Adaptive Filtering Algorithm based on Reinforcement Learning

1st Hongrui Lin
School of Automation
Southeast University
Nanjing, China
975682939@qq.com

2nd Kelin Lu
School of Automation
Southeast University
Nanjing, China
klu@seu.edu.cn

3nd Yibai Wang
School of Automation
Southeast University
Nanjing, China
1024338113@qq.com

2024 36th Chinese Control and Decision Conference (CCDC) | 979-8-3503-8778-0/24/$31.00 ©2024 IEEE | DOI: 10.1109/CCDC62350.2024.10587732

Abstract—**Real-time state estimation of dynamic systems is a basic task in control theory and signal processing. Kalman filter is a minimum complex optimal state estimator for linear state space models with fully known system parameters, which is widely used in industrial production. However, in practical problems, accurate models and linear state-space models are often difficult to satisfy. In this paper, by combining reinforcement learning and Kalman filter, an adaptive strategy of state space model is designed based on PPO algorithm to improve the performance of state estimation under the inaccuracy conditions of the system model. The algorithm is called Reinforcement Learning Kalman filter algorithm (RLKF). Finally, we prove by numerical results that the Reinforcement Learning Kalman filtering algorithm overcomes the model mismatch and nonlinearity, and the accuracy is better than the traditional filtering algorithm.**

Index Terms—**Kalman filtering, reinforcement learning, adaptive filtering, state estimation**

I. Introduction

The Kalman Filter (KF), as a fundamental method for real-time state estimation, finds widespread application in various practical problems, such as target tracking [1], navigation and guidance [2], fault diagnosis [3], machine control [4]. Traditional Kalman filters are capable of addressing linear problems, while for nonlinear problems, extended Kalman filters [5], unscented Kalman filters [6], and other variants have been proposed. These methods primarily rely on model approximation to linearize the system, and their performance significantly degrades when confronted with strong nonlinearity [7]. Both the original Kalman filter and its variants are model-based filtering algorithms that depend on precise knowledge of the underlying state space (SS) model. However, in practical scenarios, the system's underlying dynamic model is often complex and challenging to estimate accurately as an easily manageable SS model. Inaccurate SS models may lead to a degradation in estimation performance.

To mitigate the negative impact of model inaccuracy, several adaptive filtering methods have been proposed in existing literature. One approach involves adjusting the Kalman gain in the Kalman filter to achieve adaptive filtering by modifying the Kalman gain based on Mahalanobis distance [8], [9]. Another approach involves end-to-end learning using neural networks to learn complex dynamic models from a large amount of real data, including the use of RNN [10], LSTM [11] and attention models [12] to fit the model. However, similar to model-based filtering methods, data-based filtering methods rely on the effectiveness and accuracy of the data. In practical problems, data is often mixed with a significant amount of noise. The lack of training data and the unpredictability of neural network models greatly limit the application of data-driven approaches in real-world scenarios. To address these issues, a study [13] incorporated neural networks into the Kalman filter flow solely for the adaptive computation of the Kalman gain, yielding superior results with a smaller training dataset. In addition, literature [14], [15] adaptively adjusted the noise covariance matrix by means of reinforcement learning to improve the filtering accuracy.

This paper combines the ideas from the above-mentioned methods and proposes a reinforcement learning-based adaptive Kalman filter to enhance the estimation performance of systems with partially known parameters. Proximal Policy Optimization (PPO) is an efficient implementation of reinforcement learning, where an intelligent agent learns how to select appropriate strategies in uncertain environments through trial and error [16], [17]. After the agent selects an appropriate action based on the environment, it receives corresponding rewards to evaluate the quality of the action. The main goal is to choose an appropriate policy function that maximizes the cumulative reward. PPO algorithm uses an Actor-Critic architecture, where the Actor controls the system to produce suitable actions, and the Critic evaluates the current action. Based on the effectiveness of the PPO algorithm and its success in state estimation applications [15], this paper implements adaptive filtering based on PPO.

