Kalman-SSM: Modeling Long-Term Time Series With Kalman Filter Structured State Spaces

Zheng Zhou, Xu Guo, Yu-Jie Xiong, and Chun-Ming Xia

***Abstract*—In the field of time series forecasting, time series are often considered as linear time-varying systems, which facilitates the analysis and modeling of time series from a structural state perspective. Due to the non-stationary nature and noise interference in real-world data, existing models struggle to predict long-term time series effectively. To address this issue, we propose a novel model that integrates the Kalman filter with a state space model (SSM) approach to enhance the accuracy of long-term time series forecasting. The Kalman filter requires recursive computation, whereas the SSM approach reformulates the Kalman filtering process into a convolutional form, simplifying training and enhancing model efficiency. Our Kalman-SSM model estimates the future state of dynamic systems for forecasting by utilizing a series of time series data containing noise. In real-world datasets, the Kalman-SSM has demonstrated competitive performance and satisfactory efficiency in comparison to state-of-the-art (SOTA) models.**

***Index Terms*—Kalman filter, state-space methods, time series analysis.**

I. Introduction

Accurately forecasting time series, such as weather [1] and electricity forecasts [2], can help people make better decisions and develop better solutions. Time series delves into every aspect of people's lives. However, real-world time series data has non-stationary properties and is affected by random disturbances, which poses great challenges to forecasting models.

Previous work has attempted different methods to explore modeling methods for time series forecasting [3], and the development of deep learning has provided more possibilities for modeling time series forecasting. Some famous models for time series forecasting tasks include: Informer [4], Reformer [5], iTransformer [6], PatchTST [7], these Transformer based models utilize attention mechanisms to calculate sequence correlations for modeling time series; DLinear [8], FreTS [9], these linear or MLP based models utilize sequence decomposition and frequency domain transformation to capture time series features for prediction

Received 18 July 2024; revised 2 September 2024; accepted 7 September 2024. Date of publication 10 September 2024; date of current version 20 September 2024. This work was supported in part by the Science and Technology Commission of Shanghai Municipality under Grant 21DZ2203100 and in part by the National Natural Science Foundation of China under Grant 62006150. The associate editor coordinating the review of this article and approving it for publication was Prof. Yue Deng. *(Corresponding author: Yu-Jie Xiong.)*

The authors are with the School of Electronic and Electrical Engineering, Shanghai University of Engineering Science, Shanghai 201620, China (e-mail: xiong@sues.edu.cn).

Digital Object Identifier 10.1109/LSP.2024.3457862

SSM demonstrates the potential for simultaneously optimizing performance and computational complexity, and has become a promising architecture for sequence modeling [10], [11]. These models can be interpreted as a combination of RNNs and Convolutional Neural Networks (CNNs) [12], which can effectively be computed as recursive or convolutional, with linear or near linear scaling in sequence length.

However, most of these methods use the method of stabilizing time series data to reduce the interference of noise in the data, which to a certain extent can mix real data and noise to cause data offset. Kalman filter was proposed to solve the problem of linear filter of discrete data [13], which estimates the optimal state of the system through input and output observation data. Real observation data usually contains noise and interference in the system, and Kalman filter will filter out noise in the process of estimating the state. Therefore, due to the denoising function of Kalman filter, it has been applied in various fields such as communication [14], navigation [15], and control [16]. [17] presents an improved Extended Kalman Filter (EKF) for a bioinspired navigation system, designed to handle outliers effectively in an integrated setup with insect-like polarization sensors. [18] discusses an improved Cubature Kalman Filter designed for spacecraft attitude estimation, enhancing robustness and accuracy under nonlinear and non-Gaussian conditions through adaptive algorithms with fading factors. [19] presents an ERC model that integrates Extended Kalman Filters and Residual Neural Networks to improve manipulator calibration accuracy by addressing both geometric and non-geometric errors effectively. [20] offers an overview of current data-driven vibration control methods, focusing on design and optimization. It highlights advancements in sensor networks and aims to guide future research. [21] presents a model combining Kalman filters with latent factor analysis to better handle dynamic and sparse data, improving accuracy in modeling temporal patterns. [22] introduces an efficient robot calibration algorithm using an Unscented Kalman Filter with Variable Step-Size Levenberg–Marquardt, improving accuracy and efficiency. Kalman-SSM consistently achieves SOTA performance across a variety of publicly available real-world datasets spanning multiple domains. As shown in Fig. 1, our model demonstrates advantages in predictive performance and efficiency. Overall, our contributions are summarized as follows:

