FlowEKF: Flow-based Extended Kalman Filter

Pham Hai Anh*, Tran Trong Duy*†, Do Hai Son‡¶, Karim Abed-Meraim§, Nguyen Linh Trung*

* University of Engineering and Technology, Vietnam National University, Hanoi, Vietnam
† CentraleSupélec, Université Paris-Saclay, CNRS, L2S, Gif-sur-Yvette, France
‡ School of Electrical Engineering, Computing and Mathematical Sciences, Curtin University, Australia
¶ Information Technology Institute, Vietnam National University, Hanoi, Vietnam
§ PRISME Laboratory (IUF member), University of Orléans, Orléans, France

***Abstract*—State estimation is a fundamental task in dynamical systems, with the Kalman filter (KF) and its extensions widely used due to their efficiency and optimality under controllable system models. However, in practice, the observation model is unknown or difficult to specify analytically, which limits the applicability of traditional Kalman filtering approaches. In this work, we propose a flow-based extended Kalman filter (FlowEKF) that leverages a real-valued non-volume preserving (real-NVP) model, a powerful generative model to learn the unknown observation model. We validate the effectiveness of FlowEKF on both simulated and real-world localization tasks. Experimental results demonstrate that FlowEKF significantly outperforms the classical Kalman Filter and achieves up-to 75% improvement in estimation accuracy compared to the state-of-the-art KalmanNet, particularly in scenarios where the observation model is unavailable.**

***Index Terms*—Extended Kalman filter, neural Kalman, observation model estimation, real-NVP.**

I. Introduction

State estimation is a fundamental problem in dynamical systems theory, involving the estimation of a system's state trajectory over time [1]. The most widely used approach is the Kalman Filter and its various extensions, such as the Extended Kalman Filter (EKF) and Unscented Kalman Filter (UKF) [2], [3]. Their applications range from autonomous navigation and wireless sensor networks to Internet-of-Things (IoT) localization and object tracking [4]–[6]. However, KF requires accurate knowledge of the error covariance matrices as well as precise state and observation models. Moreover, in practice, model imperfections, nonlinear system dynamics, and non-Gaussian noise can significantly degrade the filter's performance. Limitations of KF motivated numerous hybrid approaches [4] that incorporate deep learning (DL) techniques to enhance filtering accuracy, which are discussed in the following.

An early work on this approach is the neural extended Kalman filter (NEKF) [7], which enhances the traditional extended Kalman filter's (EKF) state estimation in monocular simultaneous localization and mapping (SLAM) by integrating a feedforward neural network into the prediction–correction cycle. Recent developments in generative modeling, especially normalizing flows, offer a powerful foundation for representing complex and nonlinear observation processes. The authors in [8] introduced the normalizing Kalman filter (NKF), which utilizes a real-valued non-volume preserving (real-NVP) [9], – a generative flow-based model to map pseudo-observations for forecasting real-world data and imputing missing values. They assumed the existence of a latent state that evolves according to linear dynamics, with potentially complex and nonlinear dependencies between the latent state and observations modeled by the real-NVP. Another recent hybrid approach is KalmanNet [10]. The method preserves the structural form of the traditional KF yet applies a recurrent neural network (RNN) for estimating the Kalman gain matrix as well as covariance noises. The authors also proposed a semi-supervised training method [11], which first trains KalmanNet offline, and then adapts in an unsupervised online manner to dynamics that differ from the offline trained model without providing ground-truth data. In addition, KalmanNet was adapted in Latent-KalmanNet [12] to deal with an unknown observation model by learning a latent representation from measurements using a deep neural network (DNN), and feeding the latent variables to KalmanNet to output the Kalman gain. The authors also enhanced KalmanNet with a compact hypernetwork that modulates the filter's behavior, allowing it to perform consistently across a range of unseen system conditions [13].

Although recent hybrid Kalman filters have improved tracking performance, a survey on machine learning integrated Kalman filter points out that complexity remained a challenge [4]. For example, KalmanNet requires a highly complex RNN architecture, which poses challenges in terms of computational efficiency. Moreover, some approaches inherently rely on prior knowledge of the transformation between the state variables and the observations, for example, estimating the expected signal strength at a given location using a known path-loss model. In real-world scenarios, these methods are not applicable when the underlying mathematical model is unknown.

Based on the advancements of generative flow-based models, we propose a novel *flow-based extended Kalman filter* (FlowEKF), which employs a flow-based neural network to learn the unknown observation model. We adopt the above real-NVP to learn the hidden mapping from system states to sensor measurements. The learned model is embedded into the

The research in this paper was supported by project QG.25.08, "Signal processing for 6G near-field communication and sensing", funded by Vietnam National University, Hanoi. Correspondence: Nguyen Linh Trung (linhtrung@vnu.edu.vn).

recursive filtering process and provides both predicted observations and the Jacobian for the Kalman gain computation.