The main contributions of this paper are as follows. Firstly, the paper combines the PPO algorithm with the KF algorithm to introduce the RLKF algorithm. This algorithm adjusts the state space model to fit the real system through interaction with the environment. To the best of our knowledge, state space model adaptive methods based on reinforcement learning have not been effectively addressed. RLKF, employing two parallel filters, the reference filter and the correction filter, introduces a correction factor through a reinforcement learning algorithm to continuously update the correction filter based on the difference from the innovation value of the reference filter to improve model accuracy. Secondly, this paper considers the

Authorized licensed use limited to: ULAKBIM UASL - MIDDLE EAST TECHNICAL UNIVERSITY. Downloaded on March 27,2025 at 23:14:12 UTC from IEEE Xplore. Restrictions apply.

case of imprecise system model parameters and conducts both linear and nonlinear numerical studies. Compared to traditional filters, RLKF exhibits better performance.

II. Problem formulation

State space model is used to describe the evolution of the state of a dynamic system over time. At each discrete sampling time $t \in \mathbf{Z}$, a Gaussian, nonlinear, continuous state-space model can be expressed as

$$\mathbf{x}_t = \mathbf{f}\left(\mathbf{x}_{t-1}\right) + \mathbf{w}_t \tag{1}$$

$$\mathbf{y}_t = \mathbf{h}\left(\mathbf{x}_t\right) + \mathbf{v}_t \tag{2}$$

Where, $\mathbf{x}_t$ denotes the potential state of the system at time t, which iterates through the state transition function $\mathbf{f}(\cdot)$. Let $\mathbf{y}_t$ denotes the observation vector of the latent state $\mathbf{x}_t$ at time t, which is obtained according to the measurement function $\mathbf{h}(\cdot)$. w_t and v_t denote the process noise and observation noise, respectively, which are usually assumed to follow a normal distribution with mean 0, variances R and Q are independent of each other, i.e

$$E\left(\mathbf{w}_t\right) = 0, \operatorname{Cov}\left(\mathbf{w}_k, \mathbf{w}_j\right) = \mathbf{Q}\delta_{kj} \tag{3}$$

$$E\left(\mathbf{v}_t\right) = 0, \operatorname{Cov}\left(\mathbf{v}_k, \mathbf{v}_j\right) = \mathbf{R}\delta_{kj} \tag{4}$$

$$Cov\left(\mathbf{w}_k, \mathbf{v}_j\right) = 0 \tag{5}$$

The true process and measurement noise covariance matrices Q and R are both positive definite symmetric matrices, where δ_{kj} is the Kronecker-delta function equal to 1 if $\mathrm{k} = \mathrm{j}$ and 0 at the rest of the time. In practice, the state transition function $\mathbf{f}(\cdot)$ of the system is determined by approximately modeling complex systems, the observation function $\mathbf{h}(\cdot)$ is determined by the characteristics of the observation instrument, and the covariance matrix of the process noise and the observation noise of the system is determined by the investigation and experience of the actual situation, which introduces imprecision and uncertainty into the state space model. For example, in target tracking problem, $\mathbf{x}_t$ represents the position, velocity, and acceleration state of the target, $\mathbf{y}_t$ represents the observations obtained from a variety of sensors, such as displacement and velocity. Due to the complexity of maneuvering target motion, it is difficult to describe $\mathbf{f}(\cdot)$ precisely, which leads to model mismatch and introduces imprecision. At the same time, the covariance matrices Q and R are difficult to obtain exactly, leading to the uncertainty of the system. Due to the dependence of the Kalman filter on the model parameters, the imprecise model will lead to a sharp degradation of the filtering performance.

For the considered system, the Kalman filtering process is to predict and update recursively to estimate the state. Taking the Extended Kalman Filter (EKF) algorithm as an example, the iterative process of EKF is given in Algorithm 1.

In Algorithm 1, $\mathbf{F}_t$ and $\mathbf{H_t}$ denote the linearization of the state transition matrix and the observation matrix at time $\mathbf{t}$, respectively, i.e

$$\mathbf{F_t} = \left.\frac{\partial f}{\partial x}\right|_{\mathbf{x}=\hat{\mathbf{x}}_{t-1}} \tag{6}$$

