Information Fusion 127 (2026) 103777

Contents lists available at ScienceDirect

Information Fusion

journal homepage: www.elsevier.com/locate/inffus

Ensemble of KalmanNets with innovation-based attention for robust target tracking

Marco Mari *, Lauro Snidaro

Department of Mathematics, Computer Science and Physics, University of Udine, Udine, Italy

ARTICLE INFO

Keywords:
Target tracking
Recurrent neural networks
Kalman filter
Maneuvering targets

ABSTRACT

Model-based tracking algorithms often suffer from significant performance degradation when tracking maneuvering targets, primarily due to inherent uncertainties in target dynamics. To address this limitation, we propose a novel ensemble-based approach that integrates multiple neural-aided Kalman filters, referred to as KalmanNet, within a multiple-model framework, inspired by traditional interacting multiple-model (IMM) filtering techniques. Each KalmanNet instance is specialized in tracking targets governed by a distinct motion model. The ensemble fuses their state estimates using a Recurrent Neural Network (RNN), which learns to adaptively weigh and combine the predictions based on the underlying target dynamics. This fusion mechanism enables the system to model complex motion patterns more effectively and achieves lower estimation bias and variance compared to relying on a single KalmanNet when tracking maneuvering targets, as demonstrated through extensive simulation experiments. Furthermore, we introduce an explainable, innovation-based attention mechanism to enhance the interpretability of our results, inspired by traditional model-based tracking algorithms, that aids the identification of target motion dynamics. Our findings indicate that this attention mechanism improves robustness to sensor noise, out-of-distribution data, and missing measurements. Overall, this innovative approach has the potential to advance state-of-the-art target tracking applications.

1. Introduction

Target state estimation is a critical task in target tracking. Accurate target state estimation can be achieved by the implementation of a reliable and consistent tracking algorithm. In the framework of Bayesian recursive state estimation, this task is addressed by applying mathematical models that accurately represent the dynamic state of the target over time.

In 1960, Kalman introduced the Kalman Filter (KF), a pioneering optimal estimator for linear systems with Gaussian noise. The KF's minimal complexity and solid theoretical foundation made it an ideal solution for state estimation in discrete-time systems adequately described by SS models.

In systems with strongly nonlinear dynamics, relying on linearization can introduce substantial errors, sometimes resulting in filter divergence or unreliable state estimates. The Unscented Kalman Filter (UKF) [1] addresses these issues by using the unscented transform, which propagates a set of carefully chosen sigma points through the nonlinear system. This method sidesteps the need for explicit linearization and avoids the approximation errors associated with Jacobian computations, while still achieving second-order accuracy for the mean and covariance of the state estimate.

When the distribution of target states is non-Gaussian, as in cases where external constraints restrict movement, Particle Filter (PF) [2] often outperforms Gaussian-based filters. By representing the probability distribution with weighted samples, named particles, the PF can more accurately capture complex distributions, and it can incorporate nonlinear constraints directly through importance sampling. It should be highlighted, nonetheless, that as the PF accuracy rises with the number of particles collected to approximate the posterior distribution, the computational complexity is significantly impacted.

However, tracking targets that exhibit erratic maneuvers remains a significant challenge for the sensor fusion community. As Li and Jilkov proposed [3], accurate modeling of the target's motion is crucial for optimal tracking. Representing target dynamics as a jump Markov process is a useful approach, especially when these dynamics are subject to change over time. This approach involves modeling the target's dynamics using a set of state-space models, where each model represents a specific dynamic behavior. At each time step, the target's motion can be described by one of these models. Among the algorithms that utilize this

* Corresponding authors.
E-mail addresses: mari.marco@spes.uniud.it (M. Mari), lauro.snidaro@uniud.it (L. Snidaro).

https://doi.org/10.1016/j.inffus.2025.103777
Received 6 May 2025; Received in revised form 22 August 2025; Accepted 23 September 2025
Available online 27 September 2025

approach, the Interacting Multiple Model (IMM) algorithm has proven to be highly effective for tracking maneuvering targets [4].

The unpredictable nature of target movements, particularly in complex and non-linear environments, demands the creation of stable, robust algorithms. Tracker performance may degrade without effective maneuver detection and compensation, potentially causing filter divergence [5,6]. Thus, ongoing advancements in algorithms capable of managing these challenges are crucial for maintaining the safety and effectiveness of tracking systems.

At the same time, recent Deep Learning (DL) advancements have shown significant promise in real-world applications. Recurrent Neural Networks (RNNs), in particular, Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU) [7], and, more recently, Transformer-based architectures, have been proven to be effective in analyzing sequential data and have been efficiently applied to state estimation [8,9].

Neural networks (NN), capable of learning complex patterns from data, offer a promising solution for Maneuvering Target Tracking (MTT). By learning directly from data, NNs can adapt to dynamic scenarios, reducing reliance on predefined analytical models.

In [10], Liu et al. introduced DeepMTT, a stacked Bidirectional LSTM (BiLSTM)-based neural architecture, which captures temporal dependencies from radar observations in a sliding window, producing correction terms for state estimates generated by a UKF. Due to the promising tracking performance, it established a new state-of-the-art baseline for MTT solutions, particularly in the Air Traffic Control (ATC) domain.

In [11], Liu et al. develop an NN-based tracking system that leverages angular velocity estimation to predict state transitions during target maneuvers. The system effectively detected maneuvers, offering an efficient alternative to traditional IMM frameworks, overperforming against [10].

In [12], Zhao et al. exploited a Transformer-based encoder processing radar measurements to provide improved temporal insights for a Temporal Convolutional Network (TCN) to provide accurate target state estimates, decoding the extracted features. Subsequently, in [13], Zhang et al. further refined Zhao's approach by proposing a Transformer for Maneuvering Target Tracking (TrMTT), an encoder-decoder Transformer-based architecture reaching better tracking performance.

While the aforementioned approaches demonstrate strong tracking performance, their practical utility is limited by a reliance on sliding window buffering and moving average smoothing. The majority of these methodologies process radar measurements in an end-to-end manner, a factor that restricts the integration of a priori knowledge via analytical models. In contrast, DeepMTT operates in a manner that deviates from the conventional end-to-end fashion. Instead, it employs the utilization of NNs to rectify the posterior state estimates generated by a UKF, thereby facilitating the integration of prior knowledge. However, as with the other approaches, its real-time applicability is constrained by the sliding window and averaging mechanisms.

From an industrial point of view, preserving some degree of interpretability in the tracking system is crucial, as it enables proper understanding of errors and the design of countermeasures to improve robustness and adaptability to different scenarios. In this context, limiting the integration of NNs within the Bayesian framework to specific tasks appears to be a key approach for achieving both an explainable NN-based tracking system and improved performance.

In [14], Deng et al. integrated an LSTM-based NN with the traditional IMM filter to predict motion model probabilities, to optimize model interaction and filtering, reducing recognition delays and estimation errors.

In [15–17], an NN-integration into the KF framework is proposed, named Mnemonic KF. It uses an LSTM-based NN to directly learn target dynamics from data rather than defining the motion model analytically. That allowed for increased tracking performance in their simulations when model mismatch occurs in traditional model-based (MB) filtering. Similarly, in [18], segments of radar measurements are processed to derive a linear form of the motion model for a KF. In [19], an effective fusion of data-driven and MB target dynamics is proposed to enhance tracking performance by learning a balancing coefficient that replaces the traditional Kalman gain within the KF framework. In [20,21], Yan et al. integrate memory-augmented GRUs into a Bayesian framework to address analytical model mismatches, capture complex recurrent maneuvering patterns, and enable data-driven fusion of noisy sensor measurements. This approach improves the modeling of target dynamics and enhances tracking robustness.

Very recent efforts have focused on integrating NNs within the Bayesian recursive estimation framework to improve target tracking performance. For example, Shen et al. [22] proposed the Maneuver Compensation Strong Tracker (MCST), which incorporates a Bi-LSTM architecture with maneuver compensation and attention modules, while maintaining the recursive prediction-update structure of Bayesian filters. Similarly, Lin et al. [23] introduced an Attentional Filtering (AF) framework that replaces the Kalman gain with a learned attention mechanism to balance MB priors and sensor-derived evidence. While both approaches produce competitive results, they also highlight the ongoing challenge of combining data-driven components with MB filtering, particularly with regard to addressing uncertainty and ensuring generalisability.

In our previous work [24], we experimented with an ensemble architecture resembling the IMM framework fusing state estimates from data-driven KFs, known as KalmanNets from [25]. Each KalmanNet uses a different motion model, while the ensemble architecture learns to fuse them by dynamically computing target mode likelihood and predicting a correction term for each of them, compensating for model mismatch.

In this work, we revisit the architecture in [24], improving its ability to adapt to various scenarios, and we also propose some modifications to its computational framework that improve its performance. In summary, the main contributions of the work are:

- Designing an ensemble-based tracking framework that fuses multiple specialized KalmanNet models through an RNN, enabling adaptive state estimation across diverse motion patterns and significantly improving performance on maneuvering targets.
- Addressing the problem of Multi-Task Learning (MTL) by formulating target motion mode estimation as an auxiliary task, leading to effective overfitting reduction.
- Introducing an interpretable and explainable innovation-based attention mechanism, conceptually inspired by Bayesian recursive filtering, in particular, the IMM filter, that guides both motion mode estimation and ensemble prediction by emphasizing the most probable motion model.
- Proposing a comprehensive experimental evaluation capable of distinguishing performance across a wide range of parameters that characterize real-world scenarios. This includes analysis of the proposed approach's sensitivity to out-of-distribution trajectories, varying measurement noise levels, and missing measurements. To the best of our knowledge, this constitutes a novel evaluation framework for data-driven tracking systems.

The remainder of the paper is divided as follows: in Section 2 we introduce the problem of MTT and recall how MB frameworks and the KalmanNet [25] tackle it; in Section 3 we describe the ensemble architecture, introducing the innovation-based attention mechanism; in Section 4 the experimental setting is presented and the tracking results are analyzed; finally, in Section 5, we draw some conclusions by commenting on the merits and demerits of the proposed MTT methodology, listing possible future developments and the most promising research directions.

2. Problem formulation

Let $t \in \mathbb{N}$ be a discrete-time instant at which we aim to estimate the target state x_t, comprising both cartesian position and velocity

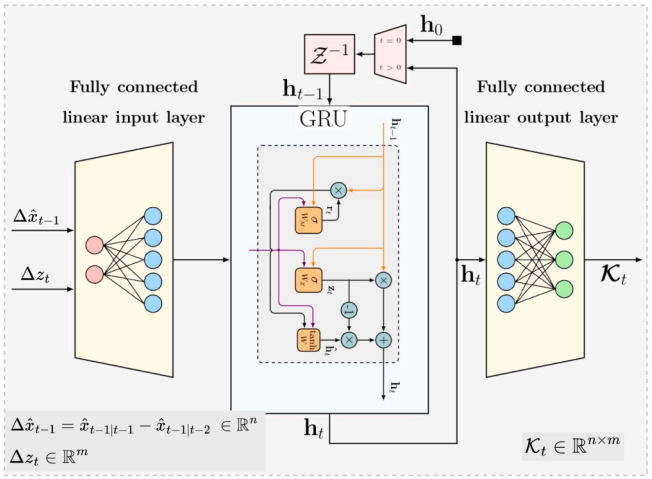

Fig. 1. KalmanNet architecture configuration #1. Image adapted from [25].