- we propose a model of integrating Kalman filter and SSM to model time series data. Kalman filter can estimate the system state well in noisy environments, and SSM can transform the recursive recursive form of Kalman filter into

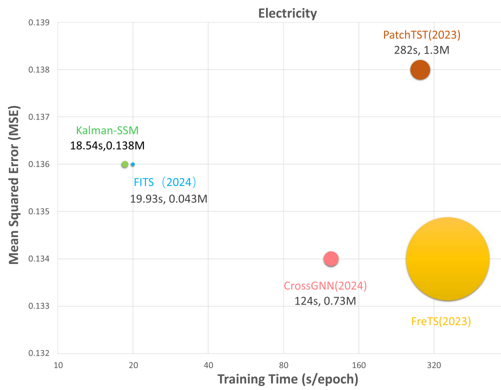

Fig. 1. Comparison of Kalman-SSM and four baselines in terms of MSE, training time, and GPU memory.

a global convolution form, avoiding the problem of slow training in recursive form and having better performance in noisy environments.
- We propose Kalman-SSM consistently demonstrates SOTA performance on real-world data.

II. Method

A. Problem Statement

Given a historical sequence data consisting of C variables and L time steps $X = [X_1^t, \ldots, X_C^t]_{t=1}^L$, where X_i^t is the value of the i_{th} variable at the t_{th} time step. The time series forecasting task is to predict the future T time steps $Y = [Y_1^t, \ldots, Y_C^t]_{t=1}^T$, where T represents the forecasting window.

B. Preliminaries

1) State Space Model: The State Space Model is a mathematical model used to describe the evolution of dynamic systems. It is applicable to systems whose states vary over time and may be subject to random disturbances. The evolution of the internal state of the system can be represented by using a first-order differential equation, as shown in (1). The input sequence $x(t) \in R^D$ can be mapped from the latent state $h(t) \in R^N$ to the output sequence $y(t) \in R^N$:

$$h(t)' = Ah(t) + Bx(t),$$
$$y(t) = Ch(t), \tag{1}$$

where $A \in R^{N\times N}$, $B, C \in R^{N\times D}$ are all learnable matrices, and A is the state matrix.

Since the collected time series are discrete points, we discretize the continuous representation of SSM. On time series data, SSM can be represented in the following:

$$x_k = \bar{A}x_{k-1} + \bar{B}u_k,$$
$$y_k = Cx_k, \tag{2}$$

where the input sequence is u_k, the hidden state is x_k and output sequence is y_k. Expanding y_k, it is not difficult to find the convolutional form of SSM as follows:

$$y_k = \overline{\boldsymbol{C}\boldsymbol{A}}^k\overline{\boldsymbol{B}}u_0 + \overline{\boldsymbol{C}\boldsymbol{A}}^{k-1}\overline{\boldsymbol{B}}u_1 + \cdots + \overline{\boldsymbol{C}\boldsymbol{B}}u_k,$$
$$Y = \overline{\boldsymbol{K}} * u. \tag{3}$$

Where $\overline{\boldsymbol{K}}$ is the convolution kernel and can be represented as:

$$\overline{\boldsymbol{K}} = \left(\overline{\boldsymbol{C}\boldsymbol{A}}^k\overline{\boldsymbol{B}}, \overline{\boldsymbol{C}\boldsymbol{A}}^{k-1}\overline{\boldsymbol{B}}, \ldots, \overline{\boldsymbol{C}\boldsymbol{B}}\right). \tag{4}$$

2) Kalman Filter: The Kalman filter is an efficient recursive filter that estimates the states of a linear dynamic system, providing optimal estimation in the presence of noise.