The main contributions of this work are as follows:

- The proposed FlowEKF method can deal with unknown nonlinear observation models while having a lower complexity compared to existing methods.
- By using the invertible flow-based architecture, the learned observation model is inherently a vanilla state estimator called Vanilla normalizing flows (NF), and in turns, works regardless of sequence length, and its estimates can be used for initialization or reference.
- We validate the effectiveness of FlowEKF through experiments on localization with both synthetic and real-world data. On synthetic data, the proposed method outperforms the traditional EKF by approximately 0.05 meter. On the real-world data, FlowEKF improves approximately 98.9% compared to the classical KF and demonstrates up to 75% improvement in root mean squared error (RMSE) accuracy compared to KalmanNet.

II. Background

A. State-space model

In engineering, the behavior of a dynamical system is described by the state-space model (SSM). SSM represents the evolution over time of the state vector $\mathbf{x}_t \in \mathbb{R}^m$, and the observation vector $\mathbf{y}_t \in \mathbb{R}^n$. SSM consists of a state and an observation equation:

$$\mathbf{x}_t = \mathbf{F}\mathbf{x}_{t-1} + \mathbf{w}_t, \tag{1}$$

$$\mathbf{y}_t = \mathbf{H}\mathbf{x}_t + \mathbf{v}_t, \tag{2}$$

where $\mathbf{F} \in \mathbb{R}^{m\times m}$ is the state transition matrix, $\mathbf{H} \in \mathbb{R}^{n\times m}$ is the observation matrix, $\mathbf{w}_t \sim \mathcal{N}(0, \mathbf{Q})$ is the process noise and $\mathbf{v}_t \sim \mathcal{N}(0, \mathbf{R})$ is the measurement noise, $\mathbf{Q} \in \mathbb{R}^{m\times m}$ and $\mathbf{R} \in \mathbb{R}^{n\times n}$ are the process and observation noise covariance matrices respectively.

A common special case of the SSM is referred to as the linear Gaussian SSM, which is utilized by KF [2]. The process noise and the measurement noise in this case are independent, zero-mean, Gaussian distributed.

B. Kalman and Extended Kalman filters

KF is an optimal estimator for linear systems with Gaussian noise. It operates by recursively estimating the state of a system based on noisy measurements. KF consists of two main steps. The prediction step performs

$$\hat{\mathbf{x}}_{t|t-1} = \mathbf{F}\mathbf{x}_{t-1|t-1}, \tag{3}$$

$$\hat{\mathbf{P}}_{t|t-1} = \mathbf{F}\mathbf{P}_{t-1|t-1}\mathbf{F}^\top + \mathbf{Q}, \tag{4}$$

$$\hat{\mathbf{y}}_{t|t-1} = \mathbf{H}\hat{\mathbf{x}}_{t|t-1}, \tag{5}$$

and the update step performs

$$\mathbf{K}_t = \hat{\mathbf{P}}_{t|t-1}\mathbf{H}^\top(\mathbf{H}\hat{\mathbf{P}}_{t|t-1}\mathbf{H}^\top + \mathbf{R})^{-1}, \tag{6}$$

$$\mathbf{x}_{t|t} = \hat{\mathbf{x}}_{t|t-1} + \mathbf{K}_t(\mathbf{y}_t - \hat{\mathbf{y}}_{t|t-1}), \tag{7}$$

$$\mathbf{P}_{t|t} = (\mathbf{I} - \mathbf{K}_t\mathbf{H})\hat{\mathbf{P}}_{t|t-1}. \tag{8}$$

Here, $\mathbf{P}_t \in \mathbb{R}^{n\times n}$ is the state covariance matrix, and $\mathbf{K}_t \in \mathbb{R}^{n\times m}$ is the Kalman gain matrix.

For nonlinear systems, KF is extended using the first-order Taylor series approximation, leading to EKF. The state transition and observation models are nonlinear, as follows:

$$\mathbf{x}_t = f(\mathbf{x}_{t-1}) + \mathbf{w}_t, \quad \mathbf{w}_t \sim \mathcal{N}(0, \mathbf{Q}), \tag{9}$$

$$\mathbf{y}_t = h(\mathbf{x}_t) + \mathbf{v}_t, \quad \mathbf{v}_t \sim \mathcal{N}(0, \mathbf{R}). \tag{10}$$

The Jacobians $\mathbf{F}_t = \frac{\partial f}{\partial \mathbf{x}_t}$ and $\mathbf{H}_t = \frac{\partial h}{\partial \mathbf{x}_t}$ are used to linearize the system around the current state estimate, making EKF suitable for nonlinear systems.