Algorithm 1: EKF algorithm.

```
1: Initialize $x_0, P_0, Q, R$
2 : for $\mathbf{t} = 1, 2,\ \mathrm{L}, \mathbf{T}$ do
3 : $\hat{\mathbf{x}}_{t-1} = \mathbf{f}\left(\hat{\mathbf{x}}_{t-1}\right) >$ predict
4 : $\mathbf{P}_{\mathrm{t}t-1} = \mathbf{F}_{\mathrm{t}}\mathbf{P}_{\mathrm{t}-1}\mathbf{F}_{\mathrm{t}}^{T} + \mathbf{Q}$
5 : $\mathbf{K}_{\mathrm{t}} = \mathbf{P}_{\mathrm{t}t-1}\mathbf{H}_{\mathrm{t}}^{T}\left(\mathbf{H}_{\mathrm{t}}\mathbf{P}_{\mathrm{t}t\mathrm{t}-1}\mathbf{H}_{\mathrm{t}}^{T} + \mathbf{R}\right)^{-1}$
6 : $\Delta\mathbf{y}_{\mathrm{t}} = \mathbf{y}_{\mathrm{t}} - \mathbf{h}\left(\hat{\mathbf{x}}_{\mathrm{t|t}-1}\right)$
7 : $\hat{\mathbf{x}}_{\mathrm{t}} = \hat{\mathbf{x}}_{\mathrm{t}t\mathrm{t}-\mathrm{I}} + \mathbf{K}_{\mathrm{t}}\Delta\mathbf{y}_{\mathrm{t}} >$ update
8 : $\mathbf{P_t} = \left(\mathbf{I} - \mathbf{K_t}\mathbf{H_t}\right)\mathbf{P}_{\mathrm{t|t}-1}\left(\mathbf{I} - \mathbf{K_t}\mathbf{H_t}\right)^{T} + \mathbf{K_t}\mathbf{R}\mathbf{K_t}^{T}$
9 : end for
10 : end function
```

$$\mathbf{H_t} = \left.\frac{\partial h}{\partial x}\right|_{\mathbf{x}=\hat{\mathbf{x}}_{\mathrm{t|t}-1}} \tag{7}$$

$\hat{\mathbf{x}}_{\mathbf{t}}$ denotes the estimation $\mathbf{x}_t$ of the latent state of the system at time $\mathbf{t}, \mathbf{P_t}$ is the estimated value of the error covariance matrix, $\Delta\mathbf{y}_{\mathrm{t}}$ represents the innovation value,$\mathbf{K}_t$ represents the Kalman gain, Q and R represents the covariance matrix of the process noise and observation noise, respectively.

The modeling of state transition function $\mathbf{f}(\cdot)$ and observation function $\mathbf{h}(\cdot)$ is the key problem of EKF algorithm design. In order to deal with the problem of decreasing accuracy of filtering algorithm caused by model mismatch. In this paper, we focus on a scenario where we have partial knowledge of the system model, that is, we approximate the state transition function $\mathbf{f}(\cdot)$ and the observation function $\mathbf{h}(\cdot)$ to some extent, which are not consistent with the true model. namely

- $\mathbf{f}(\cdot)$ and $\mathbf{h}(\cdot)$ can be nonlinear
- $\mathbf{f}(\cdot)$ and $\mathbf{h}(\cdot)$ can be the neural network black box model
- $\mathbf{f}(\cdot)$ and $\mathbf{h}(\cdot)$ are imprecise
- Lack of real data on system state

Our goal is to combine the imprecise system model with the sensor observations to approximate the true system model in real time, so as to improve the filtering accuracy of the model-based filter.

III. Reinforcement Kalman Filter

In this paper, Reinforcement Kalman filter algorithm is proposed for the dynamic system with imprecise state space model. Reinforcemen Kalman filter algorithm realizes model adaptation based on PPO algorithm, so we will briefly review the PPO algorithm here. The state $\mathbf{S}$ and action $\mathbf{A}$ of PPO algorithm are continuous, and the agent object continuously learns through trial and error through the policy function, and evaluates the state and action through the reward given by the environment. Specifically, at a specific time $\mathbf{t}$, the agent observes the state $\mathbf{S}$ of the system and selects the corresponding action $\mathbf{A}$. After executing the action, the system will give some feedback to the current action, which is a reward $\mathbf{r}$ to evaluate the quality of the action in the short term. The state $\mathbf{S}$ will be updated to $\mathbf{S}'$, and the system selects a new action $\mathbf{A}'$ based on the new state. The agent will iteratively update the appropriate action based on the experience gained from the previous interaction with the environment, in order to maximize the cumulative reward. This process is repeated until the desired optimization result is reached. Algorithm 2 shows the basic flow of the PPO algorithm.