Kalman filter primarily involves two phases: forecasting and update. It assumes that both system and measurement noises are Gaussian white noises, and the system itself is linear. The model can be described as follows:

$$h_k = Ah_{k-1} + Bx_k + w_k,$$
$$y_k = Ch_k + v_k, \tag{5}$$

where x_k is the current state, A is the state transition matrix, B is the control input matrix, x_k is input sequence, y_k is output sequence and $w_t \sim \mathcal{N}(0, Q)$, $v_t \sim \mathcal{N}(0, R)$ are process noise and observation noise.

The prediction phase involves forecasting the next state and covariance:

$$\hat{h}_{k|k-1} = A\hat{h}_{k-1|k-1} + Bx_k,$$
$$P_{k|k-1} = AP_{k-1|k-1}A^T + Q_k, \tag{6}$$

where $P_{k|k-1}$ is forecasting covariance matrix and Q_k is the process noise covariance matrix.

The update phase updates the state and covariance based on the latest measurement data:

$$K_k = P_{k|k-1}C^T(CP_{k|k-1}C^T + R_k)^{-1}, \tag{7}$$

where K_k is Kalman Gain, R_k is the measurement noise covariance matrix.

$$\hat{h}_{k|k} = \hat{h}_{k|k-1} + K_k(y_k - C\hat{h}_{k|k-1}),$$
$$P_{k|k} = (I - K_kC)P_{k|k-1}, \tag{8}$$

where I is the identity matrix.

3) Kalman-SSM: Kalman filter is used in deep learning, which approximates recursive form and updates parameters K_k and P_k in a cyclic iterative manner. However, as the length of the input sequence increases, the overall training performance of the model will sharply decline. To address this issue, we combine SSM and Kalman filter. SSM can use convolutional form during training to avoid lengthy training processes, while the large convolutional kernel has a global perspective to provide more comprehensive historical information for time series forecasting. The overall process framework is shown in Fig. 2. Similarly, we define the state space of Kalman-SSM as follows:

$$h_t = Ah_{t-1} + Bx_t + w_t,$$
$$y_t = Ch_t + v_t, \tag{9}$$

where $w_t \sim \mathcal{N}(0, Q)$, $v_t \sim \mathcal{N}(0, R)$ are process noise and observation noise, respectively.

Furthermore, we write it in the following convolutional form:

$$Y = C\left[\bar{A}^n\bar{A}^{n-1}\cdots\bar{A}I\right]\left(\bar{B}\begin{bmatrix}x_0\\x_1\\\vdots\\x_n\end{bmatrix} + \begin{bmatrix}w_0\\w_1\\\vdots\\w_n\end{bmatrix}\right) + v_n. \tag{10}$$

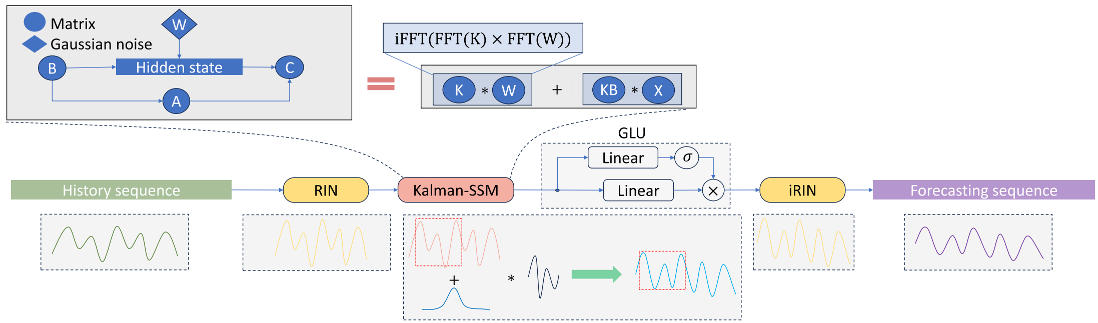

Fig. 2. Pipeline of Kalman-SSM, firstly, the time series data distribution and scale are adjusted through normalization. Then, the K-SSM module estimates the dynamic system and accelerates the training process through a global convolution form equivalent to recursive updates. This method gives the model a global receptive field and high efficiency. Finally, the prediction results are output through a gating mechanism, and the data distribution and scale are restored through inverse normalization.