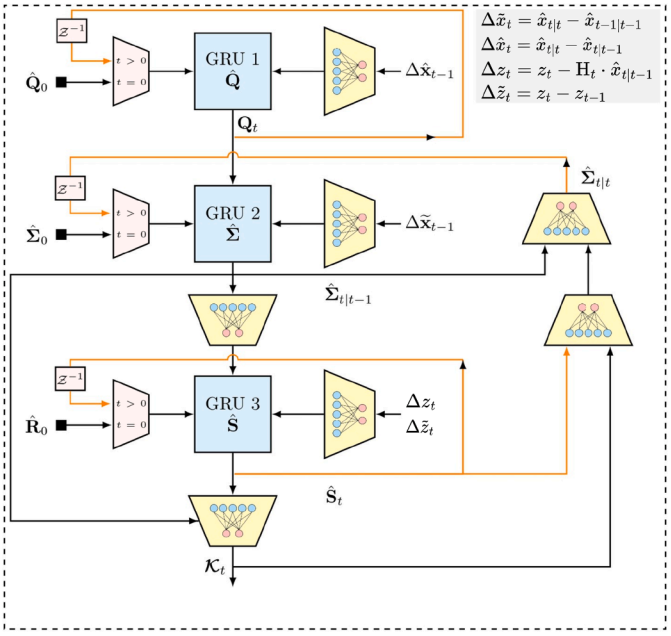

Fig. 2. KalmanNet architecture configuration #2. Image adapted from [25].

components, based on the available sequence of measurements up to time t by the sensor $z_{0:t} = \{z_0, \dots, z_t\}$. The Bayesian recursive framework identifies two consecutive steps for measurement filtering, i.e., prediction and update steps. The prediction step models the target dynamics, expressing the probability density function of the target's state from $t-1$ to t conditioned on the available information at time $t-1$, regardless of the sensor's measurement at time t, namely:

$$p(x_t|x_{t-1}, z_{0:t-1}) = \int p(x_t|x_{t-1})p(x_{t-1}|z_{0:t-1})dx_{t-1} \tag{1}$$

known as the *Chapmann-Kolmogorov equation*.

The predicted state is then updated considering the incoming sensor measurements at time t by using the Bayes' theorem, yielding:

$$p(x_t|z_{0:t}) \propto p(z_t|x_t)p(x_t|x_{t-1}) \tag{2}$$

The Kalman Filter poses some assumptions on the problem to find an analytical solution to eq. (1), which is intractable: the process is a first-order Markovian chain, meaning that the transition from $t-1$ to t only depends on the information in $t-1$ and not on the previous timesteps; states are distributed as Gaussian. Thus, the two-step Bayesian recursive filtering framework is replaced with the following equations, describing a KF recursion:

1. Predict:

$$\hat{x}_{t|t-1} = \mathrm{F}_{t-1} \cdot \hat{x}_{t-1|t-1} \tag{3}$$
$$\hat{\Sigma}_{t|t-1} = \mathrm{F}_{t-1}\hat{\Sigma}_{t-1|t-1}\mathrm{F}_{t-1}^{\top} + \mathrm{Q}_{t-1} \tag{4}$$

2. Update:

$$\Delta z_t = z_t - \mathrm{H}_{t-1} \cdot \hat{x}_{t|t-1} \tag{5}$$
$$\hat{\mathrm{S}}_{t|t-1} = \mathrm{H}_{t-1}\hat{\Sigma}_{t|t-1}\mathrm{H}_{t-1}^{\top} + \mathrm{R}_{t-1} \tag{6}$$
$$\mathcal{K}_t = \hat{\Sigma}_{t|t-1}\mathrm{H}_{t-1}^{\top}\hat{\mathrm{S}}_{t|t-1}^{-1} \tag{7}$$
$$\hat{x}_{t|t} = \hat{x}_{t|t-1} + \mathcal{K}_t \cdot \Delta z_t \tag{8}$$
$$\hat{\Sigma}_{t|t} = \hat{\Sigma}_{t|t-1} - \mathcal{K}_t\hat{\mathrm{S}}_{t|t-1}\mathcal{K}_t^{\top} \tag{9}$$

The IMM filter considers multiple models, as KFs running in parallel, named *target modes*, to compensate for possible model mismatches while tracking maneuvering targets. The accurate computation of the target modes' probabilities is crucial for mixing their state predictions properly. Two additional steps are introduced: *interaction step* before the prediction step that mixes the updated state at time $t-1$ with the modes' probabilities, *modes' probabilities update step* after individual filters' update step to compute the conditional likelihood of each mode based on the current sensor measurement and adjust the probabilities accordingly. That results in a weighted combination of the target modes.

The KalmanNet proposes a data-driven adaptation to the KF framework, which can compensate for misconfigured parameter identification in the definition of the state transition variance or the measurement likelihood model variance, i.e., the matrices Q and R. The original formulation proposes two architectures whose scope is to compute an adaptive Kalman Gain $\mathcal{K}_t$ exploiting GRU's ability to extract temporal relationships [25]. They differ in the computation of the variances that affect state prediction and measurement uncertainty, which are fixed and defined *a priori* in architecture 1, shown in Fig. 1. At the same time, specific GRUs are designed to keep track of them during filtering in architecture number 2, displayed in Fig. 2. We adopted the KalmanNet in configuration #1 as the Ensemble's base filter due to its improved ability when considering linear models [25], as we did in our experiments in Section 4.

3. Ensemble of data driven Kalman filters for MTT

In this section, we first revisit the architecture proposed in [24] and introduce a novel attention mechanism, inspired by the IMM workflow, based on normalized squared innovations derived from MB filters. This mechanism further enhances both classification and tracking robustness in complex scenarios while providing interpretable target mode classification.

3.1. Ensemble of KalmanNets

The Ensemble architecture integrates predictions from multiple KalmanNets, each using a different motion model to filter sensor measurements. The block diagram of the computational graph is illustrated in Fig. 3. The architecture consists of two task-specific branches: the *correction* branch and the *classification* branch.

Let $\mathcal{M} = \{\mathrm{CV}, \mathrm{CT}_{\max}, \mathrm{CT}_{\min}\}$ be the set of motion models used by KalmanNets for filtering sensor measurements. Here, CV represents a nearly Constant Velocity model, while CT describes a Coordinated Turn motion with a fixed turn rate in the xy-plane. The notations $\mathrm{CT}_{\max}$ and $\mathrm{CT}_{\min}$ correspond to turn rates of $+10°/s$ and $-10°/s$, respectively.

In the correction branch, sensor measurements are first filtered by the KalmanNets, each producing an *a posteriori* state estimate $\hat{x}^m_{t|t}$, $m \in \mathcal{M}$. These estimates are concatenated along the state dimension, normalized using Instance Normalization (IN) [26], and processed by an LSTM to extract temporal dependencies. IN ensures that input features remain normalized while preserving temporal variations, making it particularly suitable for tracking tasks.

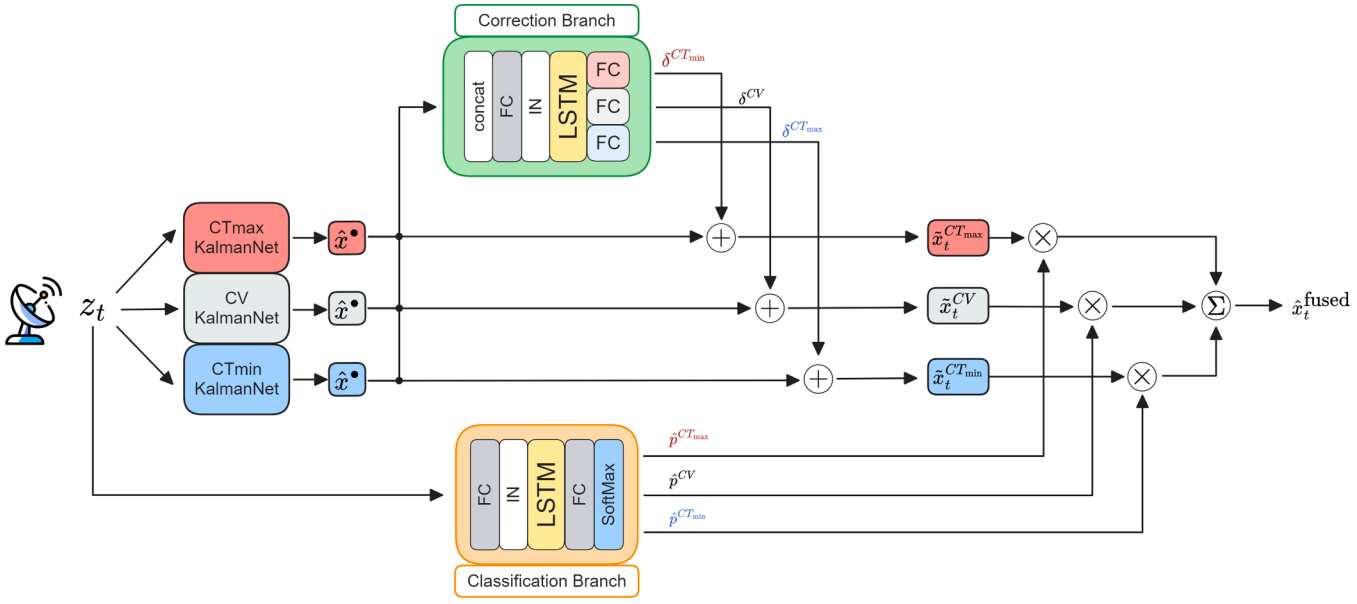

Fig. 3. Ensemble model with track correction and motion mode estimation. The *classification branch*, which processes raw sensor measurements, is placed in the lower branch, while the *correction branch*, which processes posterior state estimates from the KalmanNets, is placed in the upper branch. Image adapted from [24].

To mitigate model mismatches in KalmanNet predictions, a fully connected (FC) layer is assigned to each KalmanNet to process the LSTM's hidden state, and generate correction terms $\delta_t^m,\ m \in \mathcal{M}$. This design allows each FC layer to specialize in refining the state estimates of a specific motion model. The corrected state estimates are then computed as follows:

$$\tilde{x}_t^m = \hat{x}_{t|t}^m + \delta_t^m, \quad m \in \mathcal{M}. \tag{10}$$

In parallel, the classification branch processes the raw sensor measurements. These are normalized using IN and passed through an LSTM module followed by an FC layer, which predicts the probability of each motion mode, $\hat{p}_t^m,\ m \in \mathcal{M}$.

The final fused state estimate is obtained by computing a weighted sum of the corrected state estimates, using the predicted probabilities as weights:

$$\hat{x}_t^{\text{fused}} = \sum_{m \in \mathcal{M}} \hat{p}_t^m \cdot \tilde{x}_t^m. \tag{11}$$

where $\sum_{m \in \mathcal{M}} \hat{p}_t^m = 1$, after normalization by Softmax operator.

While KalmanNets improve over traditional filtering methods, such as the KF, they remain susceptible to model mismatch during target maneuvers, which can degrade their learned adaptability. If these mismatches are not promptly detected and corrected, tracking performance may suffer. As in the IMM filter, rapid maneuver detection and adaptation are critical. However, traditional IMM filtering often falls short in demanding applications.

The proposed architecture leverages the ability of LSTMs to capture complex temporal dependencies, enhancing the prediction of correction terms to counteract model mismatches and dynamically adapt to target maneuvers. By integrating both correction and classification, the architecture not only improves the accuracy of individual KalmanNets but also maintains interpretability by preserving explicit motion mode selection. Moreover, the flexibility to include or remove models makes the system adaptable to different tracking scenarios, whether dealing with highly maneuverable or slower-moving targets.

We will refer to this architecture as "EnsKNets" below.

3.2. Innovation-based attention mechanism

To preserve the interpretability of the IMM workflow, we introduce the *innovation-based attention* mechanism, as illustrated in Fig. 4, in the Ensemble of KalmanNets filtering method.

Drawing inspiration from the IMM filter, we designed an attention mechanism based on the *normalized squared innovation* at the current timestep, given by:

$$\bar{\Delta} z_t^m = (\Delta z_t^m)^\top \cdot (\hat{\mathrm{S}}_{t|t-1}^m)^{-1} \cdot \Delta z_t^m \tag{12}$$

where $m \in \mathcal{M}$, Δz_t^m is the innovation term, and $\hat{\mathrm{S}}_{t|t-1}^m$ is the innovation covariance, which can be derived from each KalmanNet during their processing through Eqs. (5) and (6), respectively.