Authorized licensed use limited to: ULAKBIM UASL - MIDDLE EAST TECHNICAL UNIVERSITY. Downloaded on March 27,2025 at 23:14:12 UTC from IEEE Xplore. Restrictions apply.

```
Algorithm 2: PPO algorithm
1: Initial policy paramters $\theta$, threshold $\varepsilon$
2: for k $= 1, 2, \cdots, T$ do
3: Collect trajectories D on policy $\pi = \pi(\theta)$
4: Estimate advantage $\hat{\mathrm{A}}$
5: Policy update $\theta = \arg\max \ell(\theta)$
6: by takding K steps of minibatch SGD (via Adam)
7: $\ell \cdot (\theta) = E\left[\sum[\min(r(\theta)\hat{\mathrm{A}}, \mathrm{clip}(r(\theta), 1-\varepsilon, 1+\varepsilon)\hat{\mathrm{A}}]\right]$
8: end for
```

Next we will elaborate how the PPO algorithm is combined with the Kalman filter. Let's first clarify that our goal is to adaptively fit an imprecise state-space model to improve filtering accuracy. To do this, we introduce a correction factor $\mathbf{G_t}$ and associate the actions $\mathbf{A_t}$ of the PPO algorithm with the correction factor in two ways:

- Mode 1: the dimension of $\mathbf{G_t}$ is the same as the dimension of the state $\mathbf{x}_t$, and the first moment of the state is modified in real time through the dot product operation of the two, that is:

$$\hat{\mathbf{x}}_{\mathbf{t}|\mathbf{t-1}} = \mathbf{f}\left(\hat{\mathbf{x}}_{t-1}\right) + \mathbf{G_t} \cdot \hat{\mathbf{x}}_{\mathbf{t}-1} \tag{8}$$

$$\mathbf{G_t} = [a_1, a_2, \cdots, a_m] \tag{9}$$

  Where a_i represents the ith dimensional action at the current time, m represents the dimension of the state. This method only changes the first moment of the system state, which is more suitable for the case of low model inaccuracy and low nonlinearity of the system

- Mode 2: the dimension $\mathbf{G_t}$ is the same as the dimension of the state transition matrix $\mathbf{F_t}$, and the first moment and second moment of the state are simultaneously modified by matrix multiplication, and the correction factor is introduced into the whole filtering process, that is:

$$\mathbf{x}_{\mathbf{t}tt-1} = \mathbf{f}\left(\mathbf{x}_{t-1}\right) + \mathbf{G_t}\mathbf{x}_{\mathbf{t}-1} \tag{10}$$

$$\mathbf{P}_{t|t-1} = \left(\mathbf{F_t} + \mathbf{G_t}\right)\mathbf{P}_{t-1}\left(\mathbf{F_t} + \mathbf{G_t}\right)^T + \mathbf{Q} \tag{11}$$

$$\mathbf{G_t} = \mathrm{diag}\left[a_1, a_2, \cdots, a_m\right] \tag{12}$$

Where diag $[\cdot]$ represents the diagonal matrix, the corresponding Kalman gain will also change due to the change of the second moment, and the correction factor will run through the whole process of filtering. This method is more suitable for the case of high inaccuracy of the model and high nonlinearity of the system.

The state of the system in PPO algorithm is the estimated value of the latent state vector $\hat{\mathbf{x}}_t$. At each state transition, we will obtain a reward $\mathbf{r}$. In order to make the filter adaptively fit the true model, we set a benchmark filter, and the reward is constructed by the mean square deviation of the innovation value between the reinforcement Kalman filter and the benchmark Kalman filter. Namely

$$\mathbf{r} = \left[\left(\Delta\mathbf{y_t^b}\right)^T\left(\Delta\mathbf{y_t^b}\right)\right]^{0.5} - \left[\left(\Delta\mathbf{y_t^R}\right)^T\left(\Delta\mathbf{y_t^R}\right)\right]^{0.5} \tag{13}$$

Where b is the base Kalman filter and R is the Reinforcement Kalman filter. Generally speaking, a smaller innovation value means that the model has a more accurate representation ability for the target dynamic system. Based on the cumulative reward maximization goal, the agent will adjust the correction factor $\mathbf{G_t}$ in real time to approach the true model of the system and improve the filtering accuracy.

