REVIEW ARTICLE

A Survey on Deep Learning-Based Target State Estimation and Filtering

Deqiang Han[1,*], **Wei Xiong**[2], **Jianhong Guo**[1] **and Ziling Wang**[2]

[1] School of Automation, Xi'an Jiaotong University, Xi'an 710049, China
[2] Institute of Information Fusion, Naval Aeronautical University, Yantai 264001, China

Abstract

Target state estimation is traditionally carried out within the state-space framework using the Kalman filter and its variants, and it underpins applications in control, navigation, target tracking and information fusion. The quality of the estimate depends heavily on how accurately the state model, the measurement model and the process and measurement noise statistics are specified. A growing body of work uses deep learning to learn some or all of these components from data, and such data-driven filters have shown clear advantages when prior information is imprecise or the model is mismatched. This survey reviews recent deep learning-based state estimation and filtering methods developed under the state-space assumption. We group them according to how strongly they rely on the physical model, i.e., model-constrained, model-assisted and model-free deep filtering, and discuss the design choices within each group. Representative methods (KalmanNet, KalmanFormer and APBM) are compared with the EKF and UKF in simulations covering matched models, measurement-model mismatch and maneuvering trajectories. The results indicate that the learned filters perform close to the model-based filters when the model is correct and are considerably more robust under mismatch. Open problems and possible research directions are summarized at the end.

Keywords: deep learning, filtering estimation, noise model, state model, information fusion.

1 Introduction

Filtering estimation aims to obtain optimal estimates of the system states of interest in dynamic systems subject to noise and uncertainty, and it is one of the most active research and application areas in multi-source information fusion [1–3]. The Kalman filter [4], proposed by R. E. Kalman in 1960, recursively combines the state model, the measurement model and the measurements to produce state estimates under the linear minimum mean-square error (LMMSE) criterion. It has had a lasting influence on control theory, signal processing and information fusion, and is used in navigation and positioning [5, 6], automatic control (e.g., flight control and autonomous

Academic Editor:
Weiping Ding

Submitted: 11 July 2026
Revised: 06 September 2026
Accepted: 19 September 2026
Published: 23 November 2026

Vol. 3, **No.** 4, 2026.
10.62762/CJIF.2026.227213

***Corresponding author:**
✉ Deqiang Han
deqhan@xjtu.edu.cn

Citation
Han, D., Xiong, W., Guo, J., & Wang, Z. (2026). A Survey on Deep Learning-Based Target State Estimation and Filtering. *Chinese Journal of Information Fusion*, 3(4), 262–278.

© 2026 by the Authors. Published by Institute of Central Computation and Knowledge. This is an open access article under the CC BY license (https://creativecommons.org/licenses/by/4.0/).

vehicles) [7, 8], target tracking [9], radar signal processing [10, 11], biomedical engineering [12, 13] and financial time series forecasting [14, 15].

The Kalman filter, however, assumes a linear system with Gaussian noise and requires the state model, the measurement model and the noise statistics to be known. In practice, systems are often nonlinear; noise models cannot be determined precisely; parameters of the state model may be unknown or time-varying; high-dimensional measurement equations may be too complex to write analytically; and measurements may be missing or corrupted by outliers. These conditions make the standard Kalman filter difficult to apply without modification.

To handle nonlinearity and uncertain motion patterns, many model-based improvements have been proposed. Representative examples are the extended Kalman filter (EKF) [16], the unscented Kalman filter (UKF) [17] and the particle filter (PF) [18], which approximate the statistics of nonlinearly transformed variables by local linearization, deterministic sampling and Monte Carlo sampling, respectively. For highly maneuvering targets, the interacting multiple model (IMM) approach [19] runs a bank of model-matched filters in parallel. These methods still have well-known limitations: the EKF introduces linearization errors that can accumulate over time; the UKF degrades in strongly nonlinear, high-dimensional systems; the PF suffers from particle degeneracy, and the number of particles it needs grows exponentially with the state dimension; and all of them rely on predefined motion models and noise statistics that are often unavailable in practice. These problems become most apparent in complex environments where accurate analytical models are hard to build.

Deep learning, initially applied mainly to image processing, detection and recognition, has in recent years also been used extensively for state estimation, filtering and information fusion. Advances in sequence modeling and representation learning [20] have moved this field toward data-driven filters that combine learned representations with physical models. In this survey we classify existing methods according to the degree to which they rely on the physical model, since this largely determines a method's interpretability, data efficiency and adaptability. Three classes are distinguished:

- *Model-constrained deep filtering*: methods that keep the Kalman filtering equations, assume that the state transition model $\mathbf{F}$ and measurement model $\mathbf{H}$ are known, and use neural networks only to learn specific components such as the Kalman gain [21–24], the noise covariance [25], model residual terms [26] or physics-informed training objectives [27]. Examples include KalmanNet [21], KalmanFormer [23], MM-KalmanNet [22], Recursive KalmanNet [24], A-KIT [25], APBM [26] and DAF [27].
- *Model-assisted deep filtering*: methods that use the physical model as a soft constraint or auxiliary information, so that $\mathbf{F}$ or $\mathbf{H}$ may be approximate, unknown or learned jointly. This class includes latent-space filtering for high-dimensional measurements [28], meta-learning for fast adaptation across scenarios [29], and learned motion models used as priors in Bayesian filtering [30].
- *Model-free deep filtering*: methods that estimate the state directly from measurement sequences without an explicit process model. This class includes variational approaches such as the deep Kalman filter [31], fully data-driven recurrent estimators such as DANSE [32], Transformer-based sequence filters [33] and, more recently, score-based diffusion estimators [34–36].

The classification also exposes a trade-off. Methods with stronger model constraints usually offer better physical interpretability and higher sample efficiency but adapt less well to severe model mismatch, whereas model-free methods are more flexible and can capture complex unknown dynamics at the cost of more training data and weaker physical consistency. The classification is independent of the network architecture: the same GRU, Transformer or score-based backbone can appear in any of the three classes, so it complements architecture-oriented taxonomies rather than duplicating them. It can therefore help practitioners choose a method according to model availability, data volume and noise characteristics.

The main contributions of this survey are as follows.

- We propose a taxonomy based on the degree of physical model involvement, which is complementary to the network architecture and allows deep filters to be compared across the whole range from strictly model-constrained to fully data-driven designs.
- We review representative work published mainly between 2020 and 2025, including

Transformer-based filters and score-based diffusion state estimators, which, to the best of our knowledge, have received little attention in earlier surveys.

- We compare representative methods in simulations with measurement-model mismatch and maneuvering trajectories and report their accuracy and computational cost.
- We discuss open challenges and future directions, including multi-source information fusion, uncertainty calibration and real-time deployment on resource-limited platforms.

The rest of this survey is organized as follows. Section 2 reviews the standard Kalman filter and its difficulties in complex scenarios. Section 3 presents deep learning-based Kalman filtering methods under the proposed taxonomy. Section 4 compares several of these methods by simulation. Section 5 discusses open challenges and future trends, and Section 6 concludes the survey.

2 Fundamentals of Kalman Filtering and Its Challenges

2.1 Standard Linear Kalman Filter

The standard linear Kalman filter is a recursive estimator built on the state-space model (SSM) and the LMMSE criterion. The state and measurement equations are

$$\begin{cases} \mathbf{x}_k = \mathbf{F}_k \mathbf{x}_{k-1} + \mathbf{\Gamma}_k \mathbf{w}_k \\ \mathbf{z}_k = \mathbf{H}_k \mathbf{x}_k + \mathbf{v}_k \end{cases} \tag{1}$$

where $\mathbf{x}_k$ is the state vector at time step k, $\mathbf{F}_k$ is the state transition matrix, $\mathbf{\Gamma}_k$ is the process noise input matrix, $\mathbf{w}_k$ is the zero-mean white process noise, $\mathbf{z}_k$ is the measurement vector, $\mathbf{H}_k$ is the measurement matrix, and $\mathbf{v}_k$ is the zero-mean white measurement noise.

Optimality requires a linear system, Gaussian noise and known model and noise statistics, namely the process noise covariance matrix $\mathbf{Q}_k$ and the measurement noise covariance matrix $\mathbf{R}_k$. The recursion is given in Eqs. (2)-(6).

1) Initialization:

$$\begin{cases} \hat{\mathbf{x}}_{0|0} = \hat{\mathbf{x}}_0 \\ \mathbf{P}_{0|0} = \mathbf{P}_0 \end{cases} \tag{2}$$

2) One-step prediction of the state and its covariance:

$$\begin{cases} \hat{\mathbf{x}}_{k|k-1} = \mathbf{F}_k \hat{\mathbf{x}}_{k-1|k-1} \\ \mathbf{P}_{k|k-1} = \mathbf{F}_k \mathbf{P}_{k-1|k-1} \mathbf{F}_k^T + \mathbf{\Gamma}_k \mathbf{Q}_k \mathbf{\Gamma}_k^T \end{cases} \tag{3}$$

3) Gain computation:

$$\mathbf{K}_k = \mathbf{P}_{k|k-1} \mathbf{H}_k^T \mathbf{S}_k^{-1}, \quad \mathbf{S}_k = \mathbf{H}_k \mathbf{P}_{k|k-1} \mathbf{H}_k^T + \mathbf{R}_k \tag{4}$$

where $\mathbf{S}_k$ is the innovation covariance.

4) Innovation:

$$\tilde{\mathbf{z}}_k = \mathbf{z}_k - \mathbf{H}_k \hat{\mathbf{x}}_{k|k-1} \tag{5}$$

5) State and covariance update:

$$\begin{cases} \hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k \tilde{\mathbf{z}}_k \\ \mathbf{P}_{k|k} = (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k) \mathbf{P}_{k|k-1} \end{cases} \tag{6}$$

where $\mathbf{I}$ is the identity matrix of appropriate dimension.

In practice these assumptions are rarely satisfied in full. Uncertainty in the system model ($\mathbf{F}_k$ and $\mathbf{H}_k$ not known a priori), nonlinearity, non-Gaussian noise and unknown noise statistics ($\mathbf{Q}_k$ and $\mathbf{R}_k$) all limit the applicability of the standard linear Kalman filter.