Rather than processing raw sensor measurements and KalmanNets posterior state estimates, in the case of the classification and correction branches, respectively, we compute the attentioned posterior state estimate as the input for both branches as follows:

$$x_t^c = \sum_{m \in \mathcal{M}} \mathrm{Softmax}\left(\bar{\Delta} z_t^m\right) \cdot \hat{x}_{t|t}^m \tag{13}$$

for $t = 1, \ldots, T$.

This innovation-based attention mechanism enhances the explainability of the neural filter by providing an interpretable, distance-based attention strategy for motion mode selection. It is inspired by the mode likelihood computation in the IMM framework during the update of mode probabilities, where the likelihoods $\mathcal{L}_t^m$ of each mode are computed as:

$$\mathcal{L}_t^m = \frac{\exp(-\frac{1}{2}(\Delta z_t^m)^\top \cdot (\hat{\mathrm{S}}_{t|t-1}^m)^{-1} \cdot \Delta z_t^m)}{2\pi^{m/2} \cdot |\hat{\mathrm{S}}_{t|t-1}^m|^{1/2}} \tag{14}$$

where $m \in \mathcal{M}$ and $|\hat{\mathrm{S}}_{t|t-1}^m|$ is the determinant of the m-th innovation covariance matrix. The purpose of this attention mechanism is to generate contextual information x_t^c that emphasizes the most probable motion mode in the Bayesian sense, that is, by fusing the *a priori* state estimates $\hat{x}_{t|t-1}$ produced by each motion model with the current sensor observation z_t. Moreover, by processing posterior state estimates rather than raw sensor measurements, the classification branch is allowed to focus solely on predicting mode probabilities, rather than learning the noise characteristics that are already filtered out by the KalmanNets.

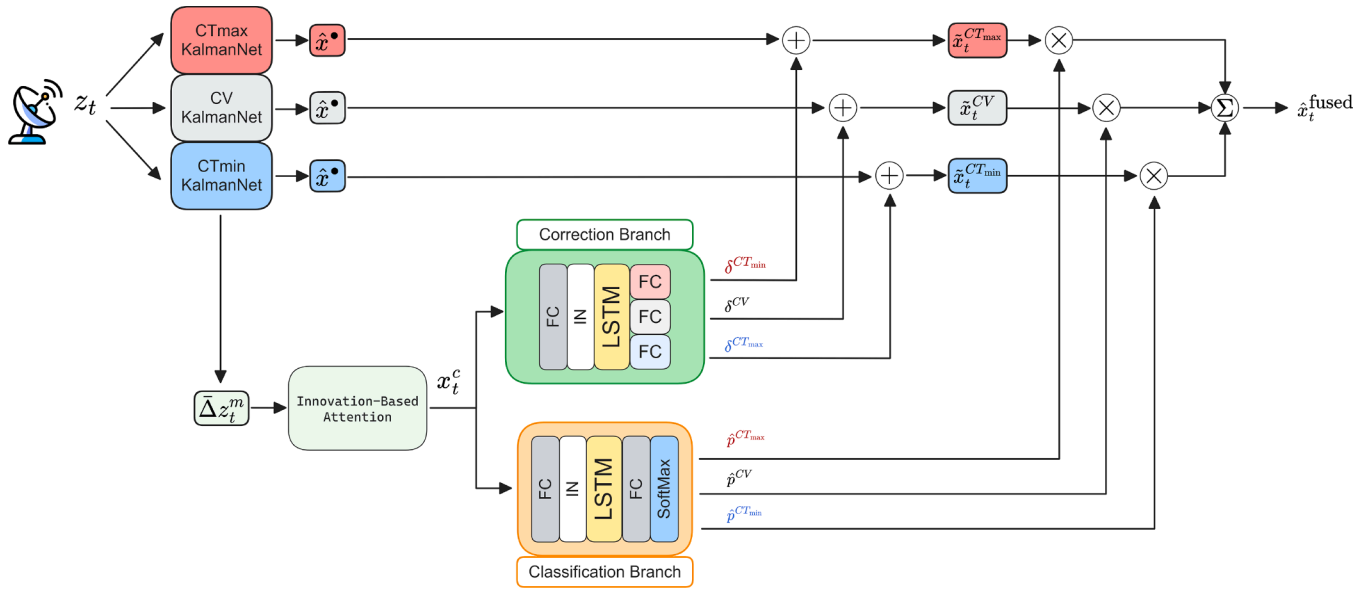

Fig. 4. AttEnsKnets model for the ensemble of KalmanNets state estimates. The major difference from the architecture in Fig. 3 lies in the introduction of the innovation-based attention mechanism that produces contextual information focusing Ensemble's correction and classification branches on the most likely motion model in a Bayesian sense at each timestep.

As in the previous architecture in Section 3.1, KalmanNet posterior state estimates are refined by computing correction terms δ_t^m, $m \in \mathcal{M}$. The key difference, however, is that these corrections are now generated based on the contextual information x_t^c, rather than on the concatenation of posterior state estimates from the KalmanNets.

We will refer to this architecture as "AttEnsKNets" below, highlighting the introduction of the attention mechanism.

3.3. Loss function for multi-task learning

To accurately predict correction terms and motion modes probabilities, which are configured as a regression and a classification task, respectively, a combined, multi-task loss is designed. Therefore, the loss function will comprise a regression term aiming at reducing the gap between the overall state predictions and the ground truth states, and a classification term to properly predict the target's motion mode.

Multi-Task Learning (MTL) is a machine learning paradigm that seeks to improve the generalization performance of multiple related tasks by learning them jointly. The central idea is to leverage the shared information embedded in related tasks to develop more accurate models for each task. Under the assumption that the tasks are related, either all or a meaningful subset, joint learning has been shown both empirically and theoretically to outperform learning tasks in isolation. MTL serves as a form of inductive transfer, allowing knowledge gained from one task to benefit the learning of others. This approach is particularly effective in settings where tasks share underlying structures or representations, enabling improved data efficiency, robustness, and predictive performance across tasks [27,28]

Let x_t^{gt} be the ground truth target state at time t, the regression task loss penalizes the Mean Squared Error (MSE) between the fused state prediction $\hat{x}_t^{\text{fused}}$ and the ground truth target state:

$$\mathcal{L}_{\text{regr}} = \frac{1}{NT}\sum_{n=1}^{N}\sum_{t=1}^{T}(\hat{x}_{n,t}^{\text{fused}} - x_{n,t}^{\text{gt}})^{\top}\cdot(\hat{x}_{n,t}^{\text{fused}} - x_{n,t}^{\text{gt}}) \tag{15}$$

where N is the number of samples, and T is the total number of time steps in the n-th trajectory.

Regarding the classification of the target mode probabilities, Cross Entropy (CE) loss has been used, which has the following form:

$$\mathcal{L}_{\text{ce}} = -\frac{1}{N}\sum_{m\in\mathcal{M}}\sum_{t=1}^{T}\bar{y}_t^m \cdot \log\left(\hat{p}_t^m\right), \tag{16}$$

where $\bar{y}_t^m$ is the one-hot encoding of the actual target mode at time t.

However, due to the potential disparity in scale between the MSE used for the regression task and the CE loss employed for the classification task, the gradients associated with the regression objective may dominate the optimization of the combined loss function. This imbalance can lead the model to prioritize minimizing the MSE of the estimated target state, irrespective of correctly identifying the target's motion mode. To address this issue, normalizing the losses of each task brings their values to a comparable scale, thereby mitigating bias in the learning process and promoting a more balanced optimization across tasks.

To this end, we first define a baseline value for both the regression and classification tasks. Let $\bar{\mathcal{L}}_{\text{ce}}$ denote the CE loss when $\hat{p}_t^m = 1/|\mathcal{M}|$, $\forall m, t$, and let $\bar{\mathcal{L}}_{\text{mse}}$ denote the MSE obtained by averaging the outputs of the KalmanNets without applying any correction. The normalized CE is thus defined as:

$$\tilde{\mathcal{L}}_{\text{ce}} = \frac{\mathcal{L}_{\text{ce}}}{\bar{\mathcal{L}}_{\text{ce}}}, \tag{17}$$

and the normalized MSE assumes the form:

$$\tilde{\mathcal{L}}_{\text{regr}} = \frac{\mathcal{L}_{\text{regr}}}{\bar{\mathcal{L}}_{\text{mse}}}. \tag{18}$$

The overall loss function is thus a linear combination of the two task losses, resulting in:

$$\mathcal{L} = \tilde{\mathcal{L}}_{\text{regr}} + \lambda_{\text{ce}} \cdot \tilde{\mathcal{L}}_{\text{ce}} \tag{19}$$

being λ_{ce} the weighting coefficient of the classification task, which provides a tuning parameter to decide on the importance associated with the classification task.

4. Experiments and results

Filtering methods are evaluated in Root Mean Squared Error (RMSE), to provide a practical measure of the filtering accuracy, computed as

Table 1
Simulation parameters ranges for training, validation, and test set generation.

Parameter	Training Set	Validation Set	Test Set
N	50000	5000	1000
T	200 s	200 s	200 s
Δt	1 s	1 s	1 s
number of switches	[1; 4]	[1; 5]	[1; 6]
σ_q	[8.; 13.] m/s^2	[5.; 15.] m/s^2	[5.; 20.] m/s^2
σ_r	10, 20, …, 100 m	10, 20, …, 100 m	10, 20, …, 100 m
σ_θ	0.10, 0.20, …, 1.0 °	0.10, 0.20, …, 1.0 °	0.10, 0.20, …, 1.0 °
Initial distance	[50; 150] km	[50; 150] km	[50; 150] km
Initial azimuth	[−180; +180] °	[−180; +180] °	[−180; +180] °
Initial velocity	[50; 400] m/s	[50; 400] m/s	[50; 400] m/s
ω	[−10; +10] °/s	[−10; +10] °/s	[−12; +12] °/s

follows:

$$\mathrm{RMSE}_n = \sqrt{\frac{1}{T}\sum_{t=1}^{T}\left(\hat{x}_{n,t} - x_{n,t}^{\mathrm{gt}}\right)^2} \tag{20}$$

where $n = 1, \ldots, N$ is the n-th sample in the set.

Furthermore, accuracy is used to assess the correctness in predicting the target's motion mode for multiple models filtering methods, and it follows:

$$\mathrm{ACC}_n = \frac{100}{T}\sum_{t=1}^{T}\mathbb{1}(\hat{y}_{n,t} = y_{n,t}) \tag{21}$$

where $\hat{y}_{n,t}$ is the predicted target mode in timestep t, $y_{n,t}$ is the true one, and $\mathbb{1}$ is an indicator function that returns 1 whether the condition in brackets is satisfied.

In the following experiments, the first 10 timesteps are excluded from the computation of the metrics to allow the IMM filter sufficient time to converge from its initial transient phase.

A large corpus of 2D trajectories is generated using SS models and simulation parameters inspired by real ATC scenarios. Although aircraft operate in the 3D space, their primary maneuvers occur in the xy-plane, and the z-axis can be neglected for tracking purposes. The simulations assume the targets can perform a CV or a CT motion model, whose equations, reported below, are derived from [29].

