theta}_t\parallel^2 . \quad (37)$$

where, γ stands for a regularization coefficient applied to the L2 regularization term.

IV. Experimental Study

In this section, we evaluate RMMNet using both simulated and real-world data. The experimental results demonstrate RMMNet's strong performance for EOT in scenarios characterized by high-order Markovian dynamics. Comparative analysis with SOTA methods validates its superiority, and ablation studies confirm the efficacy of the memory update and compensation blocks[1]. The experiments are outlined as follows:

[1]The source code is available at: https://github.com/Austin2wwzx/RMMNet.

- **High-Order Markovian Vehicle Tracking Simulation.** A high-order autoregressive (AR) motion model according to [34] is used to generate the simlulated trajectories, which induces temporal correlations to approximate high-order Markovian and non-linear dynamics of real-world vehicles. Based on this simulated data, we compare RMMNet against five MB EOT methods: B-RMM [6], VB-RMM [35], IMM-RMM [23] and MEM-EKF [8], and the recent MEM-RBPF [9], as well as two purely DB time-series methods: LSTM [13] and Transformer [36].
- **High-Order Markovian Aircraft Tracking Experiment.** We use the Northern TRACON dataset [37], a publicly available real-world dataset of aircraft landing trajectories, to compare RMMNet against the same models used in the simulation. These trajectories exhibit high-order Markovian properties due to instructions from control towers and the aircraft's historical path. Moreover, the state and extension are coupled because crosswinds cause the aircraft's orientation to vary with its velocity.
- **Ablation Study.** An ablation study evaluates the effectiveness of each DNN block within RMMNet. By masking specific blocks, we validate the model's ability to compensate for mismatch errors arising from Markovian assumptions in joint state-extension estimation tasks.

Three metrics are used to quantify estimation errors: root mean square error (RMSE) for model accuracy, intersection over union (IoU) for overlap between estimated and ground truth ellipses, and Gaussian Wasserstein distance (GWD) for discrepancies between two ellipses. These metrics are detailed as follows:

RMSE: The position RMSE is calculated by comparing the state estimations with the ground truth on the test set as follows:

$$\mathrm{RMSE} = \sqrt{\frac{1}{JK}\sum_{j=1}^{J}\sum_{k=1}^{K} \parallel \mathbf{x}_{1,2:k|k}^{j} - \bar{\mathbf{x}}_{1,2:k}^{j} \parallel^2}, \quad (38)$$

where J is the total number of test cases, K is the sequence length of each cases, $\mathbf{x}_{1,2:k|k}^{j}$, and $\bar{\mathbf{x}}_{1,2:k}^{j}$ are the estimated position and the ground truth position, respectively. And the velocity RMSE is computed analogously, using the velocity components $\mathbf{x}_{3,4:k|k}^{j}$ and $\bar{\mathbf{x}}_{3,4:k}^{j}$ in the same formulation.

IoU: The elliptical equations for the estimated and ground truth extensions at time k are defined as:

$$\mathbf{ell}_{k|k} : \mathbf{x}_{1,2:k|k}^{\mathrm{T}} \mathbf{X}_{k|k} \mathbf{x}_{1,2:k|k} = 1, \quad (39)$$

$$\overline{\mathbf{ell}}_{k} : \bar{\mathbf{x}}_{1,2:k}^{\mathrm{T}} \overline{\mathbf{X}}_{k} \bar{\mathbf{x}}_{1,2:k} = 1, \quad (40)$$

where, $\mathbf{ell}_{k|k}$ and $\overline{\mathbf{ell}}_{k}$ are the estimated and ground truth extensions of the target, respectively. The IoU for two ellipse at time k is calculated as:

$$\mathrm{IoU}_k = \frac{\mathbf{ell}_{k|k} \cap \overline{\mathbf{ell}}_{k}}{\mathbf{ell}_{k|k} \cup \overline{\mathbf{ell}}_{k}}, \quad (41)$$

where $\mathbf{ell}_{k|k} \cap \overline{\mathbf{ell}}_{k}$ is the overlap area between estimated and ground truth ellipses, and $\mathbf{ell}_{k|k} \cup \overline{\mathbf{ell}}_{k}$ is their combined area.

GWD: The GWD quantifies the discrepancies between two ellipses at time k as:

$$\begin{aligned}\mathrm{GWD}_k =& \parallel \mathbf{x}_{1,2:k|k} - \bar{\mathbf{x}}_{1,2:k} \parallel^2 \\ &+ \mathrm{tr}[\mathbf{X}_{k|k} + \overline{\mathbf{X}}_{k} - 2(\mathbf{X}_{k|k}^{\frac{1}{2}} \overline{\mathbf{X}}_{k} \mathbf{X}_{k|k}^{\frac{1}{2}})^{\frac{1}{2}}]. \end{aligned} \quad (42)$$

A. High-Order Markovian Vehicle Tracking Simulation

In the simulation, we construct vehicle trajectories with extensions using a high-order AR motion model. Unlike first-order Markov model, the AR model predicts the current state based on multiple past states, approximating the high-order Markovian dynamics of real-world vehicle by summing weighted historical states and adding random noise.

Simulation setup. The kinematic state of the simulated vehicle on two-dimensional plane is modeled with a first-order evolution process for position, while velocity is described by the AR model. The evolution for state can be formulated as follows:

$$\begin{cases} \mathbf{m}_k = \mathbf{m}_{k-1} + \dot{\mathbf{m}}_{k-1}\Delta k, \\ \dot{\mathbf{m}}_k = \sum_{i=1}^{b} \mathbf{a}_i \dot{\mathbf{m}}_{k-i} + \boldsymbol{\varepsilon}_k, \end{cases} \quad (43)$$

where $\mathbf{m}_k$ is the position, $\dot{\mathbf{m}}_k$ is the velocity, Δk is the time interval, $\boldsymbol{\varepsilon}_k$ is the independent stochastic noise, $\mathbf{a}_i$ and b are the coefficients and order of the AR model, respectively.

The length of each trajectory is set to $K{=}1000s$, with a time interval of $\Delta k{=}1s$ between consecutive frames. The order b is set to 10 to approximate the high-order Markovian dynamics of real-world targets. The coefficients $\mathbf{a}_i$ are randomly selected from the range $(-\mathbf{1}, \mathbf{1})$, sorted by their absolute values in decreasing order to model temporal decay. The stochastic noise $\boldsymbol{\varepsilon}_k$ is sampled from a zero-mean Gaussian distribution with a unit covariance, i.e., $\boldsymbol{\varepsilon}_k{\sim}\mathcal{N}(\mathbf{0}, \mathbf{I}_2)$.