Based on the previous description, the specific flow of the RLKF algorithm is shown in Algorithm 3.

```
Algorithm 3: RLKF algorithm
1 : Initialize agent $\mathbf{f}(), \mathbf{h}(), \mathbf{Q}, \mathbf{R}$
2 : Initialize $\hat{\mathbf{x}}_0^{\mathrm{R}} = \hat{\mathbf{x}}_0^{\mathrm{b}} = \hat{\mathbf{x}}_0, \mathbf{P}_0^{\mathrm{R}} = \mathbf{P}_0^{\mathrm{b}} = \mathbf{P}_0$
3 : $\mathbf{k} \leftarrow 0 \triangleright$ transation count
4 : for each epoch do
5 : $\mathbf{R} \leftarrow 0 \triangleright$ epoch reward
6 : Reset states$_0$,$\mathbf{G}_0, \hat{\mathbf{x}}_0^{\mathrm{R}}, \hat{\mathbf{x}}_0^{\mathrm{b}}, \mathbf{P}_0^{\mathrm{R}}, \mathbf{P}_0^{\mathrm{b}}$
7 : for $\mathbf{t} = 1, 2, \cdots, \mathbf{T}$ do
8 : $\mathbf{A}_t \leftarrow$ Actor_Net$(\mathbf{S}_t)$
9 : $\mathbf{G}_t = \mathbf{G}_{t-1} + diag[a_t^1, a_t^2, \cdots a_t^m]$
10 : $[\hat{\mathbf{x}}_t^{\mathrm{R}}, \mathbf{P}_t^{\mathrm{R}}, \Delta\mathrm{y}_t^{\mathrm{R}}] \leftarrow EKF(\hat{\mathbf{x}}_{t-1}^{\mathrm{R}}, \mathbf{P}_{t-1}^{\mathrm{R}}, \mathbf{Q}, \mathbf{R}, \mathbf{G}_t)$
11 : $[\hat{\mathbf{x}}_t^{\mathrm{b}}, \mathbf{P}_t^{\mathrm{b}}, \Delta\mathrm{y}_t^{\mathrm{b}}] \leftarrow EKF(\hat{\mathbf{x}}_{t-1}^{\mathrm{b}}, \mathbf{P}_{t-1}^{\mathrm{b}}, \mathbf{Q}, \mathbf{R})$
12 : $\mathbf{R} \leftarrow \mathbf{R} + \left[(\Delta\mathrm{y}_t^{\mathrm{b}})^T(\Delta\mathrm{y}_t^{\mathrm{b}})\right]^{0.5.} - \left[(\Delta\mathrm{y}_t^{\mathrm{R}})^T(\Delta\mathrm{y}_t^{\mathrm{R}})\right]^{0.5}$
13 : $\mathbf{k} \leftarrow \mathbf{k} + 1$
14 : end for
15 : end for
16 : return $\{\hat{\mathbf{x}}^{\mathrm{R}}\}, \{\mathbf{P}^{\mathrm{R}}\}$
```

In the algorithm, steps 10 and 11 involve the parallel filtering of two filters, one is the reference filter, the other is the target filter, where $\hat{\mathbf{x}}_\mathbf{t}^\mathbf{R}$ and $\mathbf{P}_\mathbf{t}^\mathbf{R}$ represent the state and covariance matrix of the target filter, $\hat{\mathbf{x}}_\mathbf{t}^\mathrm{b}$ and $\mathbf{P}_\mathbf{t}^\mathrm{b}$ represent the state and covariance matrix of the reference filter, respectively. The EKF algorithm in step 10 will introduce the algorithm of Equations (10) - (11) into step 3 and step 4 of Algorithm 1, so as to introduce the correction factor into the process of Kalman filtering to achieve the purpose of adaptive filtering. The reward of the system consists of the difference between the innovation values of the parallel filters. The overall architecture of the algorithm adopts the Actor-Critic architecture, and the experience replay mechanism is used to make full use of the training data.