Let $x_t = [p_{xt},\ v_{xt},\ p_{yt},\ v_{yt}]^T$ be the state vector in the xy-plane at time t describing the position and velocity in Cartesian components, and let Δt denote the time interval between consecutive timesteps. From [30], we define the CV motion model as:

$$x_t = \mathrm{F}_{cv}\cdot x_{t-1} + v_t,\ v_t \sim \mathcal{N}(\mathbf{0}, \mathrm{Q}_{cv}) \tag{22}$$

$$\mathrm{F}_{cv} = \mathbf{I}_2 \otimes \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \tag{23}$$

$$\mathrm{Q}_{cv} = \mathbf{I}_2 \otimes \begin{bmatrix} \frac{\Delta t^4}{4} & \frac{\Delta t^3}{2} \\ \frac{\Delta t^3}{2} & \Delta t^2 \end{bmatrix} \cdot \sigma_q^2 \tag{24}$$

being $\mathbf{I}_2$ the identity matrix, $\otimes$ the Kronecker product, and σ_q^2 describes the variance of unmodeled acceleration. The CT motion model refers to the Coordinated Turn with a fixed turn rate and follows:

$$x_t = \mathrm{F}_{ct}\cdot x_{t-1} + v_t,\ v_t \sim \mathcal{N}(\mathbf{0}, \mathrm{Q}_{ct}) \tag{25}$$

$$\mathrm{F}_{ct} = \begin{bmatrix} 1 & \frac{\sin\omega\Delta t}{\omega} & 0 & \frac{\cos\omega\Delta t}{\omega} \\ 0 & \cos\omega\Delta t & 0 & -\sin\omega\Delta t \\ 0 & \frac{1-\cos\omega\Delta t}{\omega} & 1 & \frac{\sin\omega\Delta t}{\omega} \\ 0 & \sin\omega\Delta t & 0 & \cos\omega\Delta t \end{bmatrix} \tag{26}$$

$$\mathrm{Q}_{ct} = \mathbf{I}_2 \otimes \begin{bmatrix} \frac{\Delta t^4}{4} & \frac{\Delta t^3}{2} \\ \frac{\Delta t^3}{2} & \Delta t^2 \end{bmatrix} \cdot \sigma_q^2 \tag{27}$$

where ω represents the turn rate.

The radar-based sensor measurements capture target range and azimuth, and are modeled as follows:

$$\begin{bmatrix} \rho_t \\ \theta_t \end{bmatrix} = \begin{bmatrix} \sqrt{p_{xt}^2 + p_{yt}^2} \\ \mathrm{atan2}(p_{yt}, p_{xt}) \end{bmatrix} + u_t,\ u_t \sim \mathcal{N}(\mathbf{0}, \mathrm{R}) \tag{28}$$

$$\mathrm{R} = \begin{bmatrix} \sigma_r^2 & 0 \\ 0 & \sigma_\theta^2 \end{bmatrix} \tag{29}$$

$$z_t = \begin{bmatrix} \rho_t \cos\theta_t \\ \rho_t \sin\theta_t \end{bmatrix} \tag{30}$$

where σ_r, σ_θ, are the standard deviations of the range and azimuth, respectively. The parameters for the generation of the training, validation, and test sets trajectories and their respective measurements are listed in Table 1.

We trained three KalmanNets that constitute base filters for the Ensemble, each of them uses a CV motion model (CV KalmanNet), a CT model with turn rate $\omega = +10$ °/s (named $\mathrm{CT}_{\max}$ KalmanNet), and a CT model with turn rate $\omega = -10$ °/s ($\mathrm{CT}_{\min}$ KalmanNet), respectively.

KalmanNets are trained using AdamW [31] optimizer, with learning rate $lr = 10^{-4}$ and weight decay $wd = 10^{-2}$ for 10 epochs having a batch size of 128 samples. Trained KalmanNets are integrated into the architectures in Figs. 3 and 4. The AdamW optimizer has also been used for both architectures with $lr = 5 \cdot 10^{-4}$ and $wd = 10^{-2}$.

As in the case of the IMM filter, we treat the classification of the target's motion mode as an auxiliary task, acknowledging that the selection of the appropriate motion model is inherently prone to uncertainty. For instance, different turn rates may be selected for CT motion models, or models that account for target acceleration may be included within the filter bank. To reflect the auxiliary nature of this task and control its influence on the overall optimization, we set $\lambda_{\mathrm{ce}} = 0.1$.

For both the architectures in Figs. 3 and 4, the LSTMs have both 64 neurons and stack 2 recurrent cells on top of each other; we used dropout [32] with probability 0.2 to prevent overfitting after each LSTM layer.

For both EnsKNets and EnsKNetsAtt, training took approximately 2 h for 500 epochs on an NVIDIA RTX A5000 GPU.

In the experimental evaluation, the proposed approach is compared against the IMM filter and several state-of-the-art data-driven filters. These include both filters integrated within the Bayesian recursive estimation framework, such as DeepMTT [10] and Mnemonic KF [16], and end-to-end approaches, such as TrMTT [13].

The uncertainty parameters of all MB filters, namely the process noise covariance Q and the measurement noise covariance R, are configured based on the guidelines provided in [29] and [33]. Specifically, we set $\sigma_q = 10.$ and $\sigma_r = \sqrt{1250}$ for all motion and measurement models used by both the KalmanNets and the IMM filter. In contrast, the UKF employed in the DeepMTT framework processes sensor measurements in polar coordinates and uses the following measurement noise covariance matrix:

$$R = \begin{bmatrix} 50^2 & 0. \\ 0. & 0.5^2 \end{bmatrix}$$

Table 2
Summary table of metrics in the validation set for the ablation studies in Section 4.1 with empirical mean μ and standard deviation σ per configuration.

	RMSE all ↓		RMSE pos [m] ↓		RMSE vel [m/s] ↓		ACC % ↑	
	μ	σ	μ	σ	μ	σ	μ	σ
Avg KalmanNets	249.01	129.34	341.25	184.07	79.48	28.77	33.25	30.72
Classification Branch	240.57	126.27	330.25	179.19	74.71	29.69	72.93	18.43
Correction Branch	228.65	105.35	317.10	149.11	59.61	20.54	33.25	30.72
EnsKNets [24]	218.17	97.72	302.14	138.41	58.78	19.70	74.57	17.05
AttEnsKNets	**206.05**	100.04	**285.30**	141.88	**55.28**	18.81	**79.12**	17.23

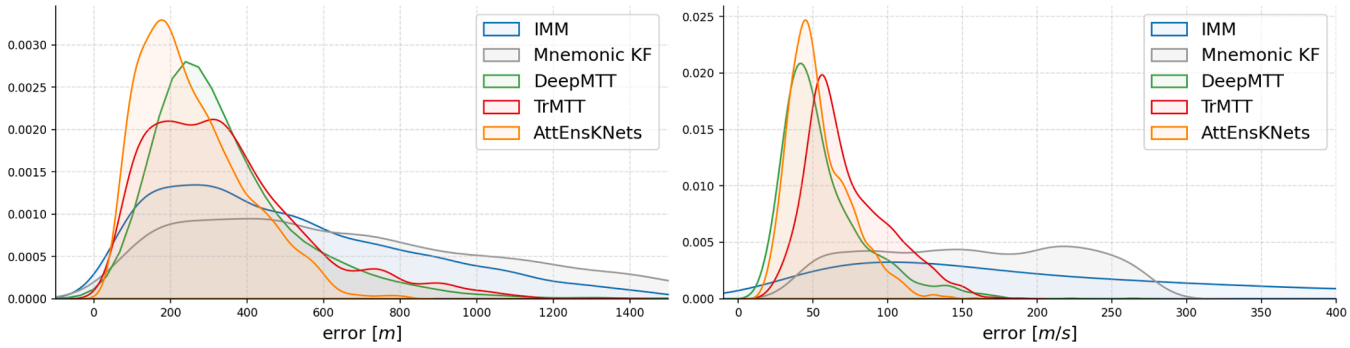

Fig. 5. RMSE distribution in position and velocity estimation per filtering method when tracking trajectories in the test set of Section 4.

Moreover, the transition probability matrix for the IMM filter is set to

$$\Pi = \begin{bmatrix} 0.95 & 0.025 & 0.025 \\ 0.05 & 0.7 & 0.25 \\ 0.05 & 0.25 & 0.7 \end{bmatrix}$$

The number of layers and hidden units for DeepMTT are adopted from [10], along with the sliding window configuration, which is set to 5 timesteps and advances by 1 timestep at a time, as originally described. The architecture of the Mnemonic Kalman Filter (Mnemonic KF) follows the specifications in [16] with respect to the number of layers and neurons used in the LSTM. Similarly, the architecture of TrMTT is based on [13]. However, in the original formulation, the sliding window spans 100 timesteps and slides forward by 25 timesteps, which would require maintaining a large measurement buffer and result in a delay of 100 s before producing the first prediction. To alleviate this limitation, we configure the sliding window to cover 25 timesteps, sliding by 5 timesteps at a time.

Notably, this requirement for measurement accumulation constitutes a practical limitation for both DeepMTT and TrMTT, whereas it does not affect filters that operate on a per-timestep basis.

All data-driven methods are trained on the same training set, and the parameter setting providing the best validation loss is saved and used for testing purposes.

4.1. Ablation studies

We assessed the effectiveness of the proposed approach through a series of ablation experiments. Specifically, we evaluated four different configurations to isolate the contributions of each component to the overall performance:

(I) an ensemble that relies solely on the classification of the target motion mode, named "Classification Branch" in Table 2 accordingly;
(II) the model without the correction branch, thus named "Correction Branch" in Table 2;
(III) the original Ensemble of KalmanNets as proposed in [24];
(IV) the full model including the innovation-based attention mechanism described in Section 3.2.

These four configurations are compared against a baseline in which the outputs of all KalmanNets are simply averaged at each timestep, without leveraging either the classification or correction branches. This baseline, referred to as "Avg KalmanNets," represents a naïve ensemble strategy with no adaptive weighting or task-specific enhancement. When the classification task is excluded from a given configuration, i.e., when only the correction branch is active, we assume uniform mode probabilities across the ensemble components, assigning equal likelihood to all target motion modes. The results on the validation set, summarized in Table 2, demonstrate that the incorporation of the innovation-based attention mechanism significantly improves both robustness and generalization. Not only does this enhancement lead to a lower RMSE in state estimation, but it also results in improved accuracy in selecting the correct target motion mode, highlighting the benefit of integrating mode-aware contextual information into the ensemble.

4.2. Evaluation

By analyzing the distribution of the RMSE for the filtering methods operating on the test set trajectories described in Section 4, displayed in Fig. 5 and whose estimated mean and standard deviation are summarized in Table 3, the proposed AttEnsKNets demonstrated overall superior performance in both position and velocity estimation, as well as, notably, a better estimation of the target motion mode when compared to the IMM filter. Notably, DeepMTT shows competitive performance in the estimation of the target velocity, but performs worse in the estimation of the position when compared to the proposed method.

The IMM filter exhibits higher variance in its RMSE distribution compared to both ensemble-based methods, as illustrated in Fig. 5. This increased variability likely arises from the suboptimal adaptation of its fixed, predefined parameters when applied to the diverse set of target trajectories observed under heterogeneous sensor configurations in the test set. Although these parameters may yield satisfactory performance in certain controlled scenarios, they tend to lack robustness and generalizability across a broader range of operational conditions. In contrast, hybrid MB/data-driven and purely end-to-end methods offer intrinsic advantages by eliminating the need for manual and precise parameter initialization. Their data-driven architecture facilitates automatic

Table 3
Summary table of metrics when tracking trajectories in the test set of Section 4 with empirical mean μ and standard deviation σ per filtering method. For filtering methods that do not account for target motion mode estimation, accuracy (ACC) is not available (n.a.).