The target's extension is modeled as an ellipse with fixed major and minor axes of $12m$ and $2.5m$, respectively, and its orientation changes over time. The scenario assumes no false alarms or missed detections in sensor measurements. Scatterers are then generated as described in [8], distributed uniformly on the elliptical extension, with the number of scatterers per time step sampled from a Poisson distribution with a mean of 20. Each scatter is added by a zero-mean Gaussian noise with a variances of $1m^2$ and $0.64m^2$ in each dimension.

Following these setup, we generate 1,000 trajectories, divided into an offline training set of 800 and an online test set 200. The RMMNet, LSTM and Transformer are trained on the training set using 5-fold cross-validation. The test set is then used to benchmark all models including traditional MB EOT approaches.

Remark 3: It should be noted that the offline data is used exclusively for the training phase of all learning-based methods. During the subsequent online evaluation on the test set, this offline data is not provided to any method as auxiliary input.

Method configurations. We set the nominal models for the RMMNet as follows:

$$f_k(\mathbf{x}_{k-1}) = \begin{bmatrix} \mathbf{I}_2 & \Delta k \mathbf{I}_2 \\ \mathbf{0}_{2\times 2} & \mathbf{I}_2 \end{bmatrix} \mathbf{x}_{k-1}, \quad (44)$$

$$\phi_k(\mathbf{X}_{k-1}) = \frac{1}{\delta_k^2}\mathbf{X}_{k-1}, \quad (45)$$

$$h_k(\mathbf{x}_k) = \begin{bmatrix} \mathbf{I}_2 & \mathbf{0}_{2\times 2} \end{bmatrix} \mathbf{x}_k, \quad (46)$$

where $f_k(\cdot)$ is the constant velocity (CV) model with time interval $\Delta k{=}1s$, $\phi_k(\cdot)$ is a 2D extension evolution function ($d{=}2$) according to [23] with $\mathbf{A}_k{=}\frac{1}{\delta_k}\mathbf{I}_2$, and $h_k(\cdot)$ is a linear measurement function. The process noise covariance $\mathbf{Q}_k$ is set as

$$\mathbf{Q}_k = \sigma_w^2 \begin{bmatrix} \frac{\Delta_k^4}{4} & \frac{\Delta_k^3}{2} \\ \frac{\Delta_k^3}{2} & \Delta_k^2 \end{bmatrix} \otimes \mathbf{I}_2, \quad (47)$$

where σ_w^2 is the standard derivation of acceleration, set to $1m/s^2$. The observation distortion matrix $\mathbf{B}_k$ is set to $\mathbf{I}_2$, then the measurement noise covariance equivalents to $\mathbf{X}_k$.

To compensate for the mismatch between the nominal models and the approximately high-order Markovian ground-truth dynamics given in Eq. (43), we increase the capacity of RMMNet by setting the dimension of both its memory vector and its linear layers responsible for generating mismatch compensation to 128.

The traditional MB EOT methods used for comparison, which are based on the first-order Markov assumption, are configured as follows: 1) B-RMM: The temporal decay constant τ is set to 0.5. 2) VB-RMM: The measurement scaling factor s is set to 0.25, and the number of variational iterations is fixed at 5. 3) IMM-RMM: The interacting model set includes a CV model, a constant acceleration model, and two coordinated turn (CT) models with turn rates of $\pm 3^{\circ}/s$, and the extension evolution and the measurement distortion matrices are same as RMMNet. 4) MEM-EKF: the prior shape variables are initialized with a covariance matrix $\mathbf{C}_0^{\boldsymbol{p}}$=diag$[1, 10^2, 3^2]$, and the transition matrix is $\mathbf{A}_k^{\boldsymbol{p}}$=$\mathbf{I}_3$. The process and measurement noise covariance are $\mathbf{C}_{\boldsymbol{p}}^{\boldsymbol{w}}$=diag$[0.1, 1, 1]$ and $\mathbf{C}^{\boldsymbol{v}}$=$\mathbf{I}_2$. 5) MEM-RBPF: the num of the particles is set to 10, the orientation noise is set to 0.2, and the initial semi-axes are set to $10m$ and $3m$, respectively.

To maintain consistency across all RMM based approaches (i.e., B-RMM, VB-RMM, IMM-RMM, and our RMMNet), the initial inverse-Wishart parameters are uniformly set as $v_{0|0}$=7 and $\mathbf{X}_{0|0}$=diag$[10^2, 3^2]$, the value for the matrix $\mathbf{X}_{0|0}$ was chosen to approximate the actual size of the target.

The purely DB models, LSTM and Transformer, both operate autoregressively. The LSTM is implemented as a first-order Markov model, using only the previous estimation and the current measurement as input. In contrast, the Transformer is high-order Markovian; its decoder-only architecture with self-attention mechanism processes the entire history of estimation along with the current measurement. The specific architectures are: 1) LSTM: A single-layer LSTM with 128 hidden nodes for both hidden and cell states. 2) Transformer: A single-layer Transformer decoder with 8 self-attention heads and an embedding feature dimension of 128. The training loss is computed as the MMSE with an L2 regularization coefficient of γ=0.01 same as RMMNet.

Benchmark results and analysis. The comparison results on the simulated test set are summarized in Table I. The mean RMSE values for position (Pos.) and velocity (Velo.), as well as the IoU and GWD for extension estimation, are presented. The results indicate that RMMNet outperforms all other methods in terms of position and extension estimation accuracy.

By leveraging a deep memory mechanism, RMMNet extracts long-term temporal information from past estimations effectively, thereby compensating for model mismatch errors induced by the first-order Markov assumption. Compared to traditional MB EOT methods, the incorporation of this sequential information enables RMMNet to better adapt to targets with high-order Markovian dynamics, resulting in superior estimation accuracy for position, velocity, and extension. On the other hand, purely DB methods are prone to divergence in EOT tasks. This is because DNNs operating in an autoregressive manner generally suffer from error accumulation, where prediction errors from earlier time steps are propagated and amplified. In contrast, RMMNet integrates prior motion models to guide the prediction of both state and extension, thereby prevents error accumulation and ensures bounded, stable estimations.

TABLE I: Benchmark results on the simulated test set.