IV. Simulation results and analysis

In this section, we will verify the effectiveness of the Reinforcement Kalman filter through a series of numerical simulations. In all numerical simulations, unless particularly stated, the process and measurement noise of the SS model for synthetic data are diagonal noise covariance matrices, i.e

$$\mathbf{Q} = q^2 \cdot \mathbf{I}, \mathbf{R} = r^2 \cdot \mathbf{I} \tag{14}$$

In the following experiments, the term full information means that the Kalman filter is working under an accurate state-space model, that is, the system parameters $\mathbf{f}(\cdot)$ and $\mathbf{h}(\cdot)$ are the same as the generative model of the true data, and the term partial information means that the Kalman filter is working under an inaccurate state-space model, that is, the model parameters are different from the parameters of the generative model of the true data. In both cases, the noise covariance matrix of the system is unknown.

Authorized licensed use limited to: ULAKBIM UASL - MIDDLE EAST TECHNICAL UNIVERSITY. Downloaded on March 27,2025 at 23:14:12 UTC from IEEE Xplore. Restrictions apply.

In Experiment 1, we compare the performance of RLKF and KF under linear systems. We evaluate the two algorithms on a linear synthetic dataset. In the linear model, we focus on the 3×3 state space model, where the sampling time $\mathbf{T} = 200$, while defining the signal-to-noise ratio(SNR)

$$v = \frac{q^2}{r^2} \tag{15}$$

Take $v \in \{0.1, 1, 5, 10, 50\}$, the imprecision of the model is achieved by inserting an evolution matrix under the original state transition matrix, i.e

$$\mathbf{F}_{\alpha^\circ} = \begin{pmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{pmatrix} \cdot \mathbf{F}, \alpha \in \{10^\circ, 15^\circ, 20^\circ\} \tag{16}$$

Later RLKF and KF will be iterated according to $\mathbf{F}_{\alpha^\circ}$. Such a scenario simulates the actual situation where the approximate modeling of a complex system differs from the true model. Figure 1 shows the mean square error of the filtering results of RLKF and KF in different cases, and Figure 2 shows the error of RLKF and KF in time step $\mathbf{T}$. As you can see from the figure, KF's performance drops sharply when the SS model does not match, while RLKF fits the model well with the correction factor and achieves a smaller mean square error (MSE). When the SNR is small, the filtering effect of RLKF has exceeded that of the model-matched Kalman filter, indicating that RLKF can also overcome the filtering error caused by the uncertainty of the noise covariance to some extent.

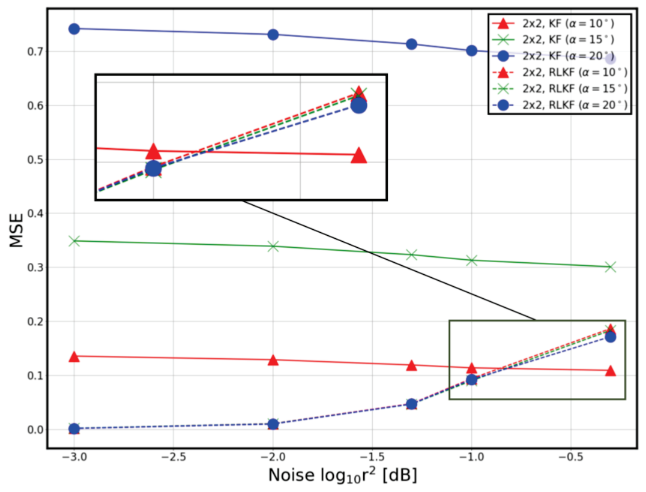


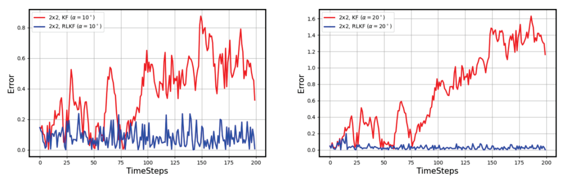
Fig. 1. Mean square error of the state of the linear SS model.