	RMSE all ↓		RMSE pos [m] ↓		RMSE vel [m/s] ↓		ACC % ↑	
	μ	σ	μ	σ	μ	σ	μ	σ
IMM	397.92	271.54	505.94	336.87	244.75	186.53	40.61	19.70
Mnemonic KF [16]	507.24	294.53	685.85	434.06	157.01	68.10	n.a.	
DeepMTT [10]	255.59	231.74	356.30	326.96	57.53	29.82	n.a.	
TrMTT [13]	251.39	145.95	345.63	208.38	72.97	28.07	n.a.	
AttEnsKNets [ours]	**187.18**	96.93	**258.18**	137.24	**54.19**	20.86	**81.99**	14.42

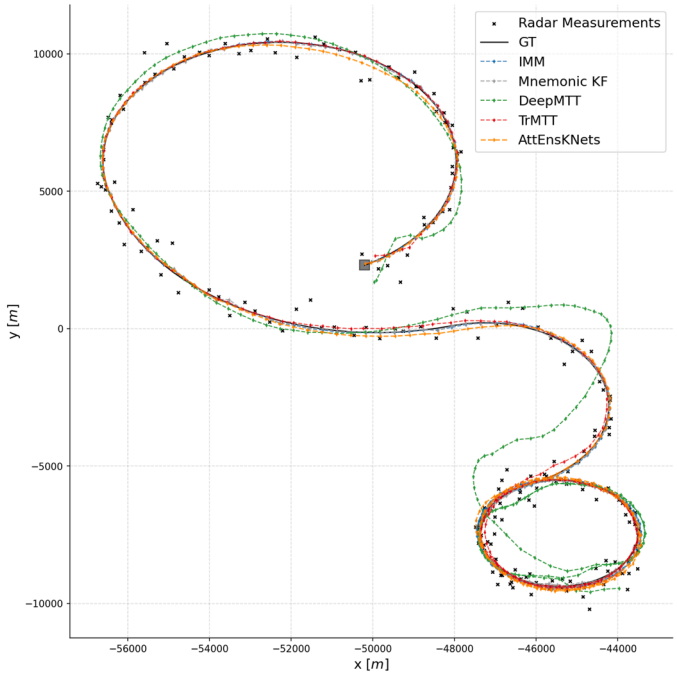

Fig. 6. Target tracking result from the test set in Section 4 on the xy-plane.

adaptationm to varying sensing environments, enabling more stable and consistent performance across the full spectrum of test conditions.

In Fig. 6, a sample trajectory from the test set in Section 4 with resulting tracks from the considered filtering methods is displayed.

Fig. 7a presents the position RMSE over the simulation time for the tracks generated by the filtering methods on 100 Monte-Carlo (MC) simulations of the trajectory depicted in Fig. 6. Similarly, Fig. 7b illustrates the RMSE of the estimated target velocity for the same trajectory. In both position and velocity estimation, AttEnsKNets demonstrates overall superior performance. It is worth noting that DeepMTT shows higher accuracy in velocity estimation than in position estimation, and it outperforms other filtering methods during certain intervals of the trajectory. However, its performance degrades when the target's motion deviates significantly from the motion model assumed by the underlying UKF responsible for filtering, that is, the CV mode.

Overall, AttEnsKNets achieves the lowest average RMSE in estimating the target state–considering both position and velocity-across 100 MC simulations of the evaluated trajectories. Specifically, the average RMSE values for each method are as follows: 165.15 for IMM, 275.22 for Mnemonic KF, 551.91 for DeepMTT, 136.88 for TrMTT, and 130.36 for AttEnsKNets. s

Finally, Fig. 8 shows the motion mode classification for the IMM and AttEnsKNets. In summary, the data-driven classification of the target's motion mode exhibits superior accuracy across the simulation compared to the IMM filter, with the AttEnsKNets providing the most accurate performance on the trajectory in Fig. 6, reaching 84.50% of accuracy, and the IMM filter, which only estimated the correct motion mode in 47.50% of the whole simulation.

4.3. Noise robustness analysis

In this section, robustness to sensor noise is evaluated by analyzing the methods' performance over the same test set while isolating the noise component. That is, the $N = 1000$ generated trajectories in the test set of Section 4 share identical dynamics, in terms of mode switching, initial position and velocity, and unmodeled acceleration, but measurements are performed using a single combination of sensor noise parameters, i.e., σ_r and σ_θ, at a time. This enables a comprehensive analysis of tracking performance under increasing noise levels, which we believe provides valuable insights into the robustness of the tracking systems.

It is well established in the literature [29,34] that measurement errors in the polar coordinate system do not translate uniformly into the Cartesian frame. Specifically, the error propagation differs between the range and cross-range directions when measurements are transformed into Cartesian components. Neglecting the cross-covariance terms, one can derive a linear relationship between the variances along the Cartesian axes and the variance in the range measurement, i.e., $\sigma_x^2, \sigma_y^2 \propto \sigma_r^2$. However, when accounting for azimuth uncertainty, the Cartesian variances become dependent on the target's distance, such that $\sigma_x^2, \sigma_y^2 \propto \rho^2 \cdot \sigma_\theta^2$. This implies that perturbations in azimuth measurements are significantly amplified for long-range targets, a condition representative of

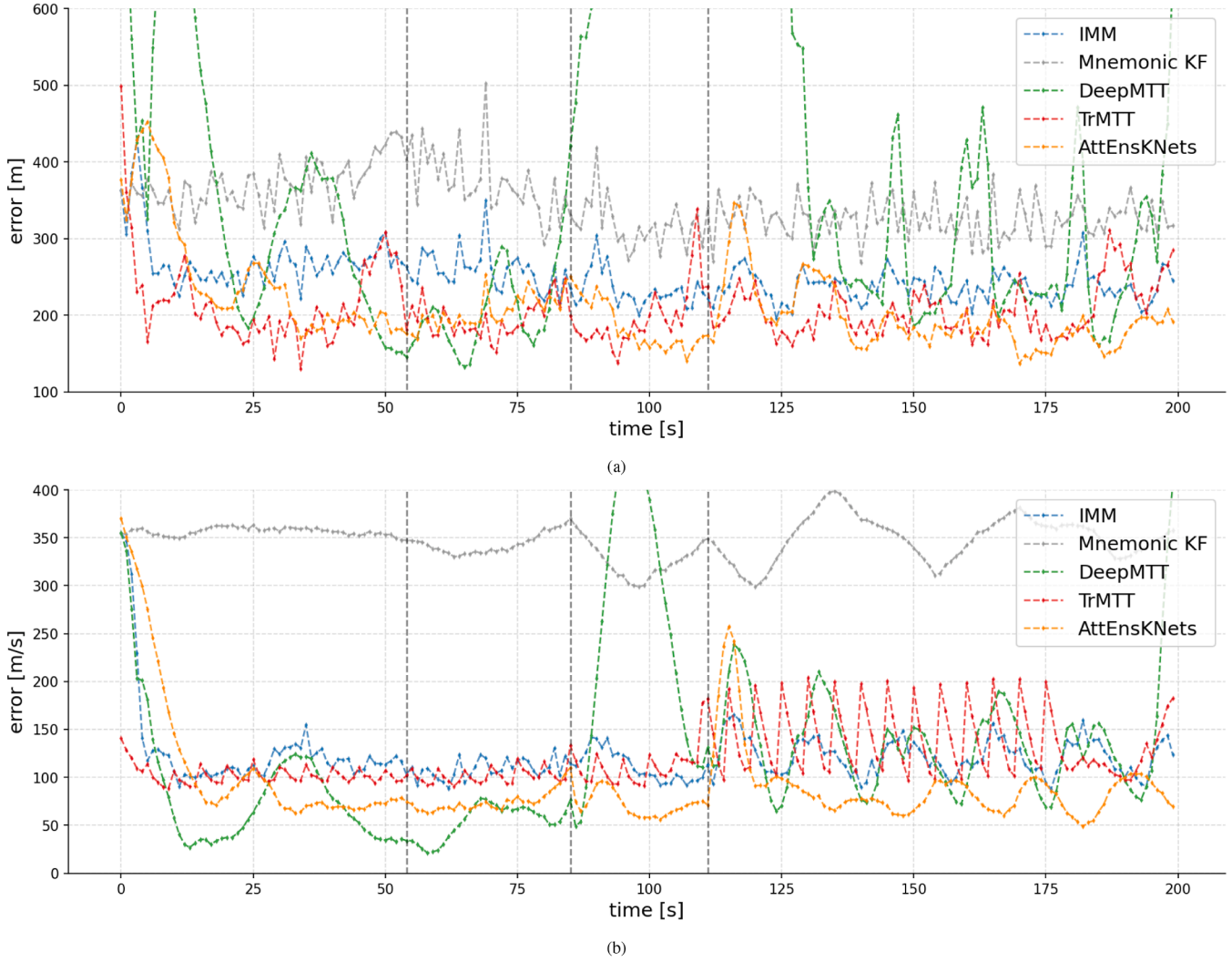

Fig. 7. Tracking results over 100 Monte-Carlo runs for the trajectory example in Section 4 and displayed in Fig. 6, in particular: (a) RMSE in position estimation of each filtering method per timestep; (b) RMSE in velocity estimation of each filtering method per timestep. The vertical dashed line indicates either a change in the target's motion mode or, in cases where the mode remains the same, a change in the turn rate.

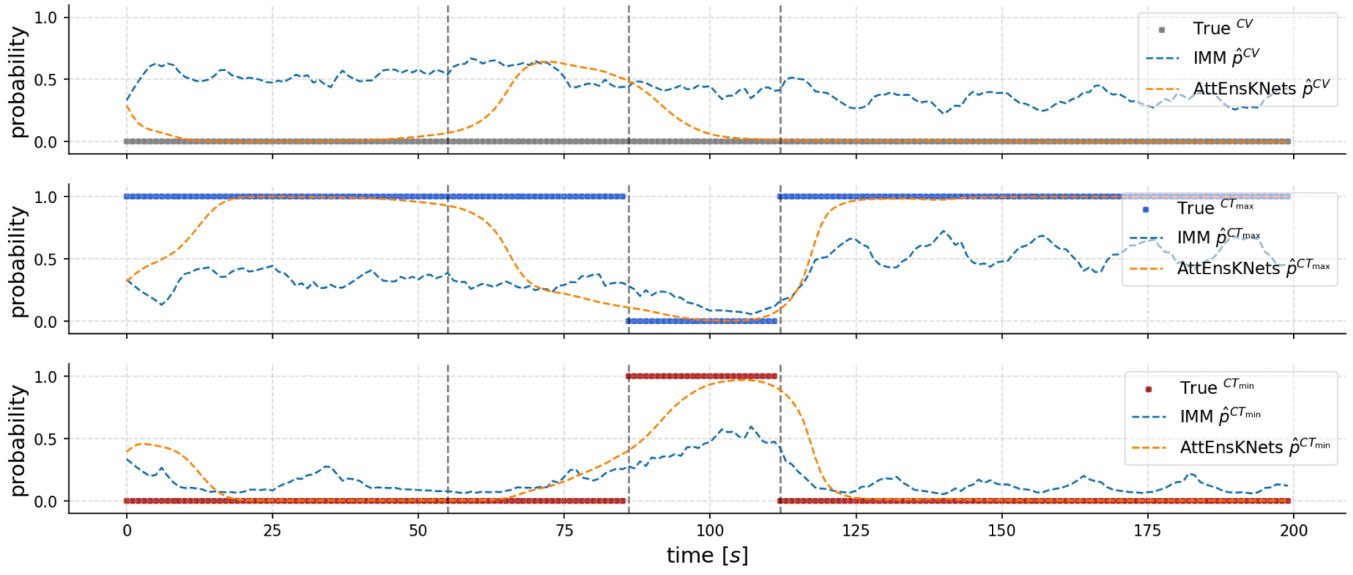

Fig. 8. Target's motion mode probabilities estimation at each timestep of the trajectory in Fig. 6 from the test set in Section 4 for IMM and AttEnsKNets.