2.2 Improvements to the Standard Kalman Filter

2.2.1 Extensions for Nonlinear Problems

For nonlinear problems, the representative improvements are the EKF, UKF and PF.

The EKF [16] approximates the nonlinear models by a first-order Taylor expansion, evaluates the Jacobian at the current estimate, and applies the Kalman filter to the linearized model. It is computationally cheap and suits mildly nonlinear problems. Its drawbacks are that linearization may introduce large errors, the models must be differentiable, and accuracy and stability cannot be guaranteed in strongly nonlinear or non-Gaussian settings.

The UKF [17] represents the mean and covariance by a set of deterministic sigma points, propagates them through the nonlinear functions via the unscented transform, and recomputes the mean and covariance from the transformed points. It requires no Jacobian and is generally more accurate than the EKF, but still describes the distribution only by its mean and covariance. Its computational cost is slightly higher than that of the EKF, and it suits moderately nonlinear problems.

The PF [18] represents the posterior distribution with a set of weighted random samples (particles) that are propagated and reweighted over time. It requires neither the Gaussian assumption nor linearization and, with enough particles, can approximate the true posterior, which makes it suitable for strongly nonlinear problems. Because it solves point estimation as a density estimation problem, however, it is computationally expensive. The number of particles needed grows rapidly with the state dimension (the curse of dimensionality), and effective resampling strategies are required to avoid degeneracy.

2.2.2 *Improvements for Model Uncertainty*

In real scenarios the state model is often time-varying: targets may switch between uniform straight-line motion, accelerated turns and evasive maneuvers; environmental changes may switch system parameters; and the model structure or parameters may be uncertain. Multiple-model filtering addresses this by defining a set of candidate models, each with its own state equation and noise characteristics, running a filter (e.g., KF, EKF or UKF) for each model, computing the model probabilities from the measurements, and combining the model-conditioned estimates into an overall estimate. Switching or mixing between motion modes in this way improves accuracy and robustness.

IMM [19] is the most widely used multiple-model framework. It handles several dynamic patterns simultaneously and achieves smooth model transitions without hard switching, and it performs well when the transition probabilities are known or can be estimated. Its cost grows with the number of models, it requires a properly specified model set and transition probability matrix, and it cannot handle motion patterns outside the model set. Other multiple-model approaches include variable-structure multiple models [37] and mode-set adaptive IMM [38].

3 Deep Learning-Based Kalman Filtering

Although the UKF, PF and IMM extend the Kalman filter to nonlinear estimation and uncertain motion patterns, their limitations become apparent for complex modern dynamic systems. First, they depend on accurate physical models and prior noise distributions, which are hard to obtain in non-stationary environments or high-dimensional, strongly nonlinear systems. Second, linearization errors (EKF) or the rapid growth of computation with dimension (PF) reduce their robustness in complex non-stationary tasks. The following subsections review deep learning-based filters according to the three classes defined in Section 1.

3.1 Model-Constrained Deep Filtering

In traditional nonlinear filters such as the EKF, the gain $\mathbf{K}_k$ depends on an accurate system model, including the state transition model $\mathbf{F}_k$, the measurement model $\mathbf{H}_k$, and the noise covariances $\mathbf{Q}_k$ and $\mathbf{R}_k$. Under model mismatch or non-Gaussian noise, the analytically computed gain can deviate considerably from the optimal one.

Model-constrained deep filtering keeps the recursive structure of the Kalman filter and assigns the computation of specific components to neural networks. The structure inherited from the physical model acts as an inductive bias, so these networks usually converge faster and generalize better than purely end-to-end models when training data are limited.

3.1.1 *Deep Gain Learning Architectures*

KalmanNet KalmanNet [21] is the best-known method in this group and retains the prediction-update structure of the Kalman filter. Consider the nonlinear discrete-time system

$$\begin{cases} \mathbf{x}_k = f(\mathbf{x}_{k-1}) + \mathbf{w}_k \\ \mathbf{z}_k = h(\mathbf{x}_k) + \mathbf{v}_k \end{cases} \tag{7}$$

where $\mathbf{x}_k$ is the state and $\mathbf{z}_k$ is the measurement.

As shown in Figure 1, KalmanNet operates in two phases.

1) Prediction: the known (or approximate) physical model gives the prior estimate

$$\hat{\mathbf{x}}_{k|k-1} = f(\hat{\mathbf{x}}_{k-1|k-1}). \tag{8}$$

2) Update: the state update keeps the linear form

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathcal{K}_{\vartheta}(\cdot)\left(\mathbf{z}_k - h(\hat{\mathbf{x}}_{k|k-1})\right) \tag{9}$$

where $\mathcal{K}_{\vartheta}(\cdot)$ is a neural network with parameters ϑ that replaces the analytical computation and outputs the learned gain $\mathbf{K}_k$ directly from time-series features such as the measurement residuals.

To track the temporal evolution of the noise statistics, KalmanNet uses gated recurrent units (GRUs) [39] as its backbone. The update and reset gates allow the network to retain or discard historical information selectively, which suits time-varying noise. A typical structure contains:

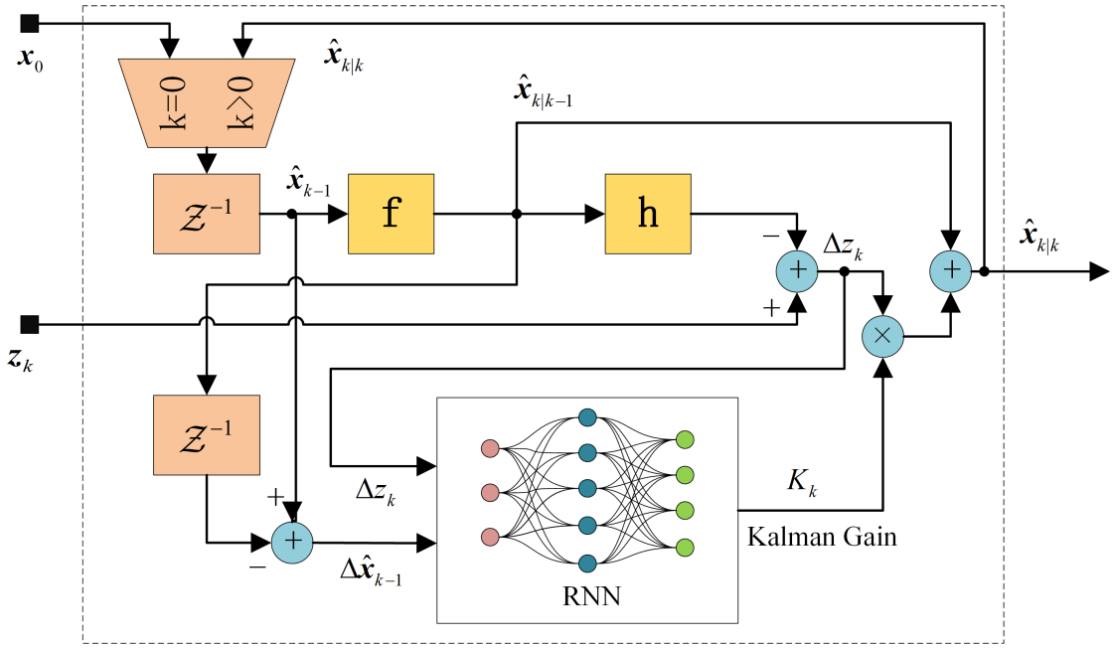


Figure 1. KalmanNet architecture. $\mathbf{K}_k$ denotes the Kalman gain learned by the GRU-based network; $\Delta\mathbf{z}_k$ denotes the innovation difference; $\Delta\hat{\mathbf{x}}_{k-1}$ denotes the forward update difference.

1) an input layer that normalizes the residual and difference features and lifts them to a higher dimension through a fully connected layer;
2) hidden GRU layers that capture temporal dependencies and implicitly track the process noise covariance $\mathbf{Q}_k$, the prior state covariance $\mathbf{P}_{k|k-1}$ and the innovation covariance $\mathbf{S}_k$ without specifying them explicitly;
3) an output layer of fully connected layers that produces a gain matrix whose dimensions match the state and measurement spaces.

The network inputs are computed from the raw measurements and past estimates:

1) innovation difference $\Delta\mathbf{z}_k = \mathbf{z}_k - h(\hat{\mathbf{x}}_{k|k-1})$, the deviation between the current measurement and its model-based prediction;
2) observation difference $\Delta\tilde{\mathbf{z}}_k = \mathbf{z}_k - \mathbf{z}_{k-1}$, which reflects the rate of change of the measurements;
3) forward update difference $\Delta\hat{\mathbf{x}}_{k-1} = \hat{\mathbf{x}}_{k-1|k-1} - \hat{\mathbf{x}}_{k-1|k-2}$, the size of the previous correction;
4) forward evolution difference $\Delta\tilde{\mathbf{x}}_{k-1} = \hat{\mathbf{x}}_{k-1|k-1} - \hat{\mathbf{x}}_{k-2|k-2}$, which reflects the smoothness of the estimates.

Because the gain is learned from the history of residuals, the network can compensate implicitly for errors in the assumed $\mathbf{Q}_k$ and $\mathbf{R}_k$ and for systematic model biases.

KalmanNet is trained end-to-end in a supervised manner. Paired state-measurement sequences are typically generated by simulating the system model under diverse conditions, or obtained from field data whose reference trajectories are produced by post-processing. The loss is the mean square error (MSE) between the estimated and true states:

$$\mathcal{L}(\boldsymbol{\vartheta}) = \mathbb{E}\left[\left\|\hat{\mathbf{x}}_{k|k}(\boldsymbol{\vartheta}) - \mathbf{x}_k^{\mathrm{gt}}\right\|_2^2\right] \tag{10}$$

where $\mathbf{x}_k^{\mathrm{gt}}$ is the ground-truth state at time step k and $\|\cdot\|_2$ is the Euclidean norm.