Methods	RMSE ↓		IoU ↑ ($\times 10^{-1}$)	GWD ↓
	Pos. (m)	Velo. (m/s)		
B-RMM	1.2891	1.2143	4.2173	5.9499
VB-RMM	0.6032	5.1542	6.0915	2.5317
IMM-RMM	0.6038	1.1618	6.2819	3.1245
MEM-EKF	1.5208	1.0163	3.0629	9.1525
MEM-RBPF	0.6013	0.6388	4.8191	8.3421
LSTM	32.2685	2.4821	0.0414	3862.7581
Transformer	47.4475	2.9012	0.0339	4521.3716
RMMNet	**0.5933**	**0.5910**	**7.0425**	**0.9182**

Case analysis. Fig. 3 visualizes the estimation results for a single case from the test set. Subfigure (a) displays the ground truth trajectory of the elliptical target, where the varying spacing between consecutive ellipses corresponds to the target's velocity, with larger intervals indicating higher speeds and vice versa. The remaining subfigures detail the estimation results from each method. The MB methods (i.e., B-RMM, VB-RMM, IMM-RMM, MEM-EKF, and MEM-RBPF), which are based on the first-order Markov assumption, all exhibit significant deviations from the ground truth. This deficiency is particularly evident in subfigure (c), where the target executes a continuous turn. Such actions introduce long-term correlations in the target's dynamics, creating a mismatch with the motion models employed by the MB methods. Although IMM-RMM is suit for maneuvering targets, its reliance on a pre-configured model set renders it ineffective when tracking targets with complex, high-order Markovian dynamics. In contrast, RMMNet delivers accurate estimations across all subfigures, owing to its ability to effectively leverage historical information for more precise state prediction.

The time-series error plots in Fig. 4 quantitatively confirm the superior performance of RMMNet on the sequence from Fig. 3, showing a consistent advantage with lower errors in both state estimation (Pos. RMSE) and extension estimation (IoU and GWD) for the majority of the sequence.

Data dependency analysis. We evaluate the data dependency of several learning-based methods: LSTM, Transformer, and our proposed RMMNet. For this analysis, we generate training sets of varying volumes (100 to 200,000 trajectories) alongside a fixed test set of 2,000 trajectories. Both the data generation process and the model configurations are kept identical to those in the main benchmark to ensure a fair comparison. Each method is subsequently trained on the different data volumes and evaluated on the test set. Fig. 5 presents the results, plotting the logarithm of the position estimation RMSE.

The results show that RMMNet is far less dependent on training data volumes than purely DB methods. Even with limited data, RMMNet maintains stable estimation accuracy. This robustness stems from the integration of prior physical knowledge into its learning framework, which effectively constrains the parameter search space. In contrast, purely DB methods struggle with scarce data, leading to underfitting and poor state estimation. Although the performance of these DB methods improves with

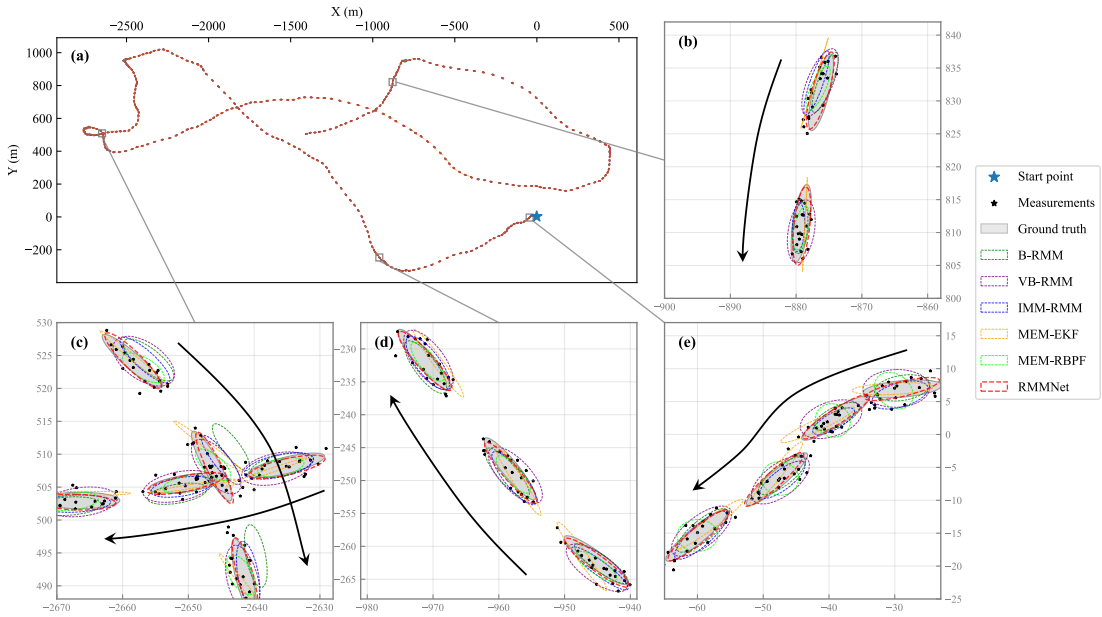


Fig. 3: Visual comparison of tracking performance in a single example of test set, showing the collected scatterers, the ground truth trajectory, and the estimates produced by all methods. The black arrowed lines denote the direction of the target's motion.

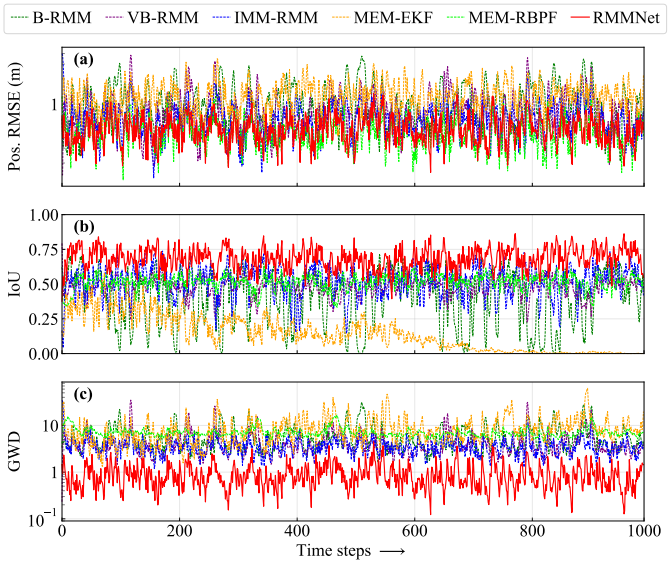


Fig. 4: Time-series of estimation errors for the single example. The plots show the (a) Position RMSE, (b) IoU, and (c) GWD.

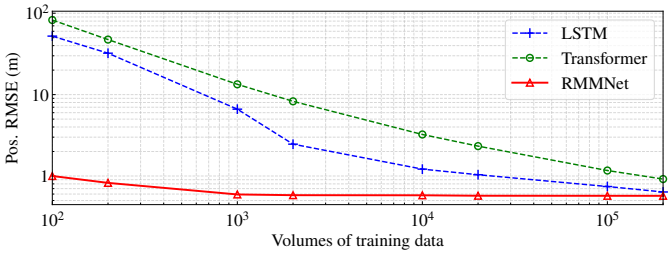


Fig. 5: Position estimation RMSE under different train data volumes.

more data, RMMNet consistently outperforms them, highlighting its distinct advantage in data-limited scenarios. As the training set grows, the performance of the purely DB methods begins to approach that of RMMNet, becoming comparable when the dataset reaches 200,000 training cases. However, acquiring such a large-scale dataset is often infeasible for real-world applications.