Research Problem: This research focuses on applying the Extended Kalman Filter in cases where the observation model $h(\cdot)$ in Eq. (10) is unknown and nonlinear.

III. Flow-based extended Kalman filter

A. The proposed FlowEKF

While EKF is effective for systems with known and moderately nonlinear observation models, it faces significant challenges in many real-world scenarios, particularly in dynamic environments where the mathematical form of the observation function is unknown or too complex to model analytically. Variants of EKF typically either assume a simplified linear observation model or indirectly perform a transformation on a latent space [8], [12], which may limit flexibility and reduce accuracy. To address these limitation, we propose a FlowEKF to tackle the absence of observation model by using real-NVP to learn the observation model in Eq. (10). Its primary objective is to accurately approximate this unknown equation, which is a critical prerequisite for computing the Kalman gain $\mathbf{K}_t$ and the state covariance $\mathbf{P}_t$ during the update step in Eq. (6).

Fig. 1 illustrates the architecture of the proposed FlowEKF. The framework follows the standard EKF structure, consisting of a prediction step and an update step, with the key innovation introduced in the prediction step, and the observation is predicted via the flow-based model of real-NVP. In the prediction step, the prior state estimate $\hat{\mathbf{x}}_{t|t-1}$ and the state covariance estimate $\hat{\mathbf{P}}_{t|t-1}$ are computed. In the update step, instead of relying on a known or hand-designed observation model, *i.e.* Eq. (10), the system uses the real-NVP model to learn the hidden mapping from the prior state estimate $\hat{\mathbf{x}}_{t|t-1}$ to the predicted measurement $\hat{\mathbf{y}}_{t|t-1}$. A fully connected layer is applied before the real-NVP model to align the dimensionality of the state vector with that of the observation vector. The real-NVP also outputs the Jacobian matrix $\mathbf{H}_t$.

Normalizing flows are a class of generative models that learn the mapping between complex and simple distributions. Real-NVP is a specific implementation of NF that introduces a structured invertible transformation [9]. It consists of multiple affine coupling layers Eq. (13), where each layer modifies only a subset of input dimensions while leaving the remaining dimensions unchanged Eq. (15). Additionally, a fully connected layer is incorporated before the affine coupling layers to ensure

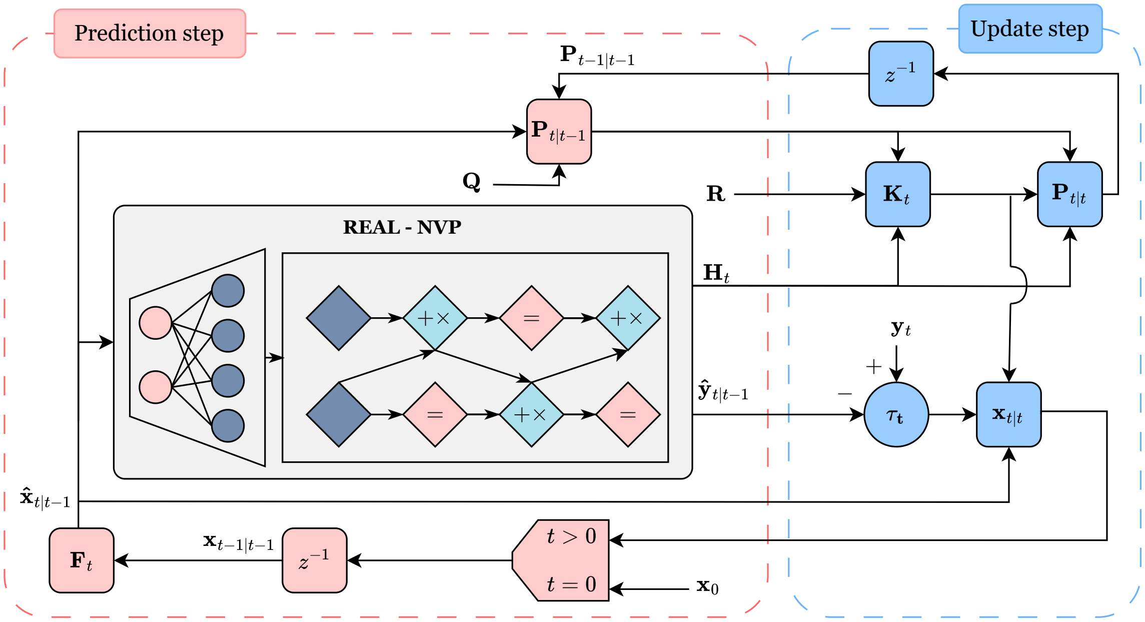


Fig. 1: FlowEKF block diagram.