KalmanFormer Although KalmanNet learns the gain successfully with GRUs, recurrent architectures suffer from vanishing gradients and make limited use of long-range history in long sequences. KalmanFormer [23] replaces the recurrent gain model with a Transformer encoder-decoder [40] that processes the measurement-side and state-side features separately and uses self-attention to capture dependencies across the whole input window.

KalmanFormer keeps the linear state update of the Kalman filter. The gain learning task is split into measurement feature extraction and estimation feedback, which handle measurement-sequence features and state-sequence features, respectively, as illustrated in Figure 2.

The encoder models the measurement side. Its inputs are the innovation-difference and

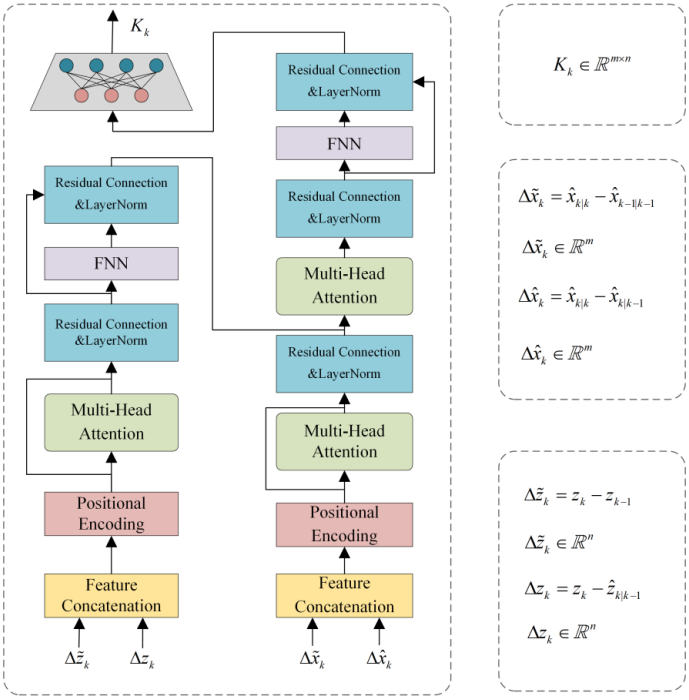


Figure 2. KalmanFormer architecture. The encoder extracts measurement-sequence features via multi-head self-attention; the decoder fuses them with posterior state-estimation feedback via cross-attention to output the learned Kalman gain.

observation-difference sequences, and multi-head self-attention identifies abrupt changes or slow drift in the measurements and compresses them into feature vectors. The decoder models the state feedback. Its inputs are the update-difference and evolution-difference sequences of past posterior estimates, and cross-attention fuses these with the encoder output to produce the gain at the current time step. The gain computation can be written as

$$\begin{aligned}\mathbf{e}_k^{\text{meas}} &= \text{Encoder}\big(\Delta\mathbf{z}_{k-W:k},\ \Delta\tilde{\mathbf{z}}_{k-W:k}\big)\\ \mathbf{e}_k^{\text{state}} &= \text{Decoder}\big(\mathbf{e}_k^{\text{meas}},\ \Delta\hat{\mathbf{x}}_{k-W:k-1},\ \Delta\tilde{\mathbf{x}}_{k-W:k-1}\big)\\ \mathbf{K}_k &= \text{FC}\big(\mathbf{e}_k^{\text{state}}\big)\end{aligned} \tag{11}$$

where W is the length of the sliding window and FC denotes a fully connected layer. The update equation keeps its physical form:

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k\Delta\mathbf{z}_k. \tag{12}$$

During training, KalmanFormer does not feed the true states back at each step. Instead, the residual and difference inputs are computed from the model's own posterior estimates at the previous step, exactly as in online inference. This lets the model learn how estimation errors evolve over time and avoids divergence caused by the accumulation of small model errors.

Extensions of KalmanNet Because learning the gain $\mathbf{K}_k$ has proved effective against model mismatch, several works extend KalmanNet while keeping its physical structure, targeting fast maneuvers, high-dimensional sensing, rapid transfer and uncertainty quantification.

1) MM-KalmanNet. MM-KalmanNet [22] introduces a mode probability network and maintains several KalmanNet sub-filters in parallel, each tuned to one motion pattern. The final estimate is a weighted combination

$$\hat{\mathbf{x}}_{k|k} = \sum_{i=1}^{M} \mu_k^{(i)}\,\hat{\mathbf{x}}_{k|k}^{(i)} \tag{13}$$

where the mode weights $\mu_k^{(i)}$ are produced online by an attention-based classification network, M is the number of predefined motion patterns (e.g., constant velocity, constant acceleration and coordinated turn), and $\hat{\mathbf{x}}_{k|k}^{(i)}$ is the posterior estimate of the sub-filter matched to the i-th pattern. Training combines the state MSE with a cross-entropy loss for mode classification, which improves response speed and tracking accuracy during maneuvers.

2) Recursive KalmanNet. Recursive KalmanNet [24] addresses KalmanNet's inability to report estimation uncertainty (second-order error moments). Besides the gain network, a second recurrent network outputs the Cholesky factor of a correction term, and the posterior covariance is propagated recursively in a generalized Joseph form, which keeps it symmetric and positive semi-definite. The MSE loss is replaced by the Gaussian negative log-likelihood (NLL), up to an additive constant:

$$\mathcal{L}(\vartheta) = \frac{1}{2}\mathbb{E}\Big[\mathbf{e}_k^T\hat{\mathbf{P}}_{k|k}^{-1}\mathbf{e}_k + \log\det\hat{\mathbf{P}}_{k|k}\Big],\quad \mathbf{e}_k = \hat{\mathbf{x}}_{k|k} - \mathbf{x}_k^{\text{gt}} \tag{14}$$

where $\hat{\mathbf{P}}_{k|k}$ is the estimated posterior covariance and $\log\det(\cdot)$ is the log-determinant. This makes the reported confidence consistent with the actual estimation error, which matters for downstream planning or fault diagnosis. The structure is shown in Figure 3.

3.1.2 *Residual Compensation Architectures*

Augmented Physics-Based Model (APBM) For complex systems, physical models often contain simplifications that limit estimation accuracy. Instead of learning the Kalman gain, the augmented

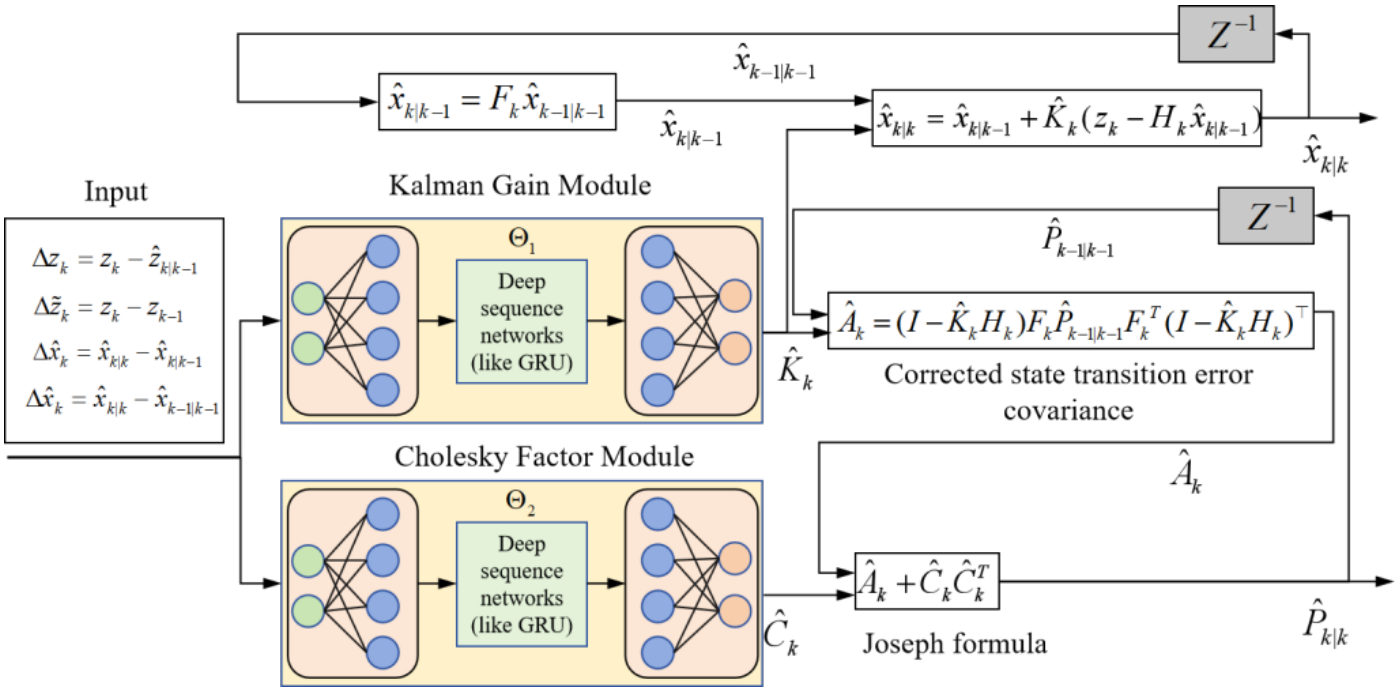


Figure 3. Recursive KalmanNet architecture. A second recurrent network, running in parallel with the gain-learning network, provides the covariance correction used to propagate the posterior covariance for uncertainty quantification.

physics-based model (APBM) introduced by Imbiriba et al. [41] treats the physical model as the backbone and uses a neural network as a residual corrector, forming a hybrid dynamic model; Straka et al. [26] later proposed an interpretable variant that explicitly bounds the deviation of the hybrid model from the physics-based one. The augmented state transition can be written as

$$\mathbf{x}_k = f_{\text{phy}}(\mathbf{x}_{k-1}) + f_{\text{nn}}(\mathbf{x}_{k-1}; \boldsymbol{\vartheta}) + \mathbf{w}_k \tag{15}$$

where $f_{\text{phy}}(\cdot)$ is the known physical model and $f_{\text{nn}}(\cdot; \boldsymbol{\vartheta})$ is a neural network with parameters $\boldsymbol{\vartheta}$ that learns the residual caused by model mismatch.