B. High-Order Markovian Aircraft Tracking Experiment

For a more challenging evaluation, we use the Northern California TRACON dataset, which contains realistic trajectories of landing aircraft. The trajectories in this benchmark are characterized by two complex properties: high-order Markovian dynamics, as aircraft adhere to guided flight paths, and a natural coupling between their state (e.g., velocity) and extension (e.g., orientation), which arises from pilots adjusting the heading to counteract crosswinds.

Experiment setup. From four months of aircraft landing trajectories for a single runway, we construct a dataset of 260 sequences. The raw data is preprocessed by first resampling all trajectories to a constant time interval of $\Delta k{=}4s$ and then segmenting each into a fixed length of $K{=}200$ time steps. This dataset is then partitioned into 208 sequences for training and 52 for testing.

To assess the RMMNet's generalization from simulation to reality, we use the AR model from Eq. (43) with varying orders ($b{=}1, 3, 5, 10$) to fit 208 real-world aircraft trajectories, then generates 208 simulated trajectories, which are then used to train variants of the RMMNet (denoted as RMMNet^b). These variants are identical in configuration to the original RMMNet, differing only in their training data. For these simulations, the random noise covariance was also increased to $10\mathbf{I}_2$, i.e. $\boldsymbol{\varepsilon}_k{\sim}\mathcal{N}(\mathbf{0}, 10\mathbf{I}_2)$. These variants, having never been trained on real data, are then evaluated on 52 real-world test trajectories.

For both real-world and synthetic data, we use an identical measurement generation process. First, the aircraft is modeled as

a ellipse with a major axis of $73.9m$ and a minor axis of $64.8m$. Consistent with the previous simulation, the scenario assumes no false alarms or missed detections. At each time step, the number of scatterers is drawn from a Poisson distribution with a mean of 10. These points are then uniformly distributed across the aircraft's surface, following the method in [38]. Finally, each scatterer's position is corrupted by zero-mean Gaussian noise with variances of $25m^2$ and $20m^2$ along the respective axes.

Method configurations. For B-RMM, VB-RMM, and RMMNet, the prior state evolution and measurement models are the same as those in Eqs. (44) - (46), using a time interval of $\Delta k{=}4s$. The process noise covariance, $\mathbf{Q}_k$, is also set according to Eq. (47) with $\sigma_w{=}5m/s^2$. For our RMMNet, the dimensions for both the memory vector and the compensation linear layers are set to 96. The purely DB methods are configured as follows: the LSTM uses 96 hidden nodes for both hidden and cell states, and the Transformer is set with 16 self-attention heads. All other configurations for these baselines remain identical to those in the previous simulation.

The remaining configurations for the traditional MB EOT methods are as follows: 1) B-RMM: the temporal decay constant τ is set to 2. 2) VB-RMM: the number of variational iterations is set to 6. 3) IMM-RMM: the model set includes a CV model, a CA model, and two CT models with turn rates of $\pm 1^{\circ}/s$. 4) MEM-EKF: the prior shape variables is set as $\mathbf{C}_0^{\boldsymbol{p}}{=}\text{diag}[1, 70^2, 60^2]$, and the transition matrix is $\mathbf{A}_k^{\boldsymbol{p}}{=}\mathbf{I}_3$. The process and measurement noise covariance are $\mathbf{C}_{\boldsymbol{p}}^{\boldsymbol{w}}{=}\text{diag}[1, 2, 2]$ and $\mathbf{C}^{\boldsymbol{v}}{=}20\mathbf{I}_2$. 5) MEM-RBPF: the num of the particles is set to 50, the orientation noise is set to 0.1, and the initial semi-axes are set to $70m$ and $60m$, respectively. The initial inverse-Wishart parameters of RMM-based methods are uniformly set as $v_{0|0}{=}10$ and $\mathbf{X}_{0|0}{=}\text{diag}[70^2, 60^2]$.

Experimental results and analysis. The position RMSE, IoU, and GWD for each method on the test set are shown in Table II. Compared to purely DB methods, RMMNet improves position estimation accuracy by integrating prior knowledge through the Bayesian filter, significantly reducing IoU and GWD. Compared to MB methods, RMMNet better extracts state and extension modes from offline data and compensates for model mismatch errors. While slightly trailing MEM-RBPF in Pos. RMSE, RMMNet achieves the lowest estimation errors across all other metrics.

Furthermore, the performance of RMMNet^b trained solely on high-order (i.e., $b{=}10$) simulated data demonstrates the model's excellent generalization. It outperforms several MB methods across multiple metrics, proving its ability to make effective estimations in complex real-world scenarios. This indicates that a high-order AR motion model aligns well with the general dynamics of real-world targets. Conversely, as the order decreases, the temporal correlation of the AR model weakens, leading to a significant mismatch with the real trajectories and a corresponding decline in performance.

Fig. 6 shows the estimation errors at different time steps. Despite some fluctuations, RMMNet outperforms other methods in prediction accuracy, demonstrating its strong adaptability to complex high-order Markovian characteristics and its ability to capture the intricate relationships between state and extension for more accurate estimations in real-world EOT tasks.

TABLE II: Experimental results on the real-world aircraft test set.

Methods	RMSE ↓		IoU ↑	GWD ↓
	Pos. (m)	Velo. (m/s)	($\times 10^{-2}$)	($\times 10^{3}$)
B-RMM	12.2317	8.8747	32.4572	0.8639
VB-RMM	7.2044	12.4817	53.8806	0.2743
IMM-RMM	9.4948	5.5473	49.1342	0.3612
MEM-EKF	20.6670	9.2847	25.6783	1.0554
MEM-RBPF	**6.3971**	21.0381	52.3381	0.2923
LSTM	481.3142	15.2947	0.3592	463.3919
Transformer	353.9528	10.4817	0.9948	250.5908
RMMNet^1	9.0285	4.7831	49.0691	0.3522
RMMNet^3	8.1534	4.3451	49.8518	0.3179
RMMNet^5	7.8182	4.2186	50.4216	0.2973
RMMNet^{10}	7.6631	3.9851	53.7617	0.2754
RMMNet	6.4819	**3.2714**	**55.7263**	**0.2639**

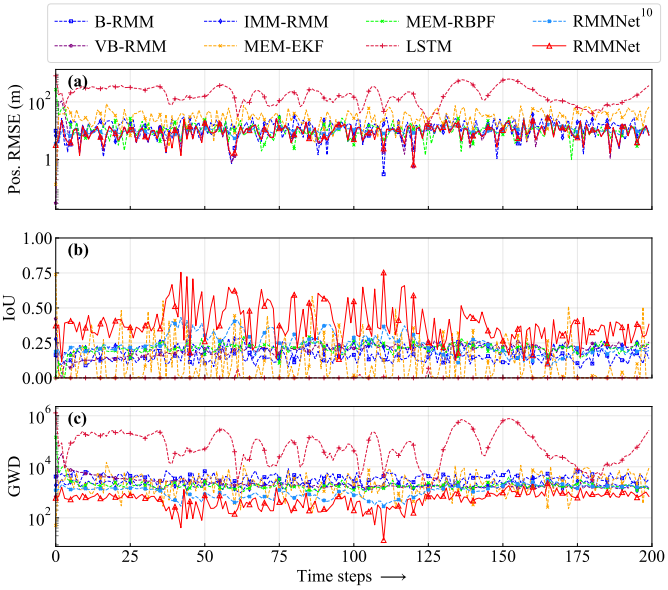


Fig. 6: Time-series of estimation errors for the single real-world aircraft trajectory. The plots show the (a) Position RMSE, (b) IoU, and (c) GWD.