dimensional consistency between the state and observation vectors Eq. (11). Given the prior state estimate vector $\hat{\mathbf{x}}_{t|t-1}$ as input, the predicted measurement vector $\hat{\mathbf{y}}_{t|t-1}$ is obtained as follows:

$$\begin{aligned} \mathrm{y}^0 &= \mathbf{A}\hat{\mathbf{x}}_{t|t-1} + b, && && (11)\\ \mathrm{x}^l &= \mathrm{y}^{l-1} && \text{for } l = 1,\dots,L && (12)\\ \mathrm{y}^l &= \mathrm{aff}(\mathrm{x}^l) && \text{for } l = 1,\dots,L && (13)\\ \hat{\mathbf{y}}_{t|t-1} &= \mathrm{y}^L, && && (14) \end{aligned}$$

where $\mathbf{A} \in \mathbb{R}^{n\times m}$, and let $d = \lfloor \frac{n}{2} \rceil$, $\mathrm{aff}(\cdot)$ transforms only one half of the input; for instance if the second part is transformed:

$$\begin{aligned} \mathrm{y}^l_{1:d} &= \mathrm{x}^l_{1:d}, \\ \mathrm{y}^l_{d+1:n} &= \mathrm{x}^l_{d+1:n} \odot \exp\left(\mathrm{S}(\mathrm{x}^l_{1:d})\right) + \mathrm{T}(\mathrm{x}^l_{1:d}), \end{aligned} \tag{15}$$

where S and T are respectively scale and translation functions learned by neural networks, and $\odot$ is the Hadamard product. Since one half is left unchanged, the Jacobian form of an affine coupling layer is an upper-triangle matrix and given by

$$J = \begin{bmatrix} I & 0 \\ \frac{\partial y_{d+1:n}}{\partial x_{1:d}} & \mathrm{diag}(\exp(\mathrm{S}(x_{1:d}))) \end{bmatrix}. \tag{16}$$

Unlike recurrent architectures, the model operates on individual input-output pairs and does not require time-series data; this simplifies training and deployment in real-time systems. We also note that, in this work, the real-NVP model is not used to learn a probabilistic generative function, but a transformation from state variables to observation variables. This is due to the fact that mean squared error (MSE) is preferred over log-likelihood in computing errors and differences in the filtering process. Our loss function is

$$\mathcal{L} = \frac{1}{n}\sum_{i=1}^{n}(\mathbf{y}_i - \hat{\mathbf{y}}_i)^2. \tag{17}$$