To prevent the network from overfitting the measurement noise and violating the physics, the network parameters are appended to the state vector and regularized toward a baseline value $\bar{\boldsymbol{\vartheta}}$, for which the network makes no contribution. This is implemented by the augmented measurement model [41]

$$\begin{bmatrix} \mathbf{z}_k \\ \bar{\boldsymbol{\vartheta}} \end{bmatrix} = \begin{bmatrix} \mathbf{H}_k \mathbf{x}_k \\ \boldsymbol{\vartheta}_k \end{bmatrix} + \begin{bmatrix} \mathbf{v}_k \\ \mathbf{r}_k^{\vartheta} \end{bmatrix}, \quad \mathbf{r}_k^{\vartheta} \sim \mathcal{N}\left(\mathbf{0}, \tfrac{1}{\lambda}\mathbf{I}\right) \tag{16}$$

where $\lambda > 0$ controls the trade-off between physical consistency and data adaptability: as $\lambda \to \infty$ the parameters are pinned to $\bar{\boldsymbol{\vartheta}}$ and the model reduces to the physics-based one, whereas as $\lambda \to 0$ the constraint vanishes and the model becomes essentially data-driven.

For the resulting high-dimensional nonlinear integrals, APBM uses the cubature Kalman filter (CKF) [42], which is usually more accurate than the EKF and cheaper than the PF. Based on the third-degree spherical-radial cubature rule, the CKF propagates $2n$ equally weighted cubature points (where n is the dimension of the, here augmented, state) through the nonlinear model and recomputes the mean and covariance, which gives numerically stable estimates in high-dimensional nonlinear systems.

Unlike KalmanNet, which needs ground-truth states for supervised training, APBM estimates the network parameters by recursive Bayesian estimation (RBE) together with the state. The underlying objective combines a data-fit term, a regularization term on the network parameters and the dynamic constraints, so the parameters can be updated online from the measurements and the physical prior alone, without labeled trajectories.

Physics-Informed Data-Driven Autoregressive Filter (DAF) Whereas APBM corrects the unmodeled dynamics explicitly, another line of work uses neural networks to extract sequence features and uses the physical equations only to guide self-supervised training. Liu et al. [27] proposed a physics-informed data-driven autoregressive nonlinear filter (DAF), which exploits the sequence modeling capability of Transformers to infer the states directly from past measurements while turning the

physical equations into training constraints.

The key feature of this method is its autoregressive marginal-likelihood training. DAF does not use true states for supervision; instead, it maximizes the product of the conditional likelihoods of the measurement sequence:

$$\mathcal{L}(\vartheta) = \prod_{k=1}^{T} p\left(\mathbf{z}_k \mid \mathbf{z}_{1:k-1}; \vartheta\right) \tag{17}$$

where T is the length of the measurement sequence and $\mathbf{z}_{1:k-1}$ denotes the measurements from time step 1 to $k-1$.

Adaptive Kalman-Informed Transformer (A-KIT) A-KIT [25] also combines attention with a model-based filter. Developed for EKF-based fusion of inertial sensors and a Doppler velocity log, it uses a set-transformer network to estimate a scaling of the process noise covariance online. Its training architecture is shown in Figure 4.

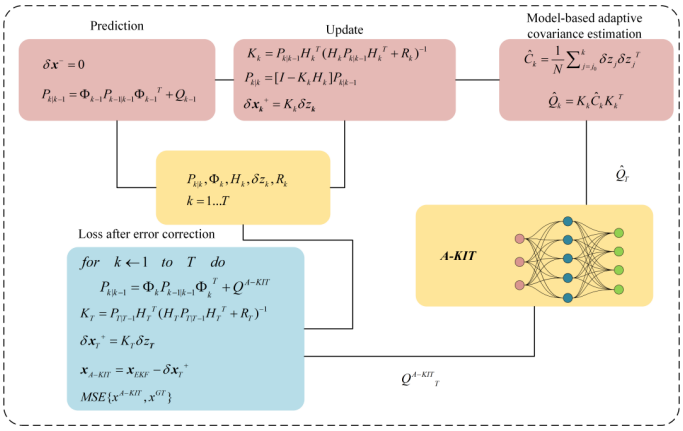


Figure 4. A-KIT training architecture. A set-transformer adaptively scales the process noise covariance online; a Kalman-informed loss embeds the full EKF prediction-update loop into the training objective.

The main novelty of A-KIT is its Kalman-informed loss. Rather than comparing network outputs directly with ground truth, the loss runs the complete EKF prediction-update loop during training: covariance propagation, gain computation and error-state correction are all simulated, and the MSE between the corrected and true states is back-propagated. The noise scaling produced by the network is thus trained to be consistent with the filter equations.

3.2 Model-Assisted Deep Filtering

When an accurate physical model is unavailable but partial structure or approximate dynamics are known, model-assisted methods use the physical knowledge as a soft rather than a hard constraint. They relax the assumption of perfectly known $\mathbf{F}_k$ or $\mathbf{H}_k$, which allows them to cope with model uncertainty or high-dimensional measurements.

3.2.1 *Latent-Space Filtering*

When the measurement model $\mathbf{H}_k$ is unknown or the measurements are high-dimensional (e.g., camera images or LiDAR point clouds), latent-space methods first map the raw measurements to a low-dimensional embedding and then perform Kalman filtering in that space. The state transition model $\mathbf{F}_k$ may be approximately known or learned jointly, so the requirement for an exact measurement model is largely removed.

Latent-KalmanNet [28] uses a deep encoder to project high-dimensional measurements onto low-dimensional latent vectors (Figure 5):

$$\mathbf{z}_k^{\text{latent}} = g_{\boldsymbol{\varphi}}(\mathbf{y}_k) \tag{18}$$

where $\mathbf{y}_k$ is the raw high-dimensional measurement at time step k, $g_{\boldsymbol{\varphi}}(\cdot)$ is the encoder with parameters $\boldsymbol{\varphi}$, and $\mathbf{z}_k^{\text{latent}}$ is the resulting latent measurement. KalmanNet then performs prediction and update in the latent space. The encoder and the filter are optimized jointly end-to-end, so low-level feature extraction is tuned for high-level state inference, and a reconstruction loss encourages the latent variables to retain the information needed to describe the dynamics.

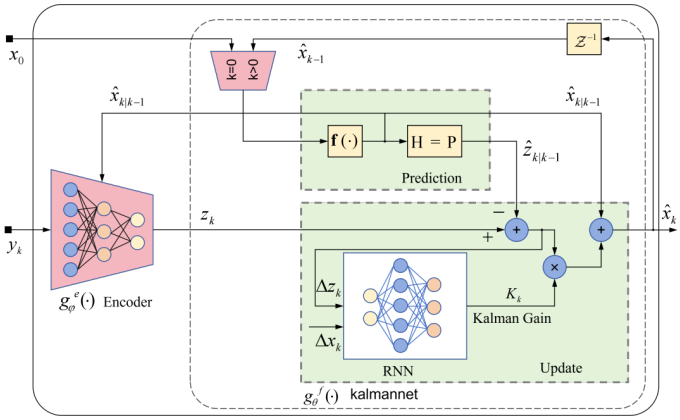


Figure 5. Latent-KalmanNet architecture. A deep feature encoder projects high-dimensional raw measurements into a low-dimensional latent space, where KalmanNet performs prediction and update.

Beyond Latent-KalmanNet, this direction also covers end-to-end visual-inertial odometry (VIO) systems that combine deep feature extraction with filtering, where the measurement model is learned implicitly by the encoder.

3.2.2 *Meta-Learning for Fast Adaptation*

As shown in Figure 6, MAML-KalmanNet [29] uses model-agnostic meta-learning (MAML) [43] to improve the generalization of deep filters deployed across scenarios.

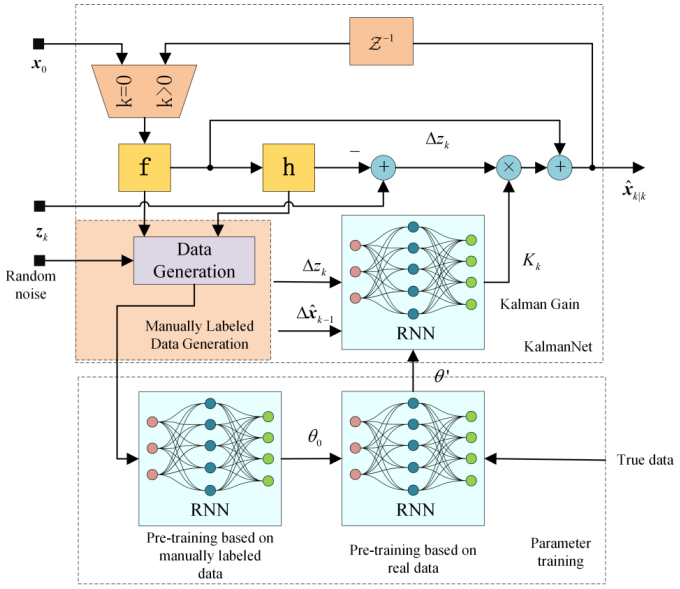


Figure 6. MAML-KalmanNet architecture. An outer meta-learning loop learns an initialization that is sensitive to task changes; an inner loop adapts it to a new filtering task with a few gradient steps.

Instead of optimizing performance on a single dataset, the model learns an initialization from which it can adapt to new sensor-noise or sampling-rate conditions (tasks) with few online samples and gradient steps:

$$\min_{\boldsymbol{\vartheta}} \mathbb{E}_{\mathcal{T}\sim p(\mathcal{T})}\Big[\mathcal{L}_{\mathcal{T}}\big(\boldsymbol{\vartheta} - \beta\nabla_{\boldsymbol{\vartheta}}\mathcal{L}_{\mathcal{T}}(\boldsymbol{\vartheta})\big)\Big] \tag{19}$$

where $\mathcal{T}$ is a filtering task drawn from the task distribution $p(\mathcal{T})$, $\mathcal{L}_{\mathcal{T}}(\cdot)$ is the task-specific loss, and β is the inner-loop step size. Compared with the offline training of KalmanNet, this gives the filter the ability to adapt quickly in unknown and non-stationary environments. When model parameters differ between environments but follow a common distribution, meta-learning relaxes the need for a precise and fixed $\mathbf{F}_k$.