To streamline the notation in your formula, we mark $C[\bar{A}^n\bar{A}^{n-1}\cdots\bar{A}I]$ as $\bar{K}$, $\bar{K}\bar{B}$ as M, $\bar{K}W$ as N. Consequently, the formula can be simplified as follows:

$$Y = M * X + N, \tag{11}$$

where M is the convolutional kernel, while N is the noise estimation term and the v_n is a single sample of Gaussian noise, we ignore v_n during the modeling process.

$$M * X = iFFT(FFT(M) \times FFT(X)),$$

$$X(k) = FFT(x) = \sum_{n=0}^{N-1} x(n)e^{-i2\pi kn/N},$$

$$x(n) = iFFT(X) = \frac{1}{N}\sum_{k=0}^{N-1} X(k)e^{i2\pi kn/N}. \tag{12}$$

Due to the fact that convolution in the time domain is equivalent to multiplication in the frequency domain, Fourier transform is applied to perform multiplication on the convolution formula 11, and then the inverse Fourier transform is used to return to the time domain to obtain the convolution result, further improving efficiency.

In this way, we represent the Kalman filter process in convolutional form, achieving better predictive performance through noise estimation.

III. Experiments and Results

A. Experimental Settings

We conduct extensive experiments on six real-world time series benchmarks and compare them with SOTA methods to evaluate the performance and efficiency of Kalman-SSM.

Datasets: All datasets are publicly available and widely used real-world datasets from different fields, mainly including Electricity, Weather, Exchange and ETT used by Autoformer [23]. Details regarding the granularity, length, and the number of variables are summarized in Table I .

Baselines: We compare Kalman-SSM with SOTA time series forecasting models to evaluate performance and efficiency, including PatchTST [7], iTransformer [6], Reformer [5], Informer [4] and DLinear [8], FreTS [9].

TABLE I
Summary of Datasets

Datasets	ETTh1&ETTh2	ETTm1 &ETTm2	Exchange-Rate	Weather
variable number	7	7	8	21
Length	17,420	69,680	7,588	52,696
Granularity	1hour	5min	1hour	10min

TABLE II
Comparison of Training Time and Parameters

models	Kalman-SSM	FreTS	iTransformer	PatchTST	Reformer
training time (epoch/s)	18.54	363	19.12	282	55
Parameters (M)	0.138	23	0.443	1.3	0.331

Implementation details: According to FITS [24] settings, we set the length of the input sequence to $T = 720$ and the length of the forecasting sequence to $S \in \{96, 192, 336, 720\}$. To prevent information leakage, normalization is applied to each time slot independently rather than to the entire dataset. The Mean Squared Errors (MSE) [25] and Mean Absolute Errors (MAE) [26] are utilized as evaluation metrics. All experiments were conducted using a single NVIDIA RTX 3080 10 GB GPU and implemented in PyTorch [27], under identical settings to ensure experimental fairness. This consistent setup allows for direct comparison of results, minimizing variability and ensuring that differences in performance are attributable to the methods tested rather than experimental conditions.

B. Overall Performance

We compare and analyze the overall performance of the Kalman-SSM model and other baseline models on all datasets, and the results are shown in Tables II and III. We can draw the following conclusions and analysis:

1) Kalman-SSM has shown satisfactory results in both time and parameters, with training time reduced by 20× and parameter count reduced by 100× compared to the largest model.
2) Kalman-SSM exhibits significant advantages on Weather and ETT datasets. The characteristic of ETT and weather datasets is that they have fewer variables, most of which are non periodic. However, Kalman-SSM still achieves the best performance on them.