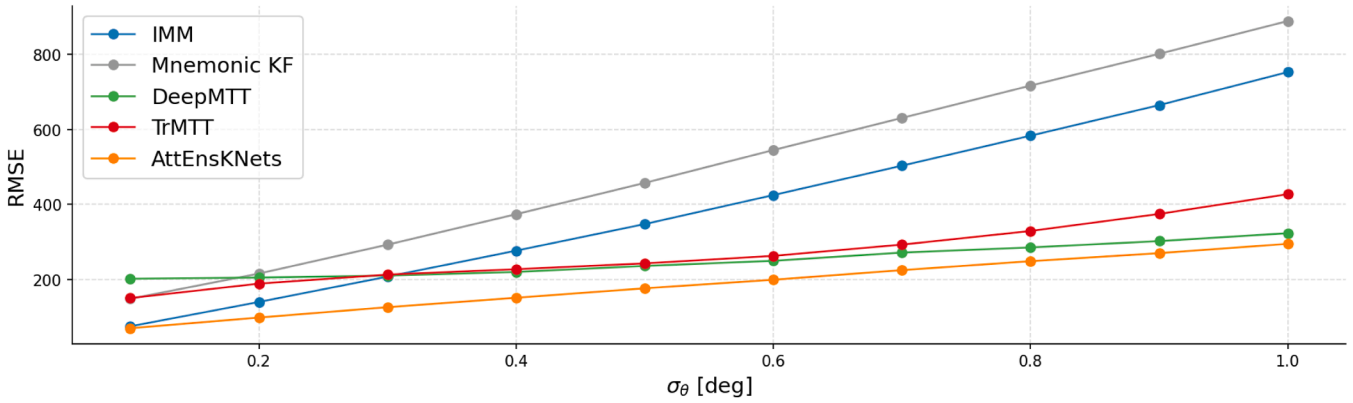

Fig. 9. Average RMSE against increasing azimuth noise levels for each filtering method tracking trajectories in the test set of Section 4.

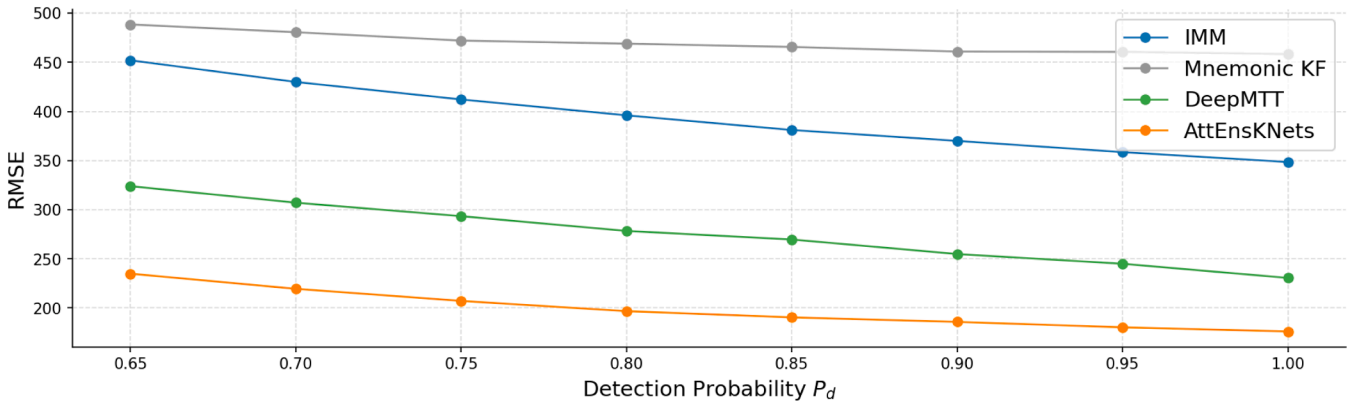

Fig. 10. Average RMSE against decreasing target detection probability for the test set in Section 4 for each filtering method.

the settings described in Section 4, compared to those originating from range measurement noise.

As a consequence, analyzing the RMSE performance of the filtering methods with respect to variations in the range noise variance σ_r^2, while keeping the azimuth variance σ_θ^2 fixed, offers limited insight, as the resulting performance differences are expected to be negligible in that direction. Therefore, to better assess the noise sensitivity of the filtering methods under realistic conditions, we perform our analysis by varying the azimuth noise variance σ_θ^2, while fixing the range standard deviation at $\sigma_r^2 = 50\ m$, which approximately corresponds to the average range noise level observed in the test set of Section 4.

The results of the noise sensitivity analysis are summarized in Fig. 9. This analysis confirms that the architecture introduced in Section 3 demonstrates increased robustness to elevated levels of sensor noise when compared to the IMM filter. Overall, the IMM filter displays a higher sensitivity to noise perturbations, whereas, except for Mnemonic KF, the hybrid MB/data-driven approaches, as well as TrMTT, demonstrate greater robustness to noise variations when compared to the IMM, with AttEnsKNets consistently exhibiting superior overall tracking accuracy among all the considered filtering methods.

4.4. Robustness to missing measurements

In real-world applications, sensors are susceptible to missing measurements for several reasons that might depend on both the sensor itself and the target. In this situation, the Bayesian-based tracking system cannot rely on measurements to update the prior distribution using Bayes' theorem in eq. (2), resulting in estimates with lower accuracy and higher uncertainty when measurements are missing.

To formally describe missing measurements, we modeled sensor measurements at time t as a Bernoulli random variable with probability P_d of successfully providing the measurement.

We then evaluated the tracking systems on the trajectories defined in the test set described in Table 1, with the sensor noise parameters fixed to $\sigma_r = 50$ m and $\sigma_\theta = 0.5°$ for range and azimuth, respectively. The detection probability P_d was varied to assess how the filtering methods degrade under conditions of increasing measurement loss.

It is important to highlight that, throughout the analyses presented in this section, none of the filtering methods requiring a training phase were retrained or fine-tuned to address the missing-measurement scenario.

To ensure fair comparisons, TrMTT was excluded from this analysis, as its original implementation does not support missing values. In contrast, DeepMTT was included because it leverages the UKF's ability to propagate state estimates without measurements, achieved by omitting the update stage in the Kalman filtering process. Similarly, Mnemonic KF was included, as it is also embedded within the KF framework and can handle missing measurements in the same manner.

This distinction underscores a key limitation of purely end-to-end learning-based approaches: they must be explicitly designed to manage missing data. In contrast, hybrid approaches that integrate NNs within the Kalman filtering framework inherit its ability to implicitly handle such scenarios, offering greater robustness in practical applications.

The average RMSE under the specified sensor noise conditions is plotted as a function of the detection probability P_d in Fig. 10, where the case $P_d = 1$ corresponds to the scenario analyzed in Section 4, by fixing $\sigma_r = 50.\ m$ and $\sigma_\theta = 0.5\ °$. The results of this analysis reinforce the superior performance of AttEnsKNets against the considered filtering methods.

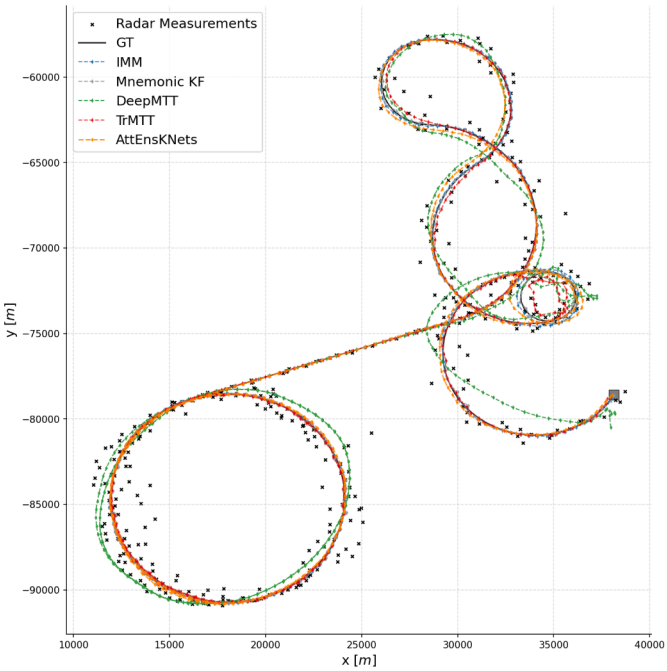

Fig. 11. Target tracking result from the HSHM test set in Section 4.5 on the xy-plane.

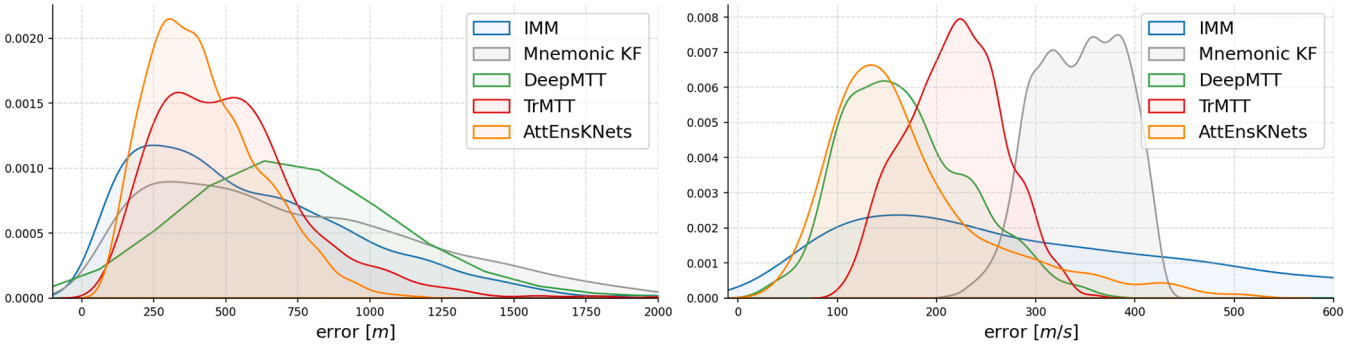

Fig. 12. RMSE distribution in position and velocity per filtering method in HSHM test set in Section 4.5.

Table 4
Simulation parameters ranges for HSHM Test Set generation.

Parameter	HSHM Test Set
N	1000
T	400 s
Δt	1 s
number of switches	[4; 8]
σ_q	[5.; 20.] m/s^2
σ_r	10, 20, …, 100 m
σ_θ	0.1, 0.2, …, 1 °
Initial distance	[50; 150] km
Initial azimuth	[−180; +180] °
Initial velocity	[400; 600] m/s
ω	[−20; +20] $°/s$

4.5. Out of distribution targets

We tested the ability of the proposed architectures to track targets outside the training distribution. To this end, we generated a novel test set with different generationen parameters, summarized in Table 4. In particular, we focused on High-Speed, Highly Maneuvering (HSHM) targets, performing 4 to 8 motion switches in longer simulations, each lasting 400 s instead of 200. This setup allows us to evaluate the effectiveness and the robustness of the filtering methods, especially on out-of-distribution samples.

The results of the evaluation of the tracking methods for out-of-distribution targets are summarized in Table 5, which represents first and second momentum estimators of the error distributions displayed in Fig. 12.

In this particular setting, where targets are from unseen statistics with respect to those observed in the training and validation scenarios, the AttEnsKNets architecture can outperform the other filtering methods in all of the considered metrics, except for the estimation of the targets' velocity, where DeepMTT confirms its effectiveness by producing a slightly lower average RMSE, at the expense of significantly poorer performance in locating the targets. Overall, these results suggest a satisfactory capability of the proposed filtering method to adapt to previously unseen, challenging scenarios, preserving its tracking performance.

Fig. 11 provides the results of tracking from the HSHM test set in the xy-plane, whose RMSE error in position and velocity at each timestep is plotted in Fig. 13.

Based on the 100 MC simulations of a single trajectory from the HSHM test set, the proposed AttEnsKNets consistently demonstrates superior performance over the entire simulation period in

Table 5
Summary table of metrics when tracking trajectories in the test set of Section 4.5 with empirical mean μ and standard deviation σ per filtering method. For filtering methods that do not account for target motion mode estimation, accuracy (ACC) is not available (n.a.).