Pulido et al. [30] propose a hybrid scheme in which a Transformer learns the target motion model and replaces the prediction step of a particle filter. The learned motion model acts as a soft physical prior, and the filter uncertainty drives an information-based guidance policy, illustrating how Transformer-based motion learning can be combined with Bayesian filtering.

3.3 Model-Free Deep Filtering

When the process model is unknown or cannot be formulated analytically, model-free methods learn the state dynamics directly from measurement sequences. They do not rely explicitly on the Kalman filter equations and are suited to scenarios with little physical modeling knowledge.

With the progress of end-to-end learning and generative modeling, data have also been used to address cases where training samples lack reliable ground truth. A representative method is DANSE (data-driven nonlinear state estimation) proposed by Ghosh et al. [32]. It uses recurrent neural networks to learn unknown dynamics directly from observed sequences and does not need a state transition equation, giving an unsupervised filter for model-free processes observed through a known linear measurement model.

DANSE combines the ability of deep networks to model nonlinear stochastic processes with the closed-form update available for linear measurements. Although the evolution of the system is unknown, a Gaussian prior for the current state is obtained by causal inference on past measurements with a GRU network:

$$p(\mathbf{x}_k \mid \mathbf{z}_{1:k-1}) = \mathcal{N}\big(\mathbf{m}_{k|k-1}(\boldsymbol{\vartheta}),\, \mathrm{diag}(\mathbf{l}_{k|k-1}(\boldsymbol{\vartheta}))\big) \tag{20}$$

where $\mathbf{m}_{k|k-1}(\boldsymbol{\vartheta})$ and $\mathbf{l}_{k|k-1}(\boldsymbol{\vartheta})$ are the prior mean vector and the vector of prior variances produced by the GRU with parameters $\boldsymbol{\vartheta}$, and $\mathrm{diag}(\cdot)$ forms a diagonal covariance matrix from a vector. Both quantities are nonlinear functions of the network parameters, yet, unlike fully black-box mappings, DANSE keeps the measurement update analytically tractable. The architecture is shown in Figure 7.

DANSE is trained without state labels by maximizing the marginal likelihood of the observed data, i.e., by minimizing the negative log-likelihood

$$\mathcal{L}(\boldsymbol{\vartheta}) = -\sum_{k=1}^{T} \log p\big(\mathbf{z}_k \mid \mathbf{z}_{1:k-1}; \boldsymbol{\vartheta}\big). \tag{21}$$

Beyond DANSE, fully end-to-end filters map measurement sequences to state sequences with sequence models such as Transformers or temporal convolutional networks (TCNs), without reference to the Kalman filter equations.

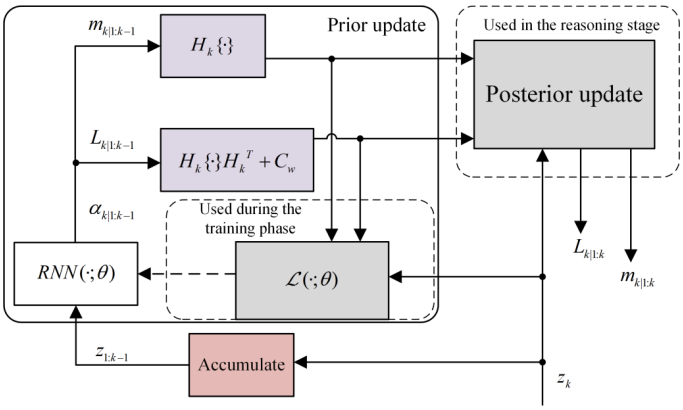


Figure 7. DANSE architecture. A GRU generates the prior mean and diagonal prior covariance from past measurements; the measurement update remains analytically tractable.

On the theoretical side, Goel and Bartlett [33] prove that a causally masked Transformer can approximate the Kalman filter with an error bound that holds uniformly in time, where the self-attention block reduces to Nadaraya-Watson kernel smoothing. This provides a formal basis for sequence-model-based filtering without explicit physical priors.

Score-based diffusion state estimation is an emerging direction in this class. The works below come from data assimilation, which estimates the state of a dynamical system from sequential observations and is therefore closely related to filtering in the state-space framework. Rozet and Louppe [34] propose score-based data assimilation, which decomposes the score of a long trajectory into a sum of segment-level scores for training; at inference, the whole trajectory is generated non-autoregressively, and the observation model does not enter training. For high-dimensional systems with sparse observations, Si and Chen [35] introduce Latent-EnSF, which maps the state to a low-dimensional latent space with a variational autoencoder (VAE) before ensemble score filtering, greatly reducing the effect of dimensionality. Sun et al. [36] present Align-DA, which incorporates preference alignment into score-based atmospheric data assimilation; by aligning the generated distribution with several preferences such as observation confidence and dynamical constraints, it achieves a better balance between physical consistency and fidelity to the observations.

4 Simulation Comparison of Deep Filters

This section evaluates model-constrained deep filters under model mismatch using a constant acceleration (CA) motion model and multiple-model maneuvering trajectories. The EKF, UKF, KalmanNet (KNet), KalmanFormer (KFormer) and APBM are compared in terms of estimation accuracy with a correct model, robustness to measurement-model mismatch, and tracking of maneuvering trajectories.

4.1 Simulation Setup

4.1.1 Models

The simulation uses a three-dimensional CA model. The state vector is $\mathbf{x}_k = [x_k, \dot{x}_k, \ddot{x}_k, y_k, \dot{y}_k, \ddot{y}_k, z_k, \dot{z}_k, \ddot{z}_k]^T$, where x_k, y_k, z_k are the position coordinates, $\dot{x}_k, \dot{y}_k, \dot{z}_k$ the velocity components and $\ddot{x}_k, \ddot{y}_k, \ddot{z}_k$ the acceleration components. The state transition matrix is

$$\mathbf{F} = \mathbf{I}_3 \otimes \begin{bmatrix} 1 & \Delta t & \frac{1}{2}\Delta t^2 \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{bmatrix} \tag{22}$$

where Δt is the sampling interval, $\otimes$ is the Kronecker product and $\mathbf{I}_3$ is the 3×3 identity matrix.

Under nominal conditions the measurement equation is linear:

$$\mathbf{z}_k = \mathbf{H}\mathbf{x}_k + \mathbf{v}_k, \quad \mathbf{v}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{R}) \tag{23}$$

where the measurement matrix extracts the positions:

$$\mathbf{H} = \mathbf{I}_3 \otimes \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}. \tag{24}$$

For measurement-model mismatch, a rotation $\Delta\psi$ is assumed between the measurement matrix used by the filter, $\mathbf{H}_{\text{filter}}$, and that of the true process, $\mathbf{H}_{\text{true}}$, representing sensor installation or calibration errors:

$$\mathbf{H}_{\text{true}} = \mathbf{T}_z(\Delta\psi)\,\mathbf{H}_{\text{filter}} \tag{25}$$

where $\mathbf{T}_z(\Delta\psi)$ is the rotation matrix about the z-axis.

4.1.2 Scenarios and Parameters

Five measurement noise levels are used to cover different signal-to-noise ratio (SNR) conditions. The noise standard deviations are 0.31 m, 0.71 m, 1.00 m, 1.41 m and 3.16 m, corresponding to SNRs of 10 dB, 3 dB, 0 dB, -3 dB and -10 dB, respectively, where the SNR is defined relative to a reference standard deviation $\sigma_{\text{ref}} = 1$ m:

$$\text{SNR (dB)} = 20\log_{10}\left(\frac{\sigma_{\text{ref}}}{\sigma}\right). \tag{26}$$

The dataset consists of independent sequences: 10 000 for training, 1 000 for validation and 1 000 for testing,

Table 1. Position RMSE (m) of the filters with a matched measurement model.

Noise std (m) (SNR)	0.31 (10 dB)	0.71 (3 dB)	1.00 (0 dB)	1.41 (−3 dB)	3.16 (−10 dB)
EKF	0.17	0.38	0.54	0.76	1.70
UKF	0.19	0.46	0.65	0.88	1.97
KNet	0.17	0.38	0.54	0.76	1.71
KFormer	0.20	0.41	0.54	0.76	1.70
APBM	0.18	0.41	0.65	0.78	1.94

each with $T = 1000$ time steps. The initial state of each sequence is drawn from $\mathbf{x}_0 \sim \mathcal{N}(\boldsymbol{\mu}_0, \mathbf{P}_0)$, where $\boldsymbol{\mu}_0$ and $\mathbf{P}_0$ are the mean and covariance of the initial state.

All neural filters are trained with the Adam optimizer [44] with weight decay. The shared hyperparameters are a learning rate $\eta = 10^{-3}$, a weight decay coefficient $\xi = 10^{-5}$ and a batch size of 64. The number of training iterations is set according to the convergence behavior of each network: 4000 for KalmanNet, 1000 for KalmanFormer, which converges faster, and 500 for APBM, whose structure is much smaller.

The root mean square error (RMSE) of the position estimate is the main performance metric:

$$\mathrm{RMSE} = \sqrt{\frac{1}{NT}\sum_{i=1}^{N}\sum_{k=1}^{T}\left\|\hat{\mathbf{p}}_{k|k}^{(i)} - \mathbf{p}_k^{(i),\mathrm{gt}}\right\|_2^2} \tag{27}$$

where N is the number of test sequences, $\hat{\mathbf{p}}_{k|k}^{(i)}$ is the filtered position estimate of the i-th sequence at time step k, and $\mathbf{p}_k^{(i),\mathrm{gt}}$ is the corresponding true position.

4.2 Performance with a Matched Model

In this scenario the measurement matrix of the filter matches the data-generating process ($\mathbf{H}_{\mathrm{filter}} = \mathbf{H}_{\mathrm{true}}$), so the basic accuracy of each algorithm can be assessed under ideal model conditions.