TABLE III
LONG-TERM FORECASTING COMPARISON. THE BEST RESULTS ARE IN **BOLD** AND THE SECOND BEST RESULTS ARE UNDERLINED. USE THE MEAN TO BALANCE THE DIFFERENCES IN DIFFERENT FORECASTING LENGTHS AND COUNT THE BEST AND SECOND RESULT

Models		**Kalman-SSM(ours)**		DLinear		FreTS		iTransformer		PatchTST		Reformer		Informer	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Weather	96	**0.149**	**0.203**	0.170	0.230	0.154	0.214	0.169	0.222	**0.149**	0.205	0.398	0.421	0.217	0.294
	192	**0.194**	**0.245**	0.220	0.280	0.199	0.261	0.211	0.250	0.199	0.248	0.651	0.562	0.303	0.353
	336	**0.241**	**0.282**	0.258	0.310	0.253	0.309	0.273	0.302	0.248	0.291	0.641	0.554	0.485	0.477
	720	**0.310**	**0.332**	0.321	0.364	0.329	0.366	0.325	0.343	0.317	0.335	0.713	0.606	0.732	0.614
	Avg	**0.224**	**0.266**	0.242	0.296	0.234	0.288	0.245	0.281	0.228	0.270	0.601	0.536	0.434	0.435
Exchange	96	0.094	0.219	**0.087**	**0.213**	0.525	0.531	0.118	0.253	0.124	0.251	1.117	0.902	1.104	0.866
	192	0.197	**0.316**	**0.196**	0.337	0.958	0.731	0.232	0.355	0.287	0.390	1.158	0.913	1.175	0.875
	336	0.395	0.442	**0.269**	**0.387**	1.100	0.765	0.434	0.489	0.702	0.586	1.291	0.959	1.297	0.960
	720	1.225	0.800	**0.946**	**0.738**	2.518	1.188	1.055	0.755	1.371	0.867	1.530	1.051	1.165	0.908
	Avg	0.478	0.444	**0.375**	**0.419**	1.275	0.804	0.460	0.463	0.621	0.524	1.274	0.956	1.185	0.902
ETT-h1	96	**0.381**	**0.408**	0.385	0.410	0.480	0.480	0.404	0.432	0.431	0.441	1.006	0.773	1.248	0.874
	192	0.446	0.454	**0.427**	**0.437**	0.552	0.530	0.465	0.473	0.497	0.486	1.025	0.779	1.246	0.870
	336	**0.474**	**0.471**	0.479	0.478	0.586	0.549	0.501	0.499	0.610	0.553	1.181	0.821	1.355	0.864
	720	**0.499**	**0.501**	0.527	0.533	0.645	0.577	0.601	0.563	0.720	0.597	1.126	0.818	1.297	0.885
	Avg	**0.450**	**0.459**	0.455	0.465	0.566	0.534	0.493	0.492	0.565	0.519	1.085	0.798	1.287	0.873
ETT-h2	96	**0.290**	**0.351**	0.296	0.362	0.428	0.454	0.368	0.393	0.318	0.384	2.774	1.306	3.023	1.440
	192	0.352	**0.391**	**0.345**	0.394	0.603	0.540	0.437	0.438	0.398	0.428	4.740	1.683	4.546	1.654
	336	**0.379**	**0.414**	0.455	0.460	0.532	0.508	0.453	0.455	0.394	0.432	4.275	1.656	3.780	1.536
	720	**0.399**	**0.437**	0.782	0.621	1.053	0.733	0.440	0.466	0.519	0.506	3.335	1.437	4.495	1.825
	Avg	**0.355**	**0.398**	0.470	0.459	0.654	0.559	0.425	0.438	0.407	0.438	3.781	1.521	3.961	1.614
ETT-m1	96	**0.295**	**0.346**	0.318	0.366	0.334	0.382	0.332	0.382	0.325	0.375	0.652	0.593	0.836	0.678
	192	**0.339**	**0.371**	0.350	0.383	0.365	0.397	0.364	0.400	0.360	0.400	0.780	0.644	0.833	0.670
	336	**0.374**	**0.391**	0.375	0.396	0.415	0.431	0.394	0.415	0.402	0.428	0.950	0.723	1.067	0.801
	720	**0.427**	**0.421**	**0.427**	0.427	0.497	0.488	0.447	0.446	0.571	0.493	1.114	0.803	1.413	0.964
	Avg	**0.359**	**0.382**	0.368	0.393	0.403	0.425	0.384	0.411	0.415	0.424	0.874	0.691	1.037	0.778
ETT-m2	96	0.168	**0.257**	**0.167**	0.259	0.188	0.278	0.185	0.268	0.179	0.267	0.753	0.678	0.512	0.547
	192	0.238	**0.308**	**0.237**	0.316	0.250	0.316	0.269	0.328	0.256	0.318	1.114	0.826	1.535	0.956
	336	0.285	**0.338**	**0.282**	0.342	0.312	0.353	0.334	0.362	0.320	0.366	2.218	1.158	2.171	1.117
	720	**0.361**	**0.389**	0.389	0.516	0.391	0.415	0.413	0.415	0.415	0.427	2.766	1.254	6.218	1.946
	Avg	**0.263**	**0.323**	0.269	0.358	0.285	0.341	0.300	0.343	0.293	0.345	1.713	0.979	2.609	1.142
1^{st} Count		**45**		16		0		0		1		0		0	