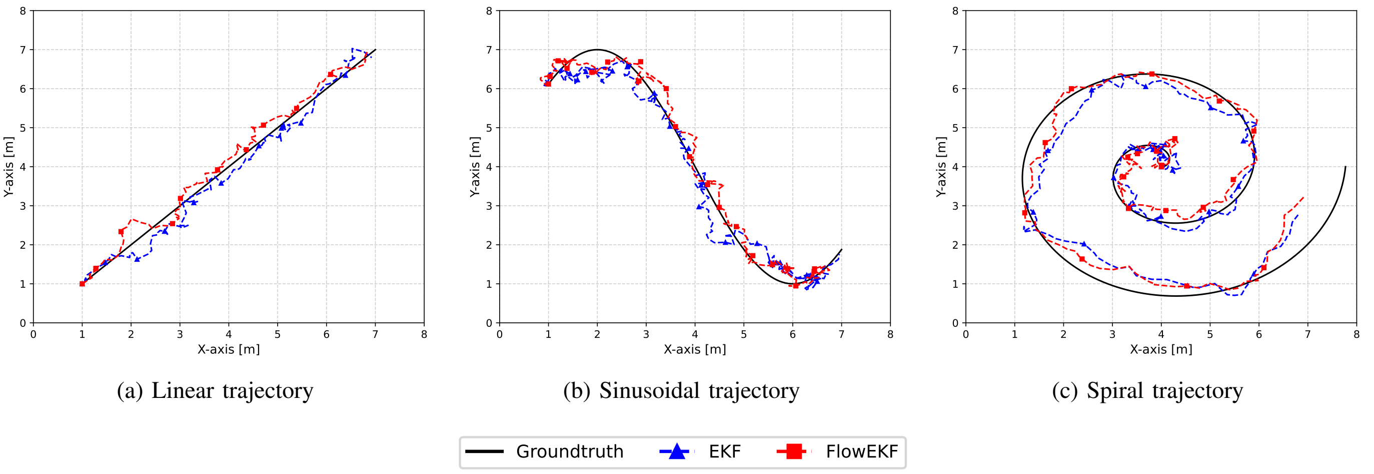
B. Comparison with existing methods

The computational complexity of the proposed FlowEKF is significantly lower than that of KalmanNet. With state vector of dimension m and observation vector of dimension n, the RNN block of KalmanNet has a complexity of: $\mathcal{O}(n^4 + m^4 + n^3m + nm^3 + n^2m^2)$, involving several higher-order polynomial terms. This makes KalmanNet much computationally expensive, particularly as m and n grow. FlowEKF, on the other hand, exhibits a relatively modest complexity of $\mathcal{O}(h^2 + nm + nh)$, where typical choice of hidden neurons is $h = Cn$. Eventually, the complexity of FlowEKF is $\mathcal{O}(n^2 + mn)$, which scales linearly with m and quadratically with n.

In NKF, the measurement generation is modeled as

$$\mathbf{y}_t = f_t(\mathbf{A}_t^\top \mathbf{l}_t + \varepsilon_t), \tag{18}$$

where $\mathbf{y}_t$ is the observed output, f_t is a nonlinear observation function modeled by real-NVP, $\mathbf{A}_t$ is the emission matrix mapping the latent state vector $\mathbf{l}_t$ to the observation vector, and $\varepsilon_t \sim \mathcal{N}(0, \Gamma_t)$ represents additive Gaussian observation noise vector. NKF assumes the existence of a pseudo-observation having a linear relationship with a latent Gaussian random state and performs the classical Kalman filter update using this observation. An RNN is used to predict the time-varying transition matrices and covariance matrices needed for forecasting the latent state evolution over time. Rather than preserving the linear transformation, our proposed FlowEKF avoids such architectural complexity by directly learning the measurement model through a lightweight, flow-based architecture, offering a significantly simpler alternative.



(a) Linear trajectory

(b) Sinusoidal trajectory

(c) Spiral trajectory

Fig. 2: Performance comparison of indoor simulation across different trajectories.

IV. Experiments

To evaluate the effectiveness of FlowEKF, we conduct a series of experiments in both simulated and real-world environments for the application of localization.

A. *Indoor simulations*

The aim of the experiment is to compare the performance of the FlowEKF with no access to observation functions to that of the regular EKF, which has complete information about the observation model.

1) Dataset: The indoor simulation is conducted in a simplified, obstruction-free indoor environment measuring $8 \times 8\,\mathrm{m}^2$. Four beacons at the corners of the area served as observation sources. The received signal strength (RSS) is calculated using a simplified path loss model defined as [14]:

$$P_{\text{received}} = P_{\text{transmit}} + K_{\text{dB}} - L_{\text{pathloss}} - S_{\text{shadowing}}. \tag{19}$$

The transmitted power is $P_{\text{transmit}} = -14.32$ dBm. The reference path loss is given by $K_{\text{dB}} = 20\log_{10}\left(\frac{\lambda}{4\pi d_0}\right)$, where the carrier frequency is $f = 3993.6$ MHz. The path loss attenuation is represented as $L_{\text{pathloss}} = 10\eta\log_{10}\left(\frac{d}{d_0}\right)$, with a path loss exponent of $\eta = 3.0$ and a reference distance of $d_0 = 1$ meter. Shadow fading is modeled as Gaussian noise, denoted by $S_{\text{shadowing}} \sim \mathcal{N}(0, 1.5^2)$ dB. A total of $10,000$ training points are uniformly sampled across the simulation space to train the real-NVP model. For each position, four RSS values are generated and included in $\mathbf{y} = [\text{RSS}_1, \text{RSS}_2, \text{RSS}_3, \text{RSS}_4]^\top$, one from each beacon, using the simplified path loss model. To evaluate performance, three types of motion trajectories are generated. Each trajectory consists of 400 points, simulating realistic object movement across the environment.

2) Model setup: Since FlowEKF retains the architecture of the standard EKF, both the full-information EKF and FlowEKF share the same setup for the motion model and covariance matrices, as described below:

a) State-space model: The motion follows a *constant velocity* model, with the state vector defined as

$$\mathbf{x}_t = [p_x, p_y, v_x, v_y]^\top, \tag{20}$$

where p_x, p_y are the positions and v_x, v_y are the velocities along the x and y directions, respectively. In our filter design, the motion covariance matrix $\mathbf{Q} = 0.005\,\mathbf{I}$, and the measurement covariance matrix $\mathbf{R} = \sigma^2\,\mathbf{I}$, where σ denotes the standard deviation of the received signal strength (RSS) noise. Finally, the initial estimation uncertainty is $\mathbf{P}_0 = 0.0001\,\mathbf{I}$.

b) Observation model: The observation vector $\mathbf{y}_t \in \mathbb{R}^4$ comprises 4 RSS values received from the corner beacons. The observation function $h(\mathbf{x}_t)$ is explicitly defined using the simplified path loss model for the full-information EKF. In contrast, FlowEKF employs a trained real-NVP model using 10,000 uniformly distributed points within an $8 \times 8\,\mathrm{m}^2$ area as input. Vanilla NF utilizes the inverse of a trained real-NVP model.