C. Ablation Study

An ablation study is conducted to validate the contributions of the memory and mismatch compensation mechanisms in RMMNet for high-order Markovian scenarios. The study was performed on the simulated dataset from Section IV-A, where we evaluated the model's performance after selectively disabling its core deep learning components.

Experiment setup. For this ablation study, the offline training data, online testing data, and general configurations are identical to those used in the previous simulation. We compare the performance of the following model variants: 1) the full RMMNet as the control group; 2) (w/o) JEB, disabling JEB to assess compensation for evolution mismatch errors; 3) (w/o) JUB, disabling JUB to evaluate compensation for measurement mismatch errors; 4) (w/o) MUB, disabling MUB by removing the LSTM Cell and using only the previous state and extension as inputs for JEB and JUB; (w/o) MUB & JEB, disabling both

MUB and JEB, leaving only the JUB component active; and 5) (w/o) MUB & JUB, disabling both MUB and JUB.

Experimental results and analysis. Table III summarizes the optimal prediction performance of RMMNet and its five variants, with checkpoints selected based on the best joint estimation accuracy on the test set. The complete RMMNet achieves superior accuracy in both position and extension estimation, highlighting the effectiveness of its gating compensation units over traditional RMM-based EOT methods. The MUB captures the coupling effects and high-order Markovian dependencies, which are compensated by JEB during evolution. JUB further corrects model mismatch errors during updates, enhancing joint state-extension estimation accuracy.

TABLE III: Ablation study results with different variants on the test set.

Methods	RMSE ↓		IoU ↑	GWD ↓
	Pos. (m)	Velo. (m/s)	($\times 10^{-1}$)	
(w/o) JEB	0.7689	0.9232	5.7742	2.0453
(w/o) JUB	0.7548	0.8341	6.0361	1.6239
(w/o) MUB	1.0982	0.9543	5.2145	4.3981
(w/o) MUB & JEB	1.2145	1.1542	4.3541	5.0254
(w/o) MUB & JUB	1.1852	1.0253	4.5169	5.3157
RMMNet	**0.5933**	**0.5910**	**7.0425**	**0.9182**

V. Conclusion

We propose RMMNet, a novel framework that augments a BF with deep memory for joint state and extension estimation in high-order Markovian tasks. By integrating a closed-form BF with a neural network that learns to compensate for model mismatch from offline data, RMMNet achieves efficient recursive estimation while mitigating errors from state-extension coupling. Experiments on simulated and real-world datasets show that RMMNet outperforms both traditional MB and purely DB methods, showing significantly less reliance on large training datasets.

While RMMNet effectively captures long-term temporal dependencies in high-order Markovian scenarios, this study focuses on targets with smooth, structured motion; addressing maneuvering or abrupt model-switching remains a subject for future investigation.

Appendix

Implementation of Theorem 2

The mean of the state prediction is calculated as follows:

$$\begin{aligned}\mathbf{x}_{k|k-1} &= \mathbb{E}_{\mathbf{x}_k|\mathbf{z}_{1:k-1},\mathcal{D}}[\mathbf{x}_k] \\ &= \iiint \mathbf{x}_k p(\mathbf{x}_k, \mathbf{X}_k, \mathbf{c}_k|\mathbf{z}_{1:k-1}, \mathcal{D})\mathrm{d}\mathbf{x}_k\mathrm{d}\mathbf{X}_k\mathrm{d}\mathbf{c}_k. \quad (48)\end{aligned}$$

Plugging Eqs. (9), (20), and (22) into Eq. (48) we have

$$\mathbf{x}_{k|k-1} = \int f_k(\mathbf{x}_{k-1}) P^x_{k-1}\mathrm{d}\mathbf{x}_{k-1} + \hat{\mathbf{\Delta}}^f_k, \quad (49)$$

where $P^x_{k-1} = \mathcal{N}(\mathbf{x}_{k-1}; \mathbf{x}_{k-1|k-1}, \mathbf{P}_{k-1|k-1})$.

However, the $f_k(\cdot)$ in the above expression may represent a nonlinear state transition process, making this integral challenging to compute. We employed a first-order Taylor's formula for linearization, similar to that used in the extended Kalman filter (EKF) [26]. Employing it into Eq. (49), then we have Eq. (25).

The covariance of the state prediction is calculated as follows:

$$\begin{aligned}\mathbf{P}_{k|k-1} &= \mathbb{E}_{\mathbf{x}_k|\mathbf{z}_{1:k-1},\mathcal{D}}[(\mathbf{x}_k - \mathbf{x}_{k|k-1})(\mathbf{x}_k - \mathbf{x}_{k|k-1})^{\mathrm{T}}] \\ &= \iiiiiiii \mathbf{M}^f_k P^2_k \mathrm{d}\mathbf{x}_{k-1}\mathrm{d}\mathbf{X}_{k-1}\mathrm{d}\mathbf{c}_{k-1}\mathrm{d}\mathbf{X}_k\mathrm{d}\mathbf{c}_k\mathrm{d}\mathbf{\Delta}^f_k\mathrm{d}\mathbf{\Delta}^\phi_k, \quad (50)\end{aligned}$$

with $\mathbf{M}^f_k = (f_k(\mathbf{x}_{k-1}) + \mathbf{\Delta}^f_k + \mathbf{w}_k - \mathbf{x}_{k|k-1})(f_k(\mathbf{x}_{k-1}) + \mathbf{\Delta}^f_k + \mathbf{w}_k - \mathbf{x}_{k|k-1})^{\mathrm{T}}$.

Substistuting Eqs. (9), (20), and (22) into Eq. (50) can be calculated as

$$\begin{aligned}\mathbf{P}_{k|k-1} = &\int f_k(\mathbf{x}_{k-1})(f_k(\mathbf{x}_{k-1}))^{\mathrm{T}} P^x_{k-1}\mathrm{d}\mathbf{x}_{k-1} \\ &+ \mathbf{x}_{k|k-1}(\mathbf{x}_{k|k-1})^{\mathrm{T}} + \mathbf{Q}_k + \mathbf{P}^f_k. \quad (51)\end{aligned}$$

And the first-order Taylor's expansion of (51) yields Eq. (26).