TABLE IV
ABLATION EXPERIMENT OF KALMAN FILTER (INPUT LENGTH $L = 720$ AND FORECASTING LENGTH $S = 96$)

Models		Exchange	Weather	ETTh1	ETTh2	ETTm1	ETTm2
Kalman ✓	MSE	0.094	0.149	0.381	0.290	0.295	0.168
	MAE	0.219	0.203	0.408	0.351	0.346	0.257
Kalman ✗	MSE	0.110	0.144	0.400	0.311	0.294	0.171
	MAE	0.240	0.198	0.418	0.364	0.343	0.260

C. Ablation Study

We conduct ablation experiments on the Kalman filter to assess its efficacy within our model. The results of these experiments are presented in Table IV. It is evident that models incorporating the Kalman filter exhibit enhanced performance across most datasets. This improvement is particularly notable in complex datasets, such as the ETT datasets, where the performance significantly surpasses that of models without the Kalman filter. On simpler datasets, the advantage of the Kalman filter is less pronounced but still observable. These findings suggest that the Kalman filter can effectively predict future trends in time series within complex environments.

IV. CONCLUSION

In this letter, we propose a novel model, Kalman-SSM, which integrates the Kalman filter with state space modeling for time series analysis. By leveraging the strengths of the Kalman filter and mitigating its limitations, we enhance the representational capabilities of the state space model while reducing the model's training duration. Furthermore, Kalman-SSM is characterized by its lightweight computational footprint, with complexity comparable to that of linear models. This approach has achieved performance levels on par with SOTA models, demonstrating its efficacy and potential applicability across various time series modeling scenarios. In nonlinear scenarios, Kalman-SSM faces certain limitations. We intend to further explore and develop simplified approaches for nonlinear Kalman filter in our future research.

REFERENCES

[1] R. A. Angryk et al., “Multivariate time series dataset for space weather data analytics,” *Sci. Data*, vol. 7, no. 1, 2020, Art. no. 227.

[2] Z. A. Khan, T. Hussain, A. Ullah, S. Rho, M. Lee, and S. W. Baik, “Towards efficient electricity forecasting in residential and commercial buildings: A novel hybrid CNN with a LSTM-AE based framework,” *Sensors*, vol. 20, no. 5, 2020, Art. no. 1399.

[3] G. P. Zhang, “Time series forecasting using a