3) Results: The evaluation used 200 Monte Carlo trials of 400 steps each, with RMSE averaged across trials:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\mathbf{y}_i - \hat{\mathbf{y}}_i)^2}. \tag{21}$$

Compared models include the proposed FlowEKF, full-information EKF (baseline), and Vanilla NF. The Vanilla model is omitted from the figure because it fails to track the trajectory and produces a significantly high MSE.

TABLE I: RMSE comparison for simple path loss experiment

Method	EKF	FlowEKF	Vanilla NF
Linear	0.186	**0.139**	1273
Sinusoidal	0.210	**0.174**	1508
Spiral	0.337	**0.316**	1245

Across all three path types, FlowEKF closely followed the ground-truth trajectory, and was comparable to the classical EKF. Although both filters introduced more jitter on the tight inner loops, Fig. 2c, FlowEKF maintained the correct spiral direction and radius better, whereas the classical EKF occasionally drifted outward. In every scenario, FlowEKF achieved the lowest RMSE. EKF, which has access to the true observation model, still performed reasonably well, but was consistently outperformed by FlowEKF. Vanilla NF , without any tracking algorithms, failed to follow the groundtruth and deviated significantly. Table I demonstrates that learning the observation update via a flow model yields more accurate state estimates even without explicit knowledge of the true observation model. This success lays the foundation for further flow-based Kalman filter method to perform robustly on real-world data.

B. Real-world data experiments

The goal of these practical experiments are to assess the performance of the proposed FlowEKF on real-world data.

1) Datasets: We used the Michigan North Campus Long-Term Vision and LIDAR Dataset (NCLT) [15], which is an open-source benchmark dataset containing various sensory data obtained from a Segway robot such as Global Positioning System (GPS), Inertial Measurement Unit (IMU), Light Detection and Ranging (LiDAR), and odometry. In this study, only odometry measurements were used as input to the models. The odometry path's shape appeares reasonably consistent with the groundtruth, however, it increasingly diverges from the groundtruth as the trajectory progresses. Two experiments were conducted using sessions recorded on January 22, 2012 and April 29, 2012, respectively, following the same evaluation setup. To ensure consistency between model input and ground truth, the odometry data was temporally aligned to match the initial timestamp of the reference trajectory,

2) Model setup and evaluation method: All evaluated models in the real-world experiment were based on a constant velocity motion model. While the underlying dynamics are shared, the methods differ in how they process observations and perform state estimation.

The system state vector is defined as in Eq. (20), and evolves according to the linear dynamic model in Eq. (3). The process noise $\mathbf{Q} \in \mathbb{R}^{4\times4}$ and observation noise $\mathbf{R} \in \mathbb{R}^{2\times2}$ are both assumed to be identity matrices, that is, $\mathbf{Q} = \mathbf{I}_4$ and $\mathbf{R} = \mathbf{I}_2$.

In this setup, the observation vector consists of position measurements:

$$\mathbf{y}_t = \begin{bmatrix} p_x & p_y \end{bmatrix}^\top, \tag{22}$$

with training param as in Table II.

TABLE II: Training param used in the experiments

Parameter	**Value**
Hidden layers	4 layers, 64 nodes each
Activation function	tanh
Optimizer	Adam
Learning rate	10^{-3}
Weight decay	10^{-5}
Gradient clipping	Max norm of 1.0
Training iterations	200,000
Validation interval	Every 100 iterations
Data split	85% train, 10% validation, 5% test

The evaluation includes KF, KalmanNet with a linear identity observation matrix, and FlowEKF with real-NVP-based mapping. Models were tested across three setups: same-day, cross-day, and mixed-temporal training and errors are reported as RMSE.

3) Results:

a) Same-Day Performance: Fig. 3a illustrates that both data-driven filters outperform the classical KF. As shown in Table IIIa, FlowEKF reduced the RMSE from 108.91 m to 4.96 m on the 22-01 trajectory, while KalmanNet achieved a slightly better performance at 4.81 m. These results confirm the capability of the real-NVP model to effectively learn and generalize the observation mapping. Notably, on the 29-04 trajectory, FlowEKF achieved an RMSE of only 1.61 m, outperforming KalmanNet's 7.16 m by a margin of over 5.5 m. Vanilla NF, as the inverse of the real-NVP in FlowEKF, closely followed the ground truth, demonstrating strong filtering capability on its own. These results indicate that both approaches perform competitively under same-day conditions. When the training and test data share the same observation model, directly learning the measurement function with real-NVP can provide superior accuracy. Meanwhile, KalmanNet's end-to-end architecture helps reduce overall estimation error, showing the strength of learning the full filtering process.