The extension predicted probability density at time k is computed as

$$\begin{aligned}p(\mathbf{X}_k|\mathbf{z}_{1:k-1}, \mathcal{D}) &= \iint p(\mathbf{x}_k, \mathbf{X}_k, \mathbf{c}_k|\mathbf{z}_{1:k-1}, \mathcal{D})\mathrm{d}\mathbf{x}_k\mathrm{d}\mathbf{c}_k \\ &= \iiiiii P^1_k p(\mathbf{x}_{k-1}, \mathbf{X}_{k-1}, \mathbf{c}_{k-1}|\mathbf{Z}_{1:k-1}, \mathcal{D})\mathrm{d}\mathbf{x}_{k-1} \\ &\quad \mathrm{d}\mathbf{X}_{k-1}\mathrm{d}\mathbf{c}_{k-1}\mathrm{d}\mathbf{x}_k\mathrm{d}\mathbf{c}_k. \quad (52)\end{aligned}$$

Adopting the method used by [25] to approximate $\mathbf{X}_{k-1}$ with $\mathbf{X}_{k|k-1}$, and utilizing the Cholesky decomposition technique, the prediction of extension can be equivalently expressed as

$$\begin{aligned}\mathbf{A}_k\mathbf{X}_k\mathbf{A}^{\mathrm{T}}_k + \mathbf{\Delta}^\phi_k &\approx (\mathbf{A}_k\mathbf{X}_{k|k-1}\mathbf{A}^{\mathrm{T}}_k + \mathbf{\Delta}^\phi_k)^{\frac{1}{2}}\mathbf{X}^{-\frac{1}{2}}_{k|k-1}\mathbf{X}_{k-1} \\ &\quad \times \mathbf{X}^{-\frac{\mathrm{T}}{2}}_{k|k-1}(\mathbf{A}_k\mathbf{X}_{k|k-1}\mathbf{A}^{\mathrm{T}}_k + \mathbf{\Delta}^\phi_k)^{\frac{\mathrm{T}}{2}} \\ &= \mathbf{A}^*_k\mathbf{X}_k(\mathbf{A}^*_k)^{\mathrm{T}}. \quad (53)\end{aligned}$$

Substistuting Eqs. (10), (21) and (53) into Eq. (52) can be calculated as

$$\begin{aligned}p(\mathbf{X}_k|\mathbf{z}_{1:k-1}, \mathcal{D}) = &\iint \mathcal{GB}^{\mathrm{II}}_d(\mathbf{X}_k; \delta_k/2, \frac{v_{k-1|k-1} - d - 1}{2}; \\ &\mathbf{A}^*_k\mathbf{X}_k(\mathbf{A}^*_k)^{\mathrm{T}}, \mathbf{0})p(\mathbf{\Delta}^\phi_k|\mathbf{c}_k, \mathcal{D}) \\ &\times \mathcal{N}(\mathbf{c}_k; \hat{\mathbf{c}}_k, \mathbf{P}^c_k)\mathrm{d}\mathbf{\Delta}^\phi_k\mathrm{d}\mathbf{c}_k, \quad (54)\end{aligned}$$

where $\mathcal{GB}^{\mathrm{II}}_d(\cdot)$ is the Generalized Beta Type II distribution (GBII).

To obtain recursive estimation, we consider approximating this GBII distribution with an inverted Wishart distribution by moment matching similarly as in [6]:

$$p(\mathbf{X}_k|\mathbf{z}_{1:k-1}, \mathcal{D}) \approx \mathcal{IW}(\mathbf{X}_k; \mathbf{v}_{k|k-1}, \mathbf{X}_{k|k-1}), \quad (55)$$

with $\mathbf{v}_{k|k-1}$ and $\mathbf{X}_{k|k-1}$ are calculated as Eqs. (32) and (33), respectively.

The likelihood to get the measurement $\mathbf{Z}_k$ given both state and extension as well as the number of measurements in Eq. (4), can be factored as

$$\begin{aligned}p(\mathbf{Z}_k|n_k, \mathbf{\Delta}^h_k, \mathbf{x}_k, \mathbf{X}_k) &= \prod_{i=1}^{n_k} \mathcal{N}(\mathbf{z}^i_k; h_k(\mathbf{x}_k + \mathbf{\Delta}^h_k, \mathbf{B}_k\mathbf{X}_k\mathbf{B}^{\mathrm{T}}_k) \\ \propto \mathcal{N}(\widetilde{\mathbf{z}}_k; h_k(\mathbf{x}_k) + \mathbf{\Delta}^h_k, &\frac{\mathbf{B}_k\mathbf{X}_k\mathbf{B}^{\mathrm{T}}_k}{n_k})\mathcal{W}(\widetilde{\mathbf{Z}}_k; n_k - 1, \mathbf{B}_k\mathbf{X}_k\mathbf{B}^{\mathrm{T}}_k). \quad (56)\end{aligned}$$

For state prediction, the mean of its Gaussian measurement is computed as

$$\begin{aligned}\mathbf{z}_{k|k-1} &= \mathbb{E}_{\mathbf{x}_k|\widetilde{\mathbf{z}}_{1:k},\mathcal{D}}[h_k(\mathbf{x}_k)+\boldsymbol{\Delta}_k^h] \\ &= \iiiint (h_k(\mathbf{x}_k)+\boldsymbol{\Delta}_k^h)P_k^3 \mathrm{d}\boldsymbol{\Delta}_k^h \mathrm{d}\mathbf{c}_k \mathrm{d}\mathbf{X}_k, \end{aligned} \tag{57}$$

with

$$P_k^3 = p(\boldsymbol{\Delta}_k^h|\mathbf{x}_k,\mathbf{X}_k,\mathcal{D})p(\mathbf{x}_k,\mathbf{X}_k,\mathbf{c}_k|\mathbf{Z}_{1:k-1},\mathcal{D}). \tag{58}$$

Substistuting Eq. (8) into Eq. (57) then we have

$$\mathbf{z}_{k|k-1} = \int h_k(\mathbf{x}_{k|k-1})P_{k-1}^x \mathrm{d}\mathbf{x}_{k-1} + \hat{\boldsymbol{\Delta}}_k^h. \tag{59}$$

whose first-order Taylor's expansion leads to Eq. (29).