Next we consider a nonlinear SS model whose state transition function and observation function can be expressed as

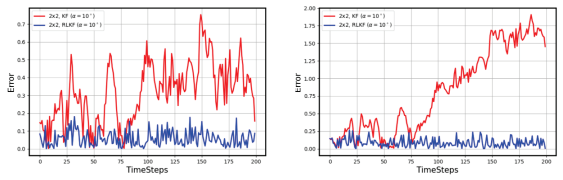
$$\mathbf{f}(\mathbf{x}) = \alpha \cdot \sin(\beta \cdot \mathbf{x} + \chi) + \delta \tag{17}$$

$$\mathbf{h}(\mathbf{x}) = a \cdot (b \cdot \mathbf{x} + c)^2 \tag{18}$$

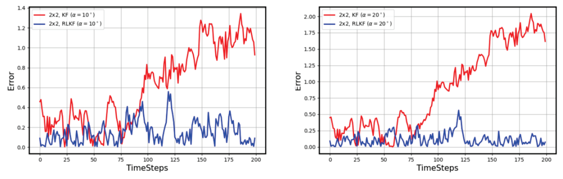
The specific parameters in the model are shown in Table 2, we set the sampling time $\mathbf{T} = 20$, SNR $v \in \{0.1, 1, 5, 10, 50\}$,



(a) v = 0.1



(b) v = 1



(c) v = 10

Fig. 2. MSE comparison among filtering algorithms.

and the MSE under different SNR for the full information and partial information models are shown in figure. 3 . It can be seen from the figure that EKF achieves the smallest MSE value, which is closest to MMSE when the model is completely known. We also note that RLKF can also approximate the real model and achieve similar results as EKF. However, when the model is partially known, the state-space model used by the filter is slightly different from the real system, and the performance of EKF is significantly degraded due to the model mismatch, while RLKF can overcome this mismatch and achieve better MSE. Therefore, we can conclude that RLKF can continuously learn the true model of the system from experience and overcome the performance degradation problem caused by model mismatch.

TABLE I
PARAMETERS OF THE NONLINEAR MODEL

	α	β	χ	δ	a	b	c
Full	0.9	1.1	0.1π	0.01	1	1	0
Partial	1	1	0	0	1	1	0

V. Conclusion

For a class of stochastic systems with inaccuracy state space model, an adaptive filtering algorithm RLKF is proposed in this paper. The algorithm mainly embed PPO algorithm into Kalman filter algorithm to adaptively adjust the state space model of the system, which can select the appropriate correction factor through interaction with the environment

Authorized licensed use limited to: ULAKBIM UASL - MIDDLE EAST TECHNICAL UNIVERSITY. Downloaded on March 27,2025 at 23:14:12 UTC from IEEE Xplore. Restrictions apply.

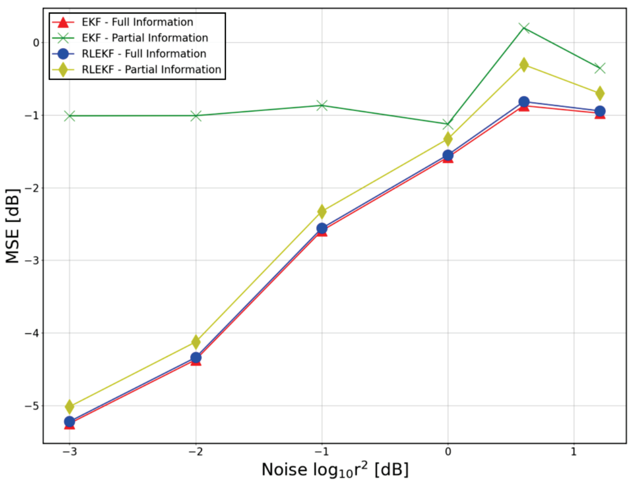
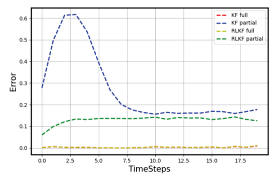
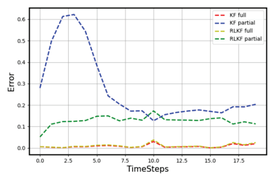


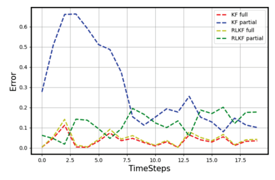
Fig. 3. Mean square error of the nonlinear SS model states.