TABLE III: RMSE results for Same-day, Cross-day, and Mixed-temporal evaluations

(a) Same-day experiment

Trajectory	KF	KalmanNet	FlowEKF	Vanilla NF
22-01	108.91	**4.81**	4.96	4.90
29-04	144.45	7.16	**1.61**	1.62

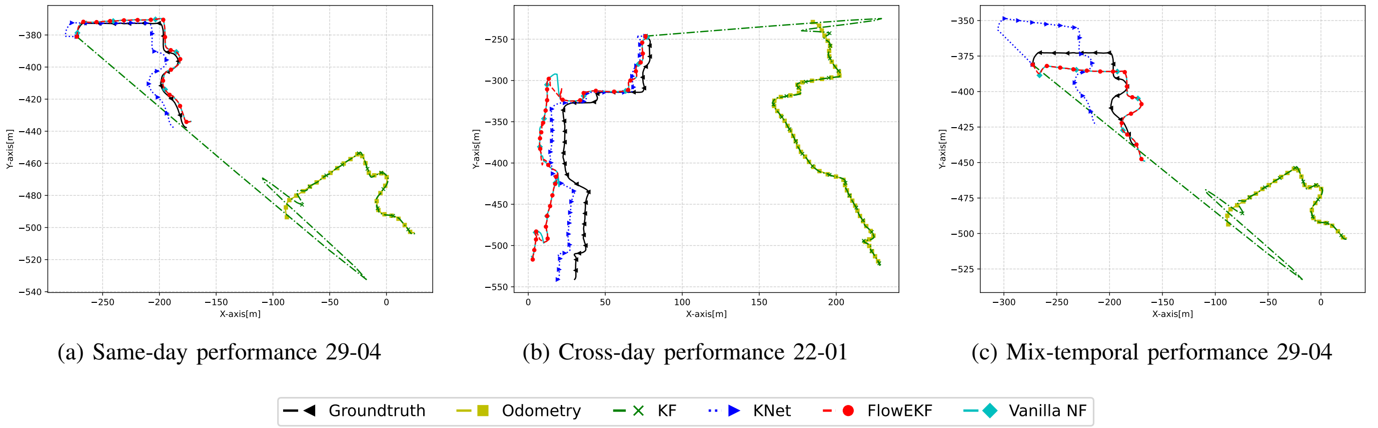
(b) Cross-day experiment

Trajectory	KF	KalmanNet	FlowEKF	Vanilla NF
22-01	108.91	**5.97**	18.70	18.81
29-04	144.45	**13.07**	19.69	19.80

(c) Mixed-temporal experiment

Trajectory	KF	KalmanNet	FlowEKF	Vanilla NF
22-01	108.91	10.80	**7.78**	7.79
29-04	144.45	26.74	**11.51**	11.56

b) Cross-Day Generalization: Fig. 3b shows the models' performance when generalizing across days. KalmanNet demonstrated better generalization, achieving RMSE of 5.97 m and 13.07 m on 22-01 and 29-04 respectively. In contrast, FlowEKF's RMSE increased to over 18 m on both trajectories, indicating sensitivity to changes in the observation model. Vanilla NF exhibited similar degradation. This performance gap can be attributed to the architectural differences: while FlowEKF performs state estimation using a pre-trained model, KalmanNet estimates the Kalman gain from additional temporal information from an RNN model, allowing it to better adapt to different observation model.



(a) Same-day performance 29-04

(b) Cross-day performance 22-01

(c) Mix-temporal performance 29-04

Fig. 3: Performance comparison on the real-world dataset.

c) Mixed-Temporal Training: In this experiment, data from both days are combined during training. In Fig. 3c, FlowEKF closely followed the ground-truth trajectory, particularly in regions with sharp turns and high drift. In contrast, KalmanNet diverged more significantly in these challenging areas. Vanilla NF's estimated trajectory remained consistently close to the ground truth, indicating that the learned inverse mapping captured the observation model effectively. As shown in Tabble IIIc, FlowEKF achieved the best overall performance with RMSE of 7.78 m and 11.51 m on 22-01 and 29-04, respectively, and outperformed significantly KalmanNet. These results suggest that the real-NVP module in FlowEKF effectively learned the measurement function that generalizes across sessions.