As shown in Table 1 and Figure 8, the RMSE of KNet is practically identical to that of the EKF at all noise levels. KFormer matches the EKF for $\sigma \geq 1.00$ m but is slightly worse at the two lowest noise levels (0.20 m vs. 0.17 m and 0.41 m vs. 0.38 m). APBM is close to the EKF at low noise but approaches the UKF at $\sigma = 1.00$ m and $\sigma = 3.16$ m. The UKF has the largest RMSE at every noise level except $\sigma = 0.31$ m (tied with APBM at $\sigma = 1.00$ m). Since the UKF and the EKF should coincide with the Kalman filter for a linear Gaussian model, the gap between them deserves further examination.

These results show that, when the physical model is correct and the system is linear, KNet reaches essentially the same accuracy as the EKF, while KFormer and APBM stay within about 20% of it. Through training, the network has implicitly learned a gain close to the optimal one,

$$\mathbf{K}_k^{\mathrm{nn}}(\boldsymbol{\vartheta}) \approx \mathbf{K}_k^{\mathrm{opt}} = \mathbf{P}_{k|k-1}\mathbf{H}^T\left(\mathbf{H}\mathbf{P}_{k|k-1}\mathbf{H}^T + \mathbf{R}\right)^{-1} \tag{28}$$

without explicit matrix inversion.

4.3 Robustness to Measurement-Model Mismatch

This scenario simulates sensor installation bias or calibration failure: the measurement matrix $\mathbf{H}_{\mathrm{true}}$ that generates the data is rotated by $\Delta\psi = 1^\circ$ relative to the matrix $\mathbf{H}_{\mathrm{filter}}$ used by the filter.

Under mismatch the innovation becomes biased. Assuming that the prediction is approximately unbiased, $\mathbb{E}[\hat{\mathbf{x}}_{k|k-1}] \approx \mathbb{E}[\mathbf{x}_k]$,

$$\begin{aligned}\mathbb{E}[\tilde{\mathbf{z}}_k] &= \mathbb{E}\left[\mathbf{z}_k - \mathbf{H}_{\mathrm{filter}}\hat{\mathbf{x}}_{k|k-1}\right] \\ &\approx \left(\mathbf{H}_{\mathrm{true}} - \mathbf{H}_{\mathrm{filter}}\right)\mathbb{E}[\mathbf{x}_k] \neq \mathbf{0}.\end{aligned} \tag{29}$$

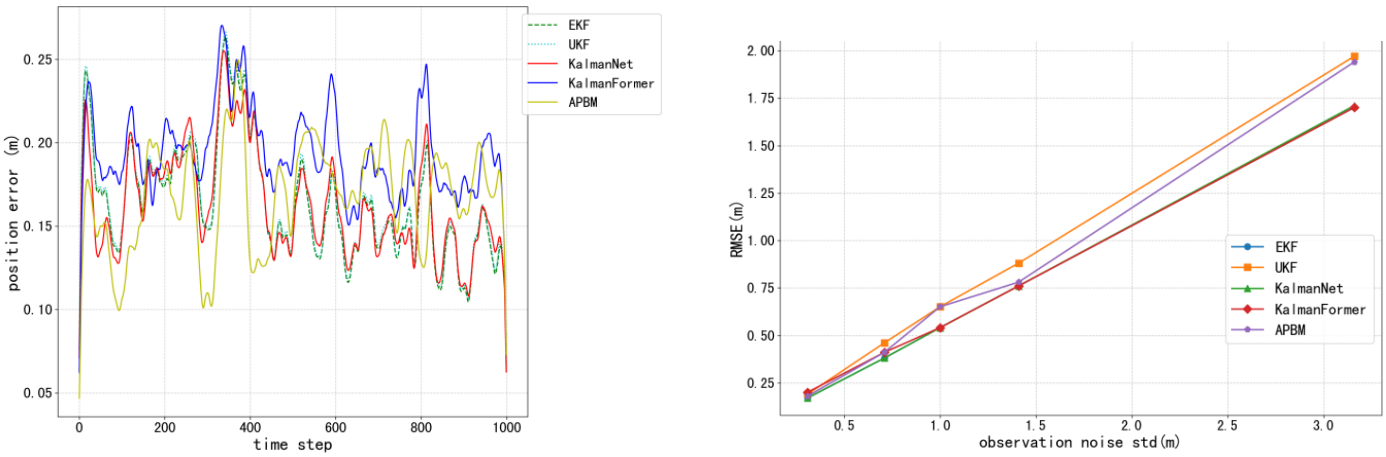
Traditional filters cannot compensate for this bias, because their gain is computed strictly from the assumed model,

$$\mathbf{K}_k^{\mathrm{EKF}} = \mathbf{P}_{k|k-1}\mathbf{H}_{\mathrm{filter}}^T\left(\mathbf{H}_{\mathrm{filter}}\mathbf{P}_{k|k-1}\mathbf{H}_{\mathrm{filter}}^T + \mathbf{R}\right)^{-1}, \tag{30}$$

which depends entirely on the incorrect $\mathbf{H}_{\mathrm{filter}}$.

As shown in Table 2 and Figure 9, the rotational mismatch markedly degrades the EKF and UKF, whose RMSE increases by a factor of roughly two to five relative to Table 1. The degradation of the learned filters is much smaller. In the time-domain curves, the EKF error rises substantially between roughly steps 200 and 550 (peaking at about 0.6 m), whereas the errors of the learned filters remain comparatively flat.

Among the learned filters, KFormer gives the lowest RMSE at four of the five noise levels, while KNet is best at $\sigma = 0.31$ m; APBM is worse than KNet at all levels except $\sigma = 0.71$ m, by 5% to 40%. Because the EKF and UKF compute the gain from the assumed model, even a small matrix error leads to an incorrect weighting



(a) (b)

Figure 8. Filter performance with a matched measurement model. (a) Position error over time at the 10 dB noise level ($\sigma = 0.31$ m); (b) RMSE versus measurement noise standard deviation.

Table 2. Position RMSE (m) of the filters under measurement-model mismatch ($\Delta\psi = 1°$).

Noise std (m) (SNR)	0.31 (10 dB)	0.71 (3 dB)	1.00 (0 dB)	1.41 (−3 dB)	3.16 (−10 dB)
EKF	0.37	1.30	2.21	3.41	5.89
UKF	0.44	1.53	2.54	4.22	7.91
KNet	0.21	0.78	1.40	1.47	3.39
KFormer	0.28	0.70	1.30	1.46	2.90
APBM	0.24	0.77	1.62	2.06	3.56

of the innovation. The learned filters instead infer the gain from data:

$$\mathbf{K}_k^{\mathrm{nn}} = \mathcal{K}_{\boldsymbol{\vartheta}}\left(\Delta\mathbf{z}_{k-W:k},\ \Delta\hat{\mathbf{x}}_{k-W:k-1}\right) \tag{31}$$

where $\mathcal{K}_{\boldsymbol{\vartheta}}(\cdot)$ maps the residual and difference sequences in a sliding window of length W to the gain; the network thereby compensates for the mismatch through the residual history instead of relying on the explicit $\mathbf{H}$ matrix.

4.4 Tracking of Multiple-Model Maneuvering Trajectories

To evaluate adaptability to complex dynamics, the filters use a single CA model while the true trajectories switch among constant velocity (CV), constant acceleration (CA) and coordinated turn (CT) models. This reproduces the model mismatch caused by target maneuvers.

Table 3. Position RMSE (m) on multiple-model maneuvering trajectories.

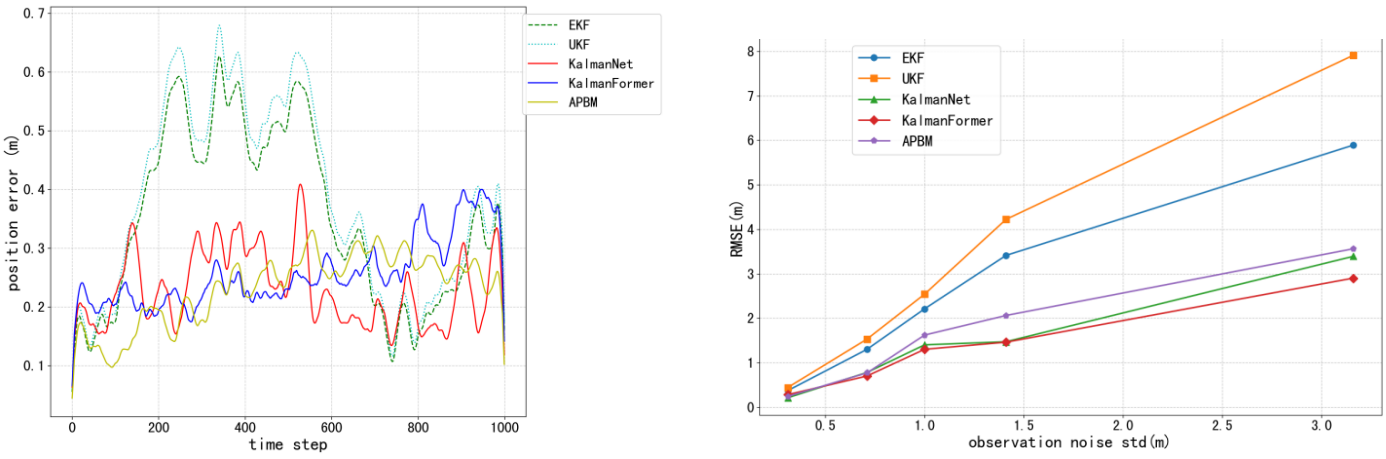
Filter	EKF	UKF	KNet	KFormer	APBM
RMSE (m)	11.26	11.26	6.42	5.85	6.63

As shown in Table 3 and Figure 10, the EKF and UKF lag behind the true trajectory during strong maneuvers because of their single-model assumption, and their windowed RMSE exceeds 10 m for much of the displayed interval. KNet, KFormer and APBM substantially reduce this maneuver-induced error and follow the true trajectory closely. Over time steps 300-600, the RMSE of KFormer mostly stays between 4 and 8 m, with a maximum of about 9 m near step 390, compared with about 12 m for KNet at the same instant.