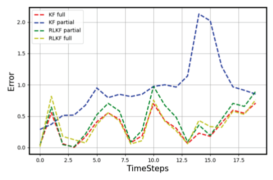
(a) v = 0.1



(b) v = 1



(c) v = 10



(d) v = 100

Fig. 4. MSE comparison among filtering algorithms.

to approximate the true model of the system. At the same time, it overcomes the dependence problem of the model-based Kalman filtering algorithm on the accurate model and the dependence problem of the neural network-based filtering algorithm on the real data. Simulation results show that RLKF can overcome the problem of model inaccuracy and improve the filtering accuracy effectively.

REFERENCES

[1] N. H. Nguyen and K. Doanay, "Improved pseudolinear kalman filter algorithms for bearings-only target tracking," *IEEE Transactions on Signal Processing*, vol. PP, no. 23, pp. 1–1, 2017.
[2] X. Hong, X. Wenchong, Y. Huadong, D. Keqing, L. Weijian, and W. Yongliang, "Fixed-point iteration gaussian sum filtering estimator with unknown time-varying non-gaussian measurement noise," *Signal Processing*, vol. 153, pp. 132–142, 2018.
[3] X. Chen and Z. Feng, "Order spectrum analysis enhanced by surrogate test and vold-kalman filtering for rotating machinery fault diagnosis under time-varying speed conditions," *Mechanical Systems and Signal Processing*, vol. 154, no. 331, p. 107585, 2021.
[4] Xu, Ying, Chen, Biyun, Chi, and Cheng, "Estimation of road friction coefficient and vehicle states by 3-dof dynamic model and hsri model based on information fusion," *Asian Journal of Control: Affiliated with ACPA, the Asian Control Professors' Association*, vol. 20, no. 3, pp. 1067–1076, 2018.
[5] M. Gruber, "An approach to target tracking," *an approach to target tracking*, 1967.
[6] S. J. Julier and J. K. Uhlmann, "A new extension of the kalman filter to nonlinear systems," *Proceedings of SPIE - The International Society for Optical Engineering*, vol. 3068, pp. 182–193, 1999.
[7] E. A. Wan and R. V. D. Merwe, "The unscented kalman filter for non-linear estimation," in *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No.00EX373)*, 2000.
[8] D. Wang, H. Zhang, and B. Ge, "Adaptive unscented kalman filter for target tacking with time-varying noise covariance based on multi-sensor information fusion.," *Sensors (Basel, Switzerland)*, vol. 21, no. 17, 2021.
[9] J. A. Ting, E. Theodorou, and S. Schaal, "A kalman filter for robust outlier detection," in *IEEE/RSJ International Conference on Intelligent Robots Systems*, pp. 1514–1519, 2007.
[10] B. Lim, S. Zohren, and S. Roberts, "Recurrent neural filters: Learning independent bayesian filtering steps for time series prediction," in *International Joint Conference on Neural Network*, 2020.
[11] S. Jung, I. Schlangen, and A. Charlish, "A mnemonic kalman filter for non-linear systems with extensive temporal dependencies," *IEEE Signal Processing Letters*, vol. 27, no. 99, pp. 1005–1009, 2020.
[12] S. K. Roy, A. Nicolson, and K. K. Paliwal, "Deeplpc-mhanet: Multi-head self-attention for augmented kalman filter-based speech enhancement," *IEEE Access*, vol. 9, pp. 70516–70530, 2021.
[13] 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.
[14] X. Dai, V. Nateghi, H. Fourati, and C. Prieur, "Q-learning-based noise covariance adaptation in kalman filter for marg sensors attitude estimation," in *2022 IEEE International Symposium on Inertial Sensors and Systems (INERTIAL)*, pp. 1–6, 2022.
[15] J. Gu, J. Li, and K. Tei, "A reinforcement learning approach for adaptive covariance tuning in the kalman filter," in *2022 IEEE 5th Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC)*, vol. 5, pp. 1569–1574, 2022.
[16] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017.
[17] S. Kakade and J. Langford, "Approximately optimal approximate reinforcement learning," 2008.

Authorized licensed use limited to: ULAKBIM UASL - MIDDLE EAST TECHNICAL UNIVERSITY. Downloaded on March 27,2025 at 23:14:12 UTC from IEEE Xplore. Restrictions apply.