V. Conclusion

In this work, we have introduced the FlowEKF - a hybrid Kalman filtering framework designed to handle unknown and nonlinear observation models with lower computational complexity than existing approaches. Leveraging the invertible architecture, the learned observation function has inherently acted as a standalone state estimator. Experimental results on both simulated and real-world data for localization have validated the effectiveness of the proposed FlowEKF method. In the simplified path-loss simulation using RSS data, FlowEKF has consistently tracked the ground truth with approximately 5 cm lower error than the classical EKF. In the real-world experiment, FlowEKF has reduced the RMSE by over 100 m compared to the Kalman Filter and achieved an average error of 10.7 m, outperforming KalmanNet, which yield an average error of approximately 11.4 m. In addition to FlowEKF's improved accuracy, FlowEKF is significantly more computationally effective with complexity scaling quadratically, while KalmanNet involved fourth-order terms. Despite these promising results, a key limitation of the current framework is that the flow-based observation model is trained independently of the Kalman update, making it unable to optimize end-to-end with respect to the final state estimation errors. We will focus on integrating the network in the filtering process and adopting more flexible flow-based architectures.

References

[1] B.-N. Vo, B.-T. Vo, T. T. D. Nguyen, and C. Shim, "An Overview of Multi-Object Estimation via Labeled Random Finite Set," *IEEE Transactions on Signal Processing*, vol. 72, pp. 4888–4917, 2024.

[2] R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *Transactions of the ASME–Journal of Basic Engineering, Series D*, vol. 82, no. 1, pp. 35–45, 1960.

[3] M. Khodarahmi and V. Maihami, "A Review on Kalman Filter Model," *Archives of Computational Methods in Engineering*, vol. 30, no. 1, 2022.

[4] S. Kim, I. Petrunin, and H.-S. Shin, "A Review of Kalman Filter with Artificial Intelligence Techniques," in *Integrated Communication, Navigation and Surveillance Conference (ICNS)*, 2022.

[5] G. Liu, P. Neupane, H.-C. Wu, W. Xiang, L. Pu, and S. Y. Chang, "Novel Robust Indoor Device-Free Moving-Object Localization and Tracking Using Machine Learning With Kalman Filter and Smoother," *IEEE Systems Journal*, vol. 16, no. 4, pp. 6253–6264, 2022.

[6] Q. Ye, X. Fan, G. Fang, and H. Bie, "Exploiting Temporal Dependency of RSS Data with Deep Learning for IoT-Oriented Wireless Indoor Localization," *Internet Technology Letters*, vol. 6, no. 5, p. e366, 2022.

[7] Z. Miljković, N. Vuković, and M. Mitić, "Neural Extended Kalman Filter for Monocular SLAM in Indoor Environment," *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, vol. 230, no. 5, pp. 856–866, 2015.

[8] E. de Bézenac, S. S. Rangapuram, K. Benidis, M. Bohlke-Schneider, R. Kurle, L. Stella, H. Hasson, P. Gallinari, and T. Januschowski, "Normalizing Kalman Filters for Multivariate Time Series Analysis," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[9] L. Dinh, J. Sohl-Dickstein, and S. Bengio, "Density Estimation Using Real NVP," in *International Conference on Learning Representations (ICLR)*, 2017.

[10] G. Revach, N. Shlezinger, X. Ni, A. L. Escoriza, R. J. G. van Sloun, and Y. C. Eldar, "KalmanNet: Neural Network Aided Kalman Filtering for Partially Known Dynamics," *IEEE Transactions on Signal Processing*, vol. 70, pp. 1532–1547, 2022.

[11] G. Revach, N. Shlezinger, T. Locher, X. Ni, R. J. G. van Sloun, and Y. C. Eldar, "Unsupervised Learned Kalman Filtering," in *30th European Signal Processing Conference (EUSIPCO)*, 2022, pp. 1571–1575.

[12] I. Buchnik, G. Revach, D. Steger, R. J. G. van Sloun, T. Routtenberg, and N. Shlezinger, "Latent-KalmanNet: Learned Kalman Filtering for Tracking From High-Dimensional Signals," *IEEE Transactions on Signal Processing*, vol. 72, pp. 352–366, 2024.

[13] X. Ni, G. Revach, and N. Shlezinger, "Adaptive KalmanNet: Data-Driven Kalman Filter with Fast Adaptation," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2024, pp. 5970–5974.

[14] A. Goldsmith, *Wireless Communications*. Cambridge University Press, 2005.

[15] N. Carlevaris-Bianco, A. K. Ushani, and R. M. Eustice, "The University of Michigan North Campus Long-Term Vision and LIDAR Dataset," 2016. [Online]. Available: https://robots.engin.umich.edu/nclt