The learned filters therefore handle highly dynamic motion better than single-model traditional filters. KalmanFormer gives the lowest overall RMSE and the smallest error peaks, which is consistent with the ability of attention to exploit a longer measurement history.

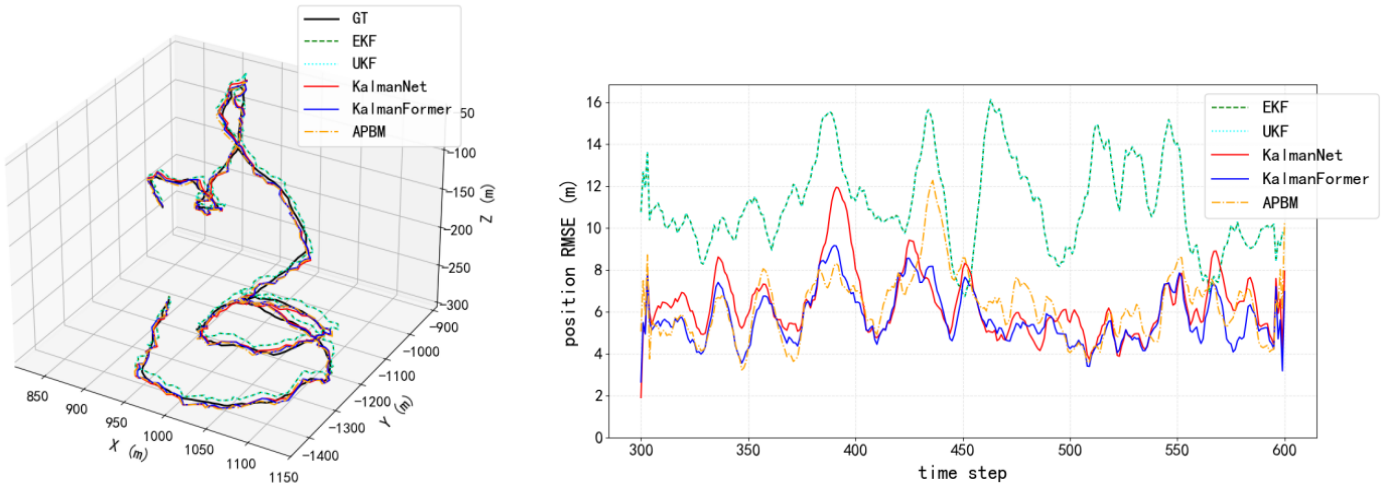
4.5 Summary of the Experiments

Table 4 lists the processing time for 200 test sequences together with the number of parameters and the floating-point operations (FLOPs) per time step. The run times of KNet and KFormer are of the same order as those of the conventional filters.



(a) (b)

Figure 9. Filter performance under measurement-model mismatch ($\Delta\psi = 1°$). (a) Position error over time at the 10 dB noise level ($\sigma = 0.31$ m); (b) RMSE versus measurement noise standard deviation.



(a) (b)

Figure 10. Filter performance on multiple-model maneuvering trajectories (time steps 300-600). (a) Three-dimensional trajectories of the filters and the ground truth; (b) position RMSE with respect to the ground truth (moving-average window of 10 steps).

Table 4. Processing time (200 test sequences), number of parameters and FLOPs per time step.

Filter	EKF	UKF	KNet	KFormer	APBM
Time (s)	0.94	1.41	1.10	2.06	4.63
Parameters	-	-	526 116	45 579	4 082
FLOPs/step	-	-	1.04 M	8.78 M	7.87 K

Across the three scenarios, the learned filters perform similarly to the traditional filters when the model is linear Gaussian and correct; under model mismatch, KNet, KFormer and APBM are clearly more accurate than the traditional filters, showing that the learning-based correction tolerates deviations of the physical model; and on maneuvering trajectories KFormer shows the most stable RMSE and the smallest error peaks. The complexity figures also show that parameter count and computational cost are not proportional. KalmanNet has the most parameters (526 116), mainly in its second fully connected layer ($90 \to 3600 \to 27$, about 425 000 parameters), but its per-step cost (1.04 MFLOPs) is about one-eighth of that of KalmanFormer (8.78 MFLOPs). KalmanNet is purely recurrent with a fixed per-step cost, whereas KalmanFormer, with only 45 579 parameters, computes self-attention over a window of $W = 64$ at every step, which costs $O(W^2)$ and dominates its FLOPs. APBM has the fewest parameters (4 082) and the lowest network FLOPs (7.87 K), yet the longest run time (4.63 s). The reason is that its network parameters are part of the CKF state: the augmented

state has about 4×10^3 dimensions, so each step propagates roughly $2n \approx 8 \times 10^3$ cubature points and updates a covariance of the same dimension, a cost not reflected in the per-step network FLOPs. KFormer's higher FLOPs, on the other hand, are partly offset by optimized Transformer kernels on the GPU.

5 Challenges and Future Directions

5.1 Current Challenges

Despite rapid progress, several major problems remain.

1) *Limited physical consistency and interpretability.* Many methods use deep networks to learn the evolution and measurement mechanisms, the noise or the gain directly. They may learn shortcuts that appear effective but contradict the underlying mechanisms and are hard to interpret. When operating conditions change (e.g., sensor replacement, a different sampling rate or new external disturbances), performance is not guaranteed and failures are difficult to diagnose.

2) *Limited generalization and out-of-distribution (OOD) robustness.* Training data usually cover a limited range of states, noise conditions and motion patterns. Under unforeseen conditions, such as unseen motion patterns, occlusions, sudden anomalies and heavy-tailed or correlated noise, the update rules learned by end-to-end networks may be less reliable than those of the traditional Kalman filter.

3) *Poor uncertainty calibration.* The noise covariances $\mathbf{Q}$ and $\mathbf{R}$ or implicit uncertainties learned by deep networks are often numerically valid but statistically poor: confidence intervals do not cover the errors and the filter is overconfident. This directly affects the safety and reliability of downstream planning, control and fusion.

4) *No guarantees of stability and constraints.* Positive definiteness of the covariance, numerical stability, divergence bounds and observability or controllability conditions may be violated in purely data-driven frameworks, and in long sequences such small violations can accumulate into serious failures.

5) *Inconsistent training and evaluation protocols.* Papers differ widely in datasets, noise settings, metrics (RMSE versus NLL or energy score (ES)), online versus offline estimation, and whether non-causal information is allowed. These differences hinder fair comparison and practical deployment.

6) *Computational cost and deployment.* Powerful networks such as Transformers face limits on computing power and energy in embedded, low-latency and multi-target scenarios. Maintainability also matters, including model version management and drift monitoring.

5.2 Future Directions

1) *Combining physical knowledge with learning.* Rather than purely black-box learning, future work should keep the known dynamics and measurement structure and learn only the components that are hard to model, such as time-varying noise, unmodeled disturbances, sensor bias and calibration drift, and nonlinear residuals. Such "gray-box" frameworks improve interpretability and cross-domain transfer.

2) *Better uncertainty representation (e.g., for $\mathbf{Q}$ and $\mathbf{R}$).* Training and evaluation should use metrics such as NLL, the continuous ranked probability score (CRPS), coverage and ES; heavy-tailed or mixture noise, heteroscedasticity and correlated noise should be modeled; and overconfident estimates should be avoided.

3) *Structurally constrained networks with stability guarantees.* Parameterizations can enforce positive definite covariances (e.g., Cholesky factors or projection onto the positive semi-definite cone), bounded gains and numerical stability, and observability or energy constraints can be built in as hard constraints or verifiable modules.

4) *Distributed and online adaptive systems.* Meta-learning, domain adaptation, test-time adaptation (TTA) and online estimation of noise and model bias are promising, together with change-point detection and robust updates for sudden anomalies.

5) *Integration with smoothing and factor-graph optimization.* Learning can be applied to factor weights, robust kernels (to suppress outliers), prior terms or initialization, combined with differentiable graph optimization, to balance global consistency and real-time performance.

6) *Standard benchmarks and engineering practice.* A unified open benchmark covering multiple sensors and modalities, several noise types, long sequences and OOD scenarios is needed. At the same time, lightweight models and a monitored fallback mechanism should be developed so that the system can revert to conventional filtering when the learned module fails.

6 Conclusion

This survey reviewed recent work that integrates deep learning with Kalman filtering within the Bayesian framework, organized by the degree of reliance on the physical model. Simulations with a matched model, measurement-model mismatch and maneuvering trajectories showed that model-constrained deep filters match or stay close to the EKF when the model is correct and are clearly more robust when it is not. KalmanNet and KalmanFormer run in a time of the same order as the conventional filters, whereas APBM is about five times slower than the EKF. We also summarized the main open challenges and future directions, and we hope the survey will be useful for both theoretical research and engineering applications in this area.

Data Availability Statement

Data will be made available on request.

Funding

This work was supported by the National Natural Science Foundation of China under Grant 62473304 and Grant U22A2045, and the Aeronautical Science Foundation of China under Grant 2025L015070001.

Conflicts of Interest

Deqiang Han served as a Specialty Chief Editor, and Ziling Wang served as a Managing Editor of the *Chinese Journal of Information Fusion* at the time of manuscript submission. To ensure the integrity of the peer-review process, neither Deqiang Han nor Ziling Wang was involved in the editorial handling, peer review, or decision-making process for this manuscript, which was handled independently by another editor. The remaining authors declare no conflicts of interest.

AI Use Statement

The authors declare that no generative AI was used in the preparation of this manuscript.

Ethical Approval and Consent to Participate

Not applicable.