We denote the residual between the measurement and the predicted measurement as $\boldsymbol{\sigma}_k^z = \widetilde{\mathbf{z}}_k - \mathbf{z}_{k|k-1}$, and the corresponding covariance of measurement error is given by

$$\mathbf{P}_{k|k-1}^{zz} = \mathbb{E}_{\mathbf{x}_k|\widetilde{\mathbf{z}}_k,\mathcal{D}}[\boldsymbol{\sigma}_k^z(\boldsymbol{\sigma}_k^z)^{\mathrm{T}}]. \tag{60}$$

Substistuting the Gaussian term in Eq. (56) and Eq. (23) into Eq. (60) thus we have Eq. (31). The mutual covariance of the state prediction and the predicted measurement is given by

$$\mathbf{P}_{k|k-1}^{xz} = \mathbb{E}_{\mathbf{x}_k|\widetilde{\mathbf{z}}_k,\mathcal{D}}[(\mathbf{x}_k-\mathbf{x}_{k|k-1})(\boldsymbol{\sigma}_k^z)^\top] \tag{61}$$

Following the conclusions in [32], we have the posterior state and covariance as Eqs. (27) and (28), respectively. The update of the extension follows the same derivation method as [39], leading to the Eqs. (34) and (35).

REFERENCES

[1] K. Granstrom, M. Baum, and S. Reuter, "Extended object tracking: Introduction, overview and applications," *J. Adv. Inf. Fusion*, vol. 12, no. 2, p. 139–174, 2017.

[2] K. Granström and M. Baum, "A tutorial on multiple extended object tracking," *Authorea Preprints*, 2023.

[3] T. Hirscher, A. Scheel, S. Reuter, and K. Dietmayer, "Multiple extended object tracking using Gaussian processes," in *International Conference on Information Fusion (FUSION)*. IEEE, 2016, pp. 868–875.

[4] G. Vivone and P. Braca, "Joint probabilistic data association tracker for extended target tracking applied to X-band marine radar data," *IEEE J. Ocean. Eng.*, vol. 41, no. 4, pp. 1007–1019, 2016.

[5] W. Liu, S. Zhu, C. Wen, and Y. Yu, "Structure modeling and estimation of multiple resolvable group targets via graph theory and multi-Bernoulli filter," *Automatica*, vol. 89, pp. 274–289, 2018.

[6] J. W. Koch, "Bayesian approach to extended object and cluster tracking using random matrices," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 44, no. 3, pp. 1042–1059, 2008.

[7] M. Baum and U. D. Hanebeck, "Extended object tracking with random hypersurface models," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 50, no. 1, pp. 149–159, 2014.

[8] S. Yang and M. Baum, "Tracking the orientation and axes lengths of an elliptical extended object," *IEEE Trans. Signal Process.*, vol. 67, no. 18, pp. 4720–4729, 2019.

[9] S. Steuernagel and M. Baum, "Extended object tracking by rao-blackwellized particle filtering for orientation estimation," *IEEE Transactions on Signal Processing*, vol. 73, pp. 2590–2602, 2025.

[10] M. Khodarahmi and V. Maihami, "A review on Kalman filter models," *Archives of Computational Methods in Engineering*, vol. 30, no. 1, pp. 727–747, 2023.

[11] B. Lim and S. Zohren, "Time-series forecasting with deep learning: a survey," *Philosophical Transactions of the Royal Society A*, vol. 379, no. 2194, 2021.

[12] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

[13] J. Schmidhuber, S. Hochreiter *et al.*, "Long short-term memory," *Neural Comput*, vol. 9, no. 8, pp. 1735–1780, 1997.

[14] S. Capobianco, L. M. Millefiori, N. Forti, P. Braca, and P. Willett, "Deep learning methods for vessel trajectory prediction based on recurrent neural networks," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 57, no. 6, pp. 4329–4346, 2021.

[15] G. Revach, N. Shlezinger, X. Ni, A. L. Escoriza, R. J. Van Sloun, and Y. C. Eldar, "KalmanNet: Neural network aided Kalman filtering for partially known dynamics," *IEEE Trans. Signal Process.*, vol. 70, pp. 1532–1547, 2022.

[16] G. Revach, X. Ni, N. Shlezinger, R. J. van Sloun, and Y. C. Eldar, "RTSNet: Learning to smooth in partially known state-space models," *IEEE Trans. Signal Process.*, vol. 71, pp. 4441–4456, 2023.

[17] T. Imbiriba, O. Straka, J. Duník, and P. Closas, "Augmented physics-based machine learning for navigation and tracking," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 60, no. 3, pp. 2692–2704, 2024.

[18] I. Buchnik, G. Revach, D. Steger, R. J. Van Sloun, T. Routtenberg, and N. Shlezinger, "Latent-KalmanNet: Learned Kalman filtering for tracking from high-dimensional signals," *IEEE Trans. Signal Process.*, vol. 72, pp. 352–367, 2023.

[19] F. Gama, N. Zilberstein, M. Sevilla, R. G. Baraniuk, and S. Segarra, "Unsupervised learning of sampling distributions for particle filters," *IEEE Trans. Signal Process.*, vol. 71, pp. 3852–3866, 2023.

[20] I. Nuri and N. Shlezinger, "Learning flock: Enhancing sets of particles for multi sub-state particle filtering with neural augmentation," *IEEE Trans. Signal Process.*, 2024.

[21] N. Shlezinger, G. Revach, A. Ghosh, S. Chatterjee, S. Tang, T. Imbiriba, J. Dunik, O. Straka, P. Closas, and Y. C. Eldar, "AI-aided Kalman filters," *arXiv preprint arXiv:2410.12289*, 2024.

[22] N. Shlezinger and Y. C. Eldar, "Model-based deep learning," *Foundations and Trends® in Signal Processing*, vol. 17, no. 4, pp. 291–416, 2023.

[23] J. Lan and X. R. Li, "Tracking of maneuvering non-ellipsoidal extended object or target group using random matrix," *IEEE Trans. Signal Process.*, vol. 62, no. 9, pp. 2450–2463, 2014.

[24] R. E. Kalman and R. S. Bucy, "New results in linear filtering and prediction theory," 1961.

[25] M. Feldmann and D. Franken, "Tracking of extended objects and group targets using random matrices—a new approach," in *International Conference on Information Fusion*. IEEE, 2008.

[26] J. Durbin and S. J. Koopman, *Time series analysis by state space methods*. OUP Oxford, 2012, vol. 38.

[27] D. González, F. Chinesta, and E. Cueto, "Learning non-Markovian physics from data," *Journal of Computational Physics*, vol. 428, p. 109982, 2021.

[28] S. Yan, Y. Liang, L. Zheng, M. Fan, X. Wang, and B. Wang, "Explainable gated Bayesian recurrent neural network for non-Markov state estimation," *IEEE Trans. Signal Process.*, vol. 72, pp. 4302–4317, 2024.

[29] A. Chiuso and G. Pillonetto, "System identification: A machine learning perspective," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 2, no. 1, pp. 281–304, 2019.