	RMSE all ↓		RMSE pos [m] ↓		RMSE vel [m/s] ↓		ACC % ↑	
	μ	σ	μ	σ	μ	σ	μ	σ
IMM	471.18	322.29	576.85	391.69	332.32	234.81	46.71	18.95
Mnemonic KF [16]	595.47	304.13	735.37	482.87	344.08	43.30	n.a.	
DeepMTT [10]	573.12	376.52	789.42	433.60	**170.25**	65.88	n.a.	
TrMTT [13]	404.25	184.67	521.23	271.03	218.13	48.38	n.a.	
AttEnsKNets [ours]	**328.98**	143.28	**426.41**	193.15	175.54	87.05	**75.06**	15.14

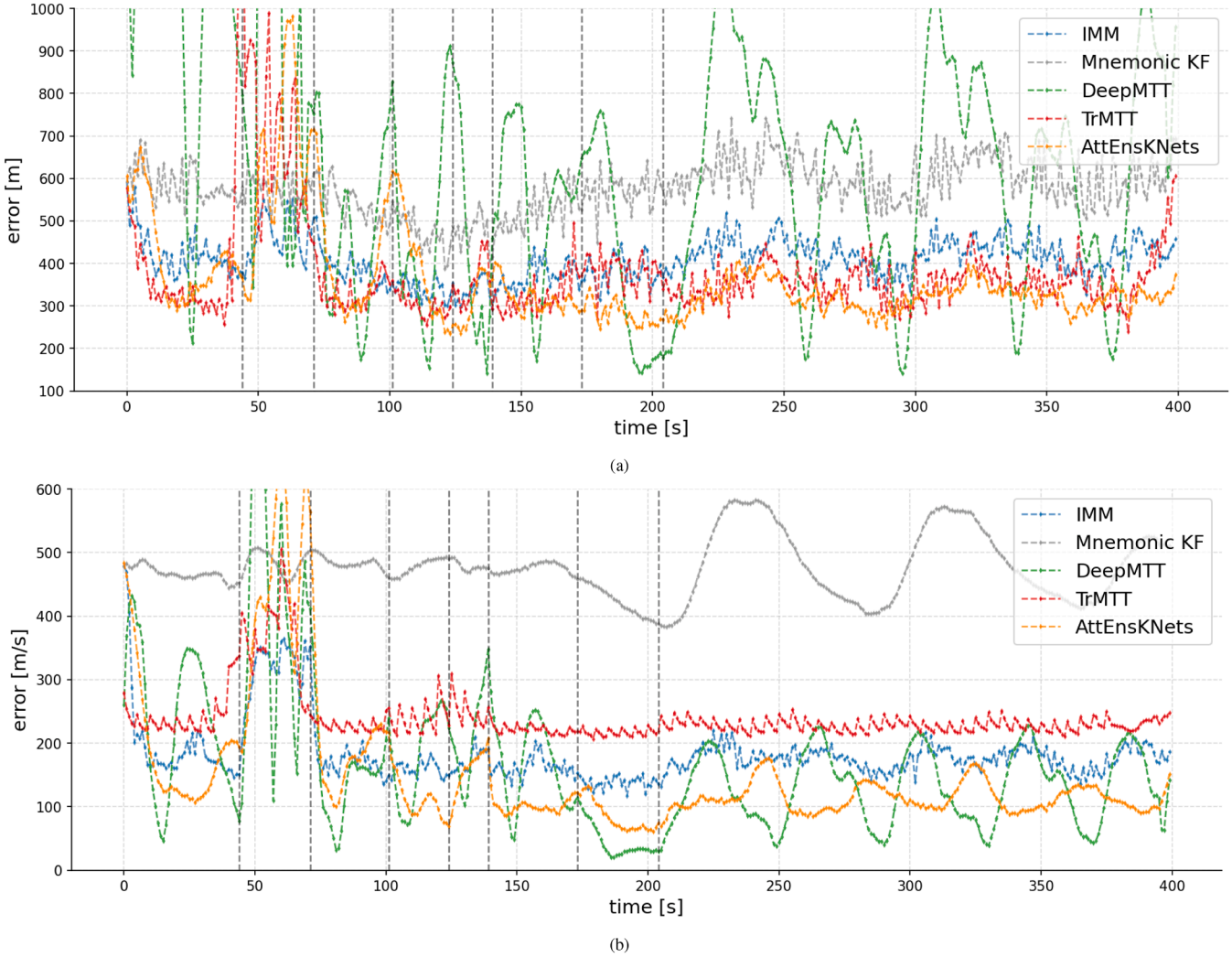

Fig. 13. Tracking results over 100 Monte-Carlo runs for the trajectory example from the HSHM test set in Section 4.5 and displayed in Fig. 11, in particular: (a) RMSE in position estimation of each filtering method per timestep; (b) RMSE in velocity estimation of each filtering method per timestep. The vertical dashed line indicates either a change in the target's motion mode or, in cases where the mode remains the same, a change in the turn rate.

both position and velocity estimation. While certain methods achieve better performance over short segments of the trajectory, AttEnsKNets attains the lowest overall error across the full duration, with an average RMSE of 237.43, compared to 273.24 for the IMM, 431.22 for the Mnemonic KF, 441.90 for DeepMTT, and 264.65 for TrMTT.

Notably, all filtering methods incorporating data-driven components exhibit performance degradation between 44 s and 71 s. This segment corresponds to a right-hand turn at a rate of $-19.0\ °/s$, a turn rate that was not present in any of the training set trajectories, thereby posing a challenging out-of-distribution scenario.

Moreover, AttEnsKNets has preserved its superior accuracy in predicting targets' motion mode when compared to the IMM, which is struggling in the particular trajectory shown in Fig. 11.

Indeed, in Fig. 14 one can see an overall high accuracy, reaching 87.00% when classifying the motion modes of the target in the sampled trajectory in Fig. 11, and a quick adaptation to target mode changes, when compared to the IMM filter, which only correctly estimated the target's motion mode for 11.25% of the simulation duration.

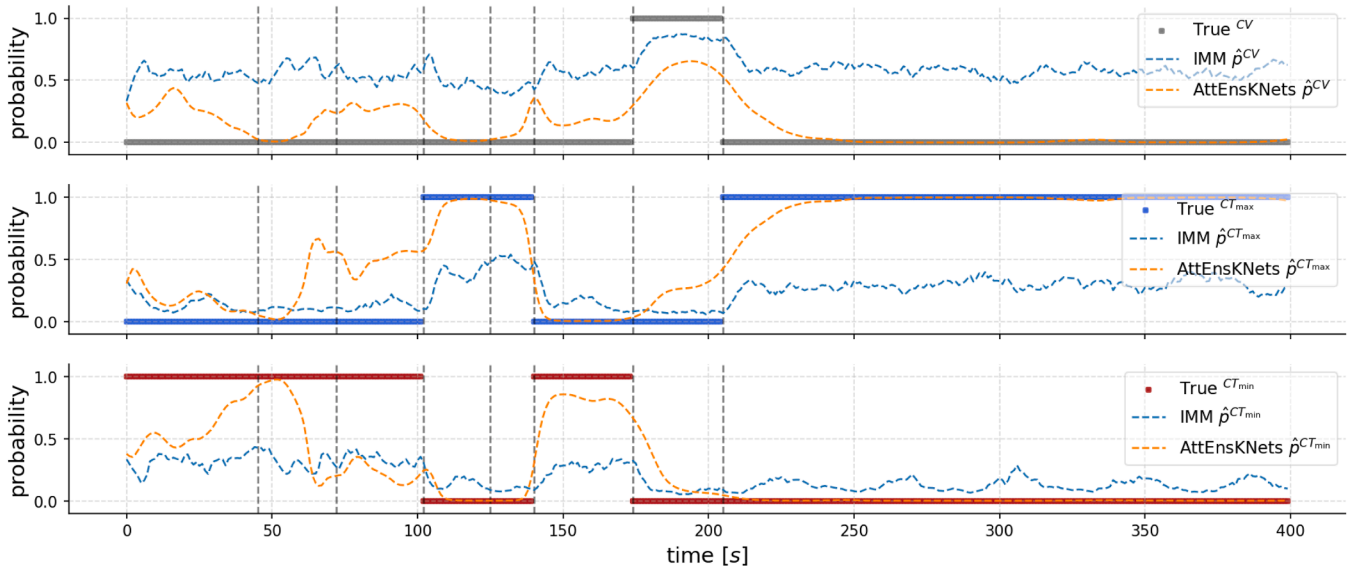

Fig. 14. Target's motion mode probabilities estimation at each timestep of the HSHM trajectory example displayed in Fig. 11 for IMM and AttEnsKNets.

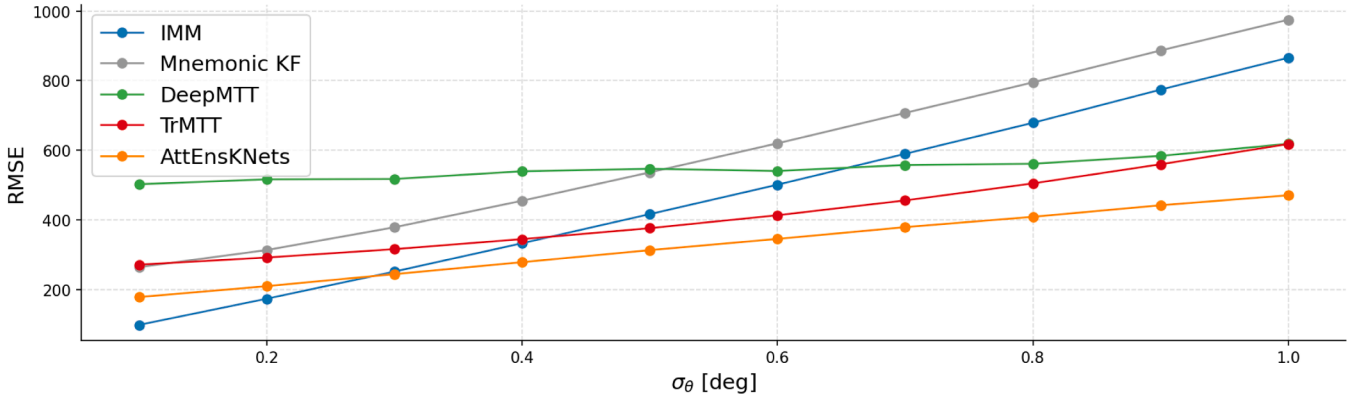

Fig. 15. Average RMSE against increasing azimuth noise levels for each filtering method tracking trajectories in the HSHM test set of Section 4.5.

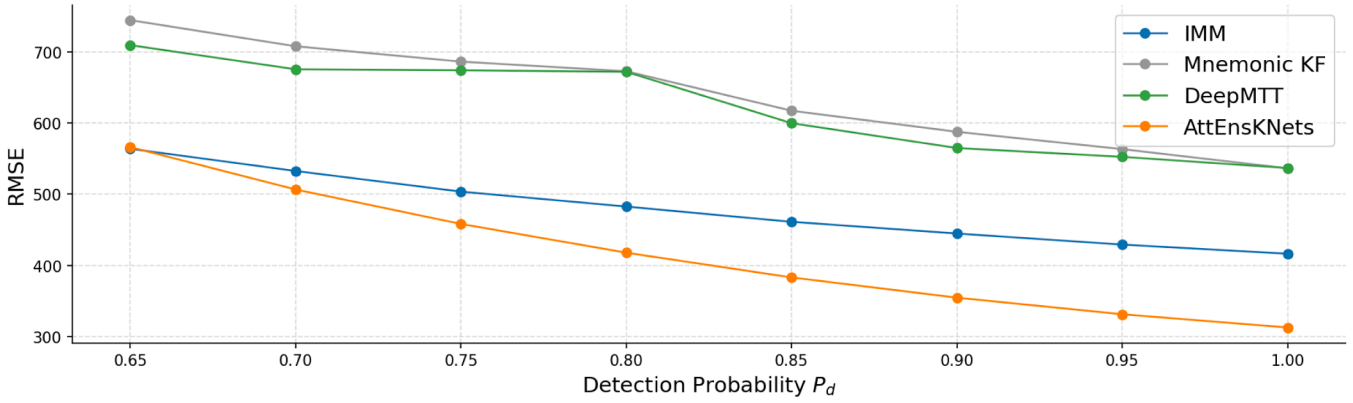

Fig. 16. Average RMSE against decreasing target detection probability for the HSHM test set in Section 4.5 for each filtering method.