References

[1] Khaleghi, B., Khamis, A., Karray, F. O., & Razavi, S. N. (2013). Multisensor data fusion: A review of the state-of-the-art. *Information Fusion, 14*(1), 28-44. [CrossRef]

[2] Hall, D. L., & Llinas, J. (1997). An introduction to multisensor data fusion. *Proceedings of the IEEE, 85*(1), 6-23. [CrossRef]

[3] Castanedo, F. (2013). A review of data fusion techniques. *The scientific world journal, 2013*(1), 704504. [CrossRef]

[4] Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Journal of Basic Engineering, 82*(1), 35-45. [CrossRef]

[5] Zhang, S., Yang, Z., & Cai, C. (2026). EKF-and ESKF-based GNSS/INS integrated navigation under the interaction multi-filter framework. *Machines, 14*(2), 217. [CrossRef]

[6] Yuan, Y., Li, F., Chen, J., Wang, Y., & Liu, K. (2024). An improved Kalman filter algorithm for tightly GNSS/INS integrated navigation system. *Mathematical Biosciences and Engineering, 21*(1), 963-983. [CrossRef]

[7] Reis, J., Yu, G., & Silvestre, C. (2023). Kalman-based velocity-free trajectory tracking control of an underactuated aerial vehicle with unknown system dynamics. *Automatica, 155*, 111148. [CrossRef]

[8] Badar, T., Backman, J., & Visala, A. (2024). Vehicle modeling and state estimation for autonomous driving in terrain. *Control Engineering Practice, 152*, 106046. [CrossRef]

[9] Tian, F., Guo, X., & Fu, W. (2024). Target tracking algorithm based on adaptive strong tracking extended Kalman filter. *Electronics, 13*(3), 652. [CrossRef]

[10] Moon, T. (2024, May). Beyond FFT: Precision vibration tracking with FMCW radar and Kalman estimators. In *2024 IEEE Radar Conference (RadarConf24)* (pp. 1-6). IEEE. [CrossRef]

[11] Zhang, J., Dang, X., & Hao, Z. (2024). Twpt: Through-wall position detection and tracking system using ir-uwb radar utilizing kalman filter-based clutter reduction and clean algorithm. *Electronics, 13*(19), 3792. [CrossRef]

[12] Zhang, N., Zhai, Y., Li, Y., Zhou, J., Zhai, M., Tang, C., & Xie, K. (2024). Kalman filtering to reduce measurement noise of sample entropy: An electroencephalographic study. *Plos one, 19*(7), e0305872. [CrossRef]

[13] Yadav, S., Saha, S. K., & Kar, R. (2023). An application of the Kalman filter for EEG/ERP signal enhancement with the autoregressive realisation. *Biomedical Signal Processing and Control, 86*, 105213. [CrossRef]

[14] Date, P., & Maunthrooa, J. (2025). Modelling and forecasting of exchange rate pairs using the Kalman filter. *Journal of Forecasting, 44*(2), 606-622. [CrossRef]

[15] Makatjane, K., & Xaba, D. (2026). A Combined Kalman Filter–LSTM to Forecast Downside Risk of BWP/USD Returns: A Bottom-Up Hierarchical Approach. *Forecasting, 8*(2), 21. [CrossRef]

[16] Schmidt, S. F. (1966). Application of state-space methods to navigation problems. In *Advances in control systems* (Vol. 3, pp. 293-340). Elsevier. [CrossRef]

[17] Julier, S. J., Uhlmann, J. K., & Durrant-Whyte, H. F. (1995, June). A new approach for filtering nonlinear systems. In *Proceedings of 1995 American Control Conference-ACC'95* (Vol. 3, pp. 1628-1632). IEEE. [CrossRef]

[18] Gordon, N. J., Salmond, D. J., & Smith, A. F. (1993, April). Novel approach to nonlinear/non-Gaussian Bayesian state estimation. In *IEE proceedings F (radar and signal processing)* (Vol. 140, No. 2, pp. 107-113). IEE. [CrossRef]

[19] Blom, H. A. P., & Bar-Shalom, Y. (1988). The interacting multiple model algorithm for systems with Markovian switching coefficients. *IEEE Transactions on Automatic Control*, 33(8), 780-783. [CrossRef]

[20] Bishop, C. M., & Bishop, H. (2024). *Deep Learning: Foundations and Concepts*. Springer. [CrossRef]

[21] Revach, G., Shlezinger, N., Ni, X., Escoriza, A. L., Van Sloun, R. J., & Eldar, Y. C. (2022). KalmanNet: Neural network aided Kalman filtering for partially known dynamics. *IEEE Transactions on Signal Processing, 70*, 1532-1547. [CrossRef]

[22] Han, X., Ding, L., Peng, C., Zeng, W., Zhang, X., Wen, Z., & Zheng, L. (2023, December). Multi-model KalmanNet for maneuvering target tracking. In *IET Conference Proceedings CP874* (Vol. 2023, No. 47, pp. 399-405). Stevenage, UK: The Institution of Engineering and Technology. [CrossRef]

[23] Shen, S., Chen, J., Yu, G., Zhai, Z., & Han, P. (2025). KalmanFormer: using transformer to model the Kalman Gain in Kalman Filters. *Frontiers in Neurorobotics, 18*, 1460255. [CrossRef]

[24] Mortada, H., Falcon, C., Kahil, Y., Clavaud, M., & Michel, J. P. (2025, September). Recursive Kalmannet: deep learning-augmented Kalman filtering for state estimation with consistent uncertainty quantification. In *2025 33rd European Signal Processing Conference (EUSIPCO)* (pp. 885-889). IEEE. [CrossRef]

[25] Cohen, N., & Klein, I. (2025). Adaptive kalman-informed transformer. *Engineering Applications of Artificial Intelligence, 146*, 110221. [CrossRef]

[26] Straka, O., Duník, J., Closas, P., & Imbiriba, T. (2025, July). Interpretable augmented physics-based model for estimation and tracking. In *2025 28th International Conference on Information Fusion (FUSION)* (pp. 1-8). IEEE. [CrossRef]

[27] Liu, H., Sun, X., Chen, Y., & Wang, X. (2025). Physics-informed data-driven autoregressive nonlinear filter. *IEEE Signal Processing Letters, 32*, 846-850. [CrossRef]

[28] Buchnik, I., Revach, G., Steger, D., Van Sloun, R. J., Routtenberg, T., & Shlezinger, N. (2023). Latent-KalmanNet: Learned Kalman filtering for tracking from high-dimensional signals. *IEEE transactions on signal processing, 72*, 352-367. [CrossRef]

[29] Chen, S., Zheng, Y., Lin, D., Cai, P., Xiao, Y., & Wang, S. (2025). MAML-KalmanNet: A neural network-assisted Kalman filter based on model-agnostic meta-learning. *IEEE Transactions on Signal Processing, 73*, 988-1003. [CrossRef]

[30] Pulido, A., Volle, K., Waters, K., Bell, Z. I., Ganesh, P., & Shin, J. (2025, July). Uncertainty-aware guidance for target tracking subject to intermittent measurements using motion model learning. In *2025 American Control Conference (ACC)* (pp. 3140-3145). IEEE. [CrossRef]

[31] Krishnan, R. G., Shalit, U., & Sontag, D. (2015). Deep kalman filters. *arXiv preprint arXiv:1511.05121*. [CrossRef]

[32] Ghosh, A., Honoré, A., & Chatterjee, S. (2024). DANSE: Data-driven non-linear state estimation of model-free process in unsupervised learning setup. *IEEE transactions on signal processing, 72*, 1824-1838. [CrossRef]

[33] Goel, G., & Bartlett, P. (2024, June). Can a transformer represent a Kalman filter?. In *6th Annual Learning for Dynamics & Control Conference* (pp. 1502-1512). PMLR. https://proceedings.mlr.press/v242/goel24a.html

[34] Rozet, F., & Louppe, G. (2023). Score-based data assimilation. *Advances in Neural Information Processing Systems, 36*, 40521-40541. [CrossRef]

[35] Si, P., & Chen, P. (2025, May). Latent-EnSF: A latent ensemble score filter for high-dimensional data assimilation with sparse observation data. In *International Conference on Learning Representations* (Vol. 2025, pp. 94879-94895).

[36] Sun, J. A., Fan, H., Gong, J., Fei, B., Chen, K., Ling, F., ... & Bai, L. (2025). Align-DA: Align Score-based Atmospheric Data Assimilation with Multiple Preferences. *Advances in Neural Information Processing Systems, 38*, 63539-63568. [CrossRef]

[37] Li, X. R., & Bar-Shalom, Y. (1996). Multiple-model estimation with variable structure. *IEEE Transactions on Automatic control, 41*(4), 478-493. [CrossRef]

[38] Jilkov, V. P., Angelova, D. S., & Semerdjiev, T. A. (1999). Design and comparison of mode-set adaptive IMM algorithms for maneuvering target tracking. *IEEE Transactions on Aerospace and Electronic Systems, 35*(1), 343-350. [CrossRef]

[39] Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*. [CrossRef]

[40] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems, 30*. [CrossRef]

[41] Imbiriba, T., Demirkaya, A., Duník, J., Straka, O., Erdoğmuş, D., & Closas, P. (2022, July). Hybrid

neural network augmented physics-based models for nonlinear filtering. In *2022 25th International Conference on Information Fusion* (*FUSION*) (pp. 1-6). IEEE. [CrossRef]

[42] Arasaratnam, I., & Haykin, S. (2009). Cubature kalman filters. *IEEE Transactions on automatic control*, *54*(6), 1254-1269. [CrossRef]

[43] Finn, C., Abbeel, P., & Levine, S. (2017, July). Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning* (pp. 1126-1135). PMLR. https://proceedings.mlr.press/v70/finn17a.html

[44] Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. In *International Conference on Learning Representations* (*ICLR*). arXiv:1412.6980. [CrossRef]



Deqiang Han is a Professor at Xi'an Jiaotong University, Xi'an, China. His research interests include pattern classification, information fusion, and uncertainty modeling and reasoning. (Email: deqhan@xjtu.edu.cn)



Wei Xiong is a Professor at Naval Aeronautical University, Yantai, China. His research interests include target tracking, state estimation and information fusion. (Email: xiongwei@csif.org.cn)



Jianhong Guo is a master's student at Xi'an Jiaotong University, Xi'an, China. His research interests include target recognition, information fusion, and deep learning-based target detection. (Email: 2211410784@stu.xjtu.edu.cn)



Ziling Wang is an Associate Professor at Naval Aeronautical University, Yantai, China. His research interests include target tracking, state estimation and information fusion. (Email: wangziling@126.com)