[30] L. Ljung, C. Andersson, K. Tiels, and T. B. Schön, "Deep learning and system identification," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 1175–1181, 2020.

[31] Y. El-Laham and M. F. Bugallo, "Policy gradient importance sampling forBayesian inference," *IEEE Trans. Signal Process.*, vol. 69, pp. 4245–4256, 2021.

[32] X. Wang, Y. Liang, Q. Pan, and F. Yang, "A Gaussian approximation recursive filter for nonlinear systems with correlated noises," *Automatica*, vol. 48, no. 9, pp. 2290–2297, 2012.

[33] K. Ito and K. Xiong, "Gaussian filters for nonlinear filtering problems," *IEEE Transactions on Automatic Control*, vol. 45, no. 5, pp. 910–927, 2000.

[34] Z. R. Zaidi and B. L. Mark, "Mobility tracking based on autoregressive models," *IEEE Transactions on Mobile Computing*, vol. 10, no. 1, pp. 32–43, 2011.

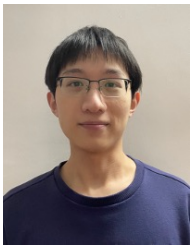
[35] U. Orguner, "A variational measurement update for extended target tracking with random matrices," *IEEE Trans. Signal Process.*, vol. 60, no. 7, pp. 3827–3834, 2012.

[36] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[37] M. Gariel, A. N. Srivastava, and E. Feron, "Trajectory clustering and an application to airspace monitoring," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 4, pp. 1511–1524, 2011.

[38] B. Tuncer and E. Özkan, "Random matrix based extended target tracking with orientation: A new model and inference," *IEEE Trans. Signal Process.*, vol. 69, pp. 1910–1923, 2021.

[39] J. Lan and X. R. Li, "Tracking of extended object or target group using random matrix—part I: New model and approach," in *International Conference on Information Fusion*. IEEE, 2012, pp. 2177–2184.

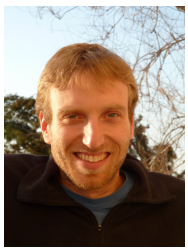


Zhixing Wang is currently working toward the Ph.D. degree with the Radar Research Laboratory, School of Information and Electronics, Beijing Institute of Technology, Beijing, China. His research interests include deep learning, target tracking, and signal processing.

Le Zheng (Senior Member, IEEE) received the B.Eng. degree from Northwestern Polytechnical University, Xi'an, China, in 2009, and the Ph.D. degree from Beijing Institute of Technology (BIT), Beijing, China, in 2015. From 2013 to 2014, he was a Visiting Researcher with the Department of Electrical Engineering, Columbia University, New York, NY, USA, where he later served as a Postdoctoral Research Fellow from 2015 to 2017. From 2018 to 2022, he was a Principal Radar Systems Engineer at Aptiv (formerly Delphi), Los Angeles, CA, USA, where he led projects on next-generation radar systems for autonomous driving. Since 2022, he has been a Professor with the School of Information and Electronics, Beijing Institute of Technology. His research interests include physical AI, signal processing, radar systems, and wireless communications.

Shi Yan received the Ph.D. degree in control science and engineering from the School of Automation, Northwestern Polytechnic University, Xi'an, China, in 2025. He is currently a Post-Doctoral Researcher with the Automation school, Northwestern Polytechnic University, Xi'an, China. His research interests include estimation theory, deep learning, and target tracking.

Ruud J. G. van Sloun (Senior Member, IEEE) received the M.Sc. and Ph.D. degrees (cum laude) in electrical engineering from Eindhoven University of Technology, Eindhoven, The Netherlands, in 2014 and 2018, respectively. He is currently an Associate Professor with the Department of Electrical Engineering, Eindhoven University of Technology. From 2019 to 2020, he was a Visiting Professor with the Department of Mathematics and Computer Science, Weizmann Institute of Science, Rehovot, Israel. From 2020 to 2023, he was a Kickstart AI Fellow with Philips Research, Eindhoven. His current research interests include closed-loop image formation, deep learning for signal processing and imaging, active signal acquisition, model-based deep learning, compressed sensing, ultrasound imaging, and probabilistic signal and image reconstruction. Dr. Sloun has been honored with an ERC Starting Grant, an NWO VIDI Grant, an NWO Rubicon Grant, and a Google Faculty Research Award.



Nir Shlezinger (M'17-SM'23) is an associate professor in the School of Electrical and Computer Engineering at Ben-Gurion University, Israel. He received his B.Sc., M.Sc., and Ph.D. degrees in 2011, 2013, and 2017, respectively, from Ben-Gurion University, Israel, all in electrical and computer engineering. From 2017 to 2019, he was a postdoctoral researcher at the Technion, and from 2019 to 2020, he was a postdoctoral researcher at the Weizmann Institute of Science, where he was awarded the FGS Prize for his research achievements. He is the recipient of the 2024 IEEE ComSoc Fred W. Ellersick Award, the 2025 IEEE ComSoc Marconi Prize, and the 2024 Krill Prize for outstanding young researchers. His research interests include communications, information theory, signal processing, and machine learning.

Yonina C. Eldar (Fellow, IEEE) received the B.Sc. degree in physics and the B.Sc. degree in electrical engineering from Tel-Aviv University, and the Ph.D. degree in electrical engineering and computer science from MIT. She is the Aoun Chair Professor of Electrical and Computer Engineering at Northeastern University and the Dorothy and Patrick Gorman Professorial Chair of Mathematics and Computer Science at the Weizmann Institute where she founded and heads the Signal Acquisition Modeling Processing and Learning Lab (SAMPL) and the Center for Biomedical Engineering. She is also a Visiting Professor at MIT and Princeton, a Visiting Scientist at the Broad Institute, and an Adjunct Professor at Duke University and was a Visiting Professor at Stanford. She is a member of the Israel Academy of Sciences and Humanities and of the Academia Europaea, a EURASIP Fellow. She has received many awards for excellence in research and teaching, including the Israel Prize (2025), Landau Prize (2024), IEEE Signal Processing Society Technical Achievement Award (2013), the IEEE/AESS Fred Nathanson Memorial Radar Award (2014) and the IEEE Kiyo Tomiyasu Award (2016). She received the Michael Bruno Memorial Award from the Rothschild Foundation, the Weizmann Prize for Exact Sciences, the Wolf Foundation Krill Prize for Excellence in Scientific Research, the Henry Taub Prize for Excellence in Research (twice), the Hershel Rich Innovation Award (three times), and the Award for Women with Distinguished Contributions. She was selected as one of the 50 most influential women in Israel, and was a member of the Israel Committee for Higher Education. She is the Editor in Chief of Foundations and Trends in Signal Processing, a member of several IEEE Technical Committees and Award Committees, and heads the Committee for Promoting Gender Fairness in Higher Education Institutions in Israel.