In addition, we conducted a noise sensitivity analysis on the more challenging settings of the HSHM test set trajectories. In this analysis, the azimuth standard deviation σ_θ was varied while the range standard deviation was fixed at $\sigma_r = 50$ m. The results are summarized in Fig. 15. At low noise levels, specifically up to $\sigma_\theta = 0.3$ °, the IMM filter achieves the best performance among the considered methods, or at least comparable results, in terms of average RMSE, which might be explained by the fact that these data are unseen for all the considered methods exploiting NNs.

Interestingly, on the HSHM test set, DeepMTT exhibits a remarkably low sensitivity to noise variations, as evidenced by the nearly flat trend observed in Fig. 15. This suggests that its underlying architecture is relatively robust to changes in measurement quality, despite other known limitations. Overall, the results support the conclusion that AttEnsKNets

is the most robust filtering method among the evaluated approaches, particularly in high-noise scenarios, where it consistently outperforms the alternatives.

Finally, we evaluated the performance with respect to the targets' detection probability on the HSHM test set, fixing $\sigma_r = 50\ m$ and $\sigma_\theta = 0.5\ °$. The corresponding results are presented in Fig. 16. These findings further support the claim that AttEnsKNets is the most robust filtering method among the considered ones, consistently achieving the lowest average RMSE when compared to DeepMTT and Mnemonic KF, even as the probability of detection decreases. Comparable performance to the IMM is observed only in the most challenging conditions, namely at $P_d = 0.65$.

5. Conclusions

In this work, we improved the architecture proposed in [24], designed to address the challenges posed by MTT and overcome the limitations of MB target tracking systems, and in particular those of the IMM filter. The ability of RNNs, and in particular LSTMs, to learn and memorize meaningful temporal relationships has been exploited to formulate a hybrid MB and data-driven architecture resembling the IMM filter framework to mix posterior predictions of a pool of KalmanNets in an ensemble fashion, leading to the design of the Ensemble of KalmanNets. We then extended its ability in mixing MB filtering by designing an innovation-based attention method to let the ensemble focus on the most likely motion model when classifying target dynamics, which also further increased the level of explainability of the hybrid filtering method.

We tested the proposed approach and the modified computational framework that integrates the attention mechanism in a broad range of MTT scenarios, gathering different target maneuvers and several sensor noise levels. We tested the performance of the proposed AttEnsKNets against the IMM filter and other hybrid MB/data-driven filtering methods, proving its effectiveness by superior tracking accuracy in all of the considered scenarios.

When testing the sensitivity to sensor noise levels, by varying noise in the azimuth direction, and the performance degradation when the sensor might be affected by missing measurements, by varying the target's detection probability, the AttEnsKNets has provided greater robustness with respect to all of the considered filtering methods.

We then evaluated the filtering methods on out-of-distribution samples, namely, trajectories performed by HSHM targets, which were not included in the generation parameters of the training set for the proposed AttEnsKNets. The proposed approaches achieved better performance than all of the considered filtering approaches, even in this challenging setting, with the exception of target velocity estimation, where DeepMTT exhibited notable generalization capability and attained an RMSE competitive with that of the proposed method. However, this slight improvement in velocity estimation was obtained at the expense of a significantly reduced accuracy in position estimation. Through sensitivity analyses in this challenging, previously unseen scenario, AttEnsKNets has further demonstrated its robustness, confirming its position as the best-performing filtering method among the considered alternatives.

It is crucial to highlight that the proposed architecture does not account for the posterior state estimation covariance in its computational framework, though it can both provide a posterior covariance, which is not, however, processing evidence that might come from data. In future work, we plan to exploit estimation uncertainty to further improve the Ensemble's performance, its robustness, and explainability, assessing its ability to produce predictions that are as confident as they are accurate.

CRediT authorship contribution statement

Marco Mari: Writing - review & editing, Writing - original draft, Validation, Software, Methodology, Investigation, Conceptualization; **Lauro Snidaro:** Writing - review & editing, Writing - original draft, Supervision, Conceptualization.

Data availability

Data will be made available on request.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The activity has been funded by a PhD scholarship granted by Leonardo Electronics Division.

References

[1] S. Julier, J. Uhlmann, H.F. Durrant-Whyte, A new method for the nonlinear transformation of means and covariances in filters and estimators, IEEE Trans. Autom. Contr. 45 (3) (2000) 477–482. https://doi.org/10.1109/9.847726
[2] B. Ristic, S. Arulampalam, N. Gordon, Beyond the Kalman filter: Particle filters for tracking applications, Artech house, 2003.
[3] X.R. Li, V.P. Jilkov, Survey of maneuvering target tracking. Part V. multiple-model methods, IEEE Trans. Aerosp. Electron. Syst. 41 (4) (2005) 1255–1321.
[4] H.A.P. Blom, Y. Bar-Shalom, The interacting multiple model algorithm for systems with Markovian switching coefficients, IEEE Trans. Autom. Contr. 33 (8) (1988) 780–783.
[5] T. Li, J.M. Corchado, H. Chen, J. Bajo, Track a smoothly maneuvering target based on trajectory estimation, in: 2017 20th International Conference on Information Fusion (Fusion), 2017, pp. 1–8. https://doi.org/10.23919/ICIF.2017.8009731
[6] T. Li, H. Chen, S. Sun, J.M. Corchado, Joint Smoothing and Tracking Based on Continuous-Time Target Trajectory Function Fitting, IEEE Trans. Autom. Sci. Eng. 16 (3) (2019) 1476–1483. https://doi.org/10.1109/TASE.2018.2882641
[7] J. Chung, C. Gulcehre, K. Cho, Y. Bengio, Empirical evaluation of gated recurrent neural networks on sequence modeling, in: NIPS 2014 Workshop on Deep Learning, December 2014, 2014.
[8] X.-B. Jin, R.J. Robert Jeremiah, T.-L. Su, Y.-T. Bai, J.-L. Kong, The New Trend of State Estimation: From Model-Driven to Hybrid-Driven Methods, Sensors 21 (6) (2021). https://doi.org/10.3390/s21062085
[9] Y. Bai, B. Yan, C. Zhou, T. Su, X. Jin, State of art on state estimation: Kalman filter driven by machine learning, Annual Rev. Contr. 56 (2023) 100909. https://doi.org/10.1016/j.arcontrol.2023.100909
[10] J. Liu, Z. Wang, M. Xu, DeepMTT: A deep learning maneuvering target-tracking algorithm based on bidirectional LSTM network, Inf. Fusion 53 (2020) 289–304.
[11] J. Liu, S. Yang, F. Yang, A cross-and-dot-product neural network based filtering for maneuvering-target tracking, Neural Comput. Appl. 34 (17) (2022) 14929–14944.
[12] G. Zhao, Z. Wang, Y. Huang, H. Zhang, X. Ma, Transformer-based maneuvering target tracking, Sensors 22 (21) (2022) 8482.
[13] Y. Zhang, G. Li, X.-P. Zhang, Y. He, A deep learning model based on transformer structure for radar tracking of maneuvering targets, Inf. Fusion 103 (2024) 102120.
[14] L. Deng, D. Li, R. Li, Improved IMM algorithm based on RNNs, in: Journal of Physics: Conference Series, 1518, IOP Publishing, 2020, p. 012055.
[15] S. Jung, I. Schlangen, A. Charlish, Time-dependent state prediction for the Kalman filter based on recurrent neural networks, in: 2020 IEEE 23rd International Conference on Information Fusion (FUSION), IEEE, 2020, pp. 1–7.
[16] S. Jung, I. Schlangen, A. Charlish, A mnemonic Kalman filter for non-linear systems with extensive temporal dependencies, IEEE Signal Processing Letters 27 (2020) 1005–1009.
[17] I. Schlangen, A. Brandenburger, M. Sun, J.R. Hopgood, Data-Driven Approaches for Modelling Target Behaviour, arXiv preprint arXiv:2410.10538 (2024).
[18] J. Liu, J. Yan, D. Wan, X. Li, S. Al-Rubaye, A. Al-Dulaimi, Z. Quan, Digital Twins based Intelligent State Prediction Method for Maneuvering-Target Tracking, IEEE J. Selected Areas Commun. 41 (11) (2023) 3589–3606.
[19] Z.-g. Liu, Z.-k. Wang, Y.-b. Yang, Y. Lu, A Data-Driven Maneuvering Target Tracking Method Aided with Partial Models, IEEE Transactions on Vehicular Technology 73 (1) (2023) 412–425.
[20] S. Yan, Y. Liang, L. Zheng, M. Fan, X. Wang, B. Wang, Explainable Gated Bayesian Recurrent Neural Network for Non-Markov State Estimation, IEEE Trans. Signal Process. 72 (2024) 4302–4317. https://doi.org/10.1109/TSP.2024.3390139
[21] S. Yan, Y. Liang, L. Zheng, M. Fan, B. Wang, Memory-biomimetic deep Bayesian filtering, Inf. Fusion 112 (2024) 102580. https://doi.org/10.1016/j.inffus.2024.102580

[22] K. Shen, W. Yuan, J. Yan, K. Ma, MCST: An Adaptive Tracking Algorithm for High-Speed and Highly Maneuverable Targets Based on Bidirectional LSTM Network, IEEE Transactions on Aerospace and Electronic Systems 61 (2) (2025) 3205–3226. https://doi.org/10.1109/TAES.2024.3484393
[23] A. Lin, W.-A. Zhang, L. Guo, Attentional filtering: Dynamic system state estimation as an attention, Inf. Fusion 123 (2025) 103224. https://doi.org/10.1016/j.inffus.2025.103224
[24] M. Mari, L. Snidaro, Ensemble of KalmanNets for Maneuvering Target Tracking, in: 2024 27th International Conference on Information Fusion (FUSION), IEEE, 2024, pp. 1–7.
[25] G. Revach, N. Shlezinger, X. Ni, A.L. Escoriza, R.J.G. Van Sloun, Y.C. Eldar, KalmanNet: Neural network aided Kalman filtering for partially known dynamics, IEEE Trans. Signal Process. 70 (2022) 1532–1547.
[26] D. Ulyanov, A. Vedaldi, V.S. Lempitsky, Instance Normalization: The Missing Ingredient for Fast Stylization, CoRR abs/1607.08022 (2016). arxiv:1607.08022
[27] Y. Zhang, Q. Yang, A Survey on Multi-Task Learning, IEEE Trans. Knowl. Data Eng. 34 (12) (2022) 5586–5609. https://doi.org/10.1109/TKDE.2021.3070203
[28] S. Chen, Y. Zhang, Q. Yang, Multi-Task Learning in Natural Language Processing: An Overview, ACM Comput. Surv. 56 (12) (2024). https://doi.org/10.1145/3663363
[29] X.R. Li, Y. Bar-Shalom, Design of an interacting multiple model algorithm for air traffic control tracking, IEEE Trans. Contr. Syst. Technol. 1 (3) (1993) 186–194. https://doi.org/10.1109/87.251886
[30] X. Rong Li, V.P. Jilkov, Survey of maneuvering target tracking. Part I. Dynamic models, IEEE Transactions on Aerospace and Electronic Systems 39 (4) (2003) 1333–1364. https://doi.org/10.1109/TAES.2003.1261132
[31] I. Loshchilov, F. Hutter, Decoupled Weight Decay Regularization, in: International Conference on Learning Representations, 2019. https://openreview.net/forum?id=Bkg6RiCqY7.
[32] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting, J. Mach. Learn. Res. 15 (1) (2014) 1929–1958.
[33] L. Xu, X.R. Li, Z. Duan, Hybrid grid multiple-model estimation with application to maneuvering target tracking, IEEE Trans. Aerosp. Electron. Syst. 52 (1) (2016) 122–136. https://doi.org/10.1109/TAES.2015.140423
[34] D. Lerro, Y. Bar-Shalom, Tracking with debiased consistent converted measurements versus EKF, IEEE Trans. Aerosp. Electron. Syst. 29 (3) (1993) 1015–1022. https://doi.org/10.1109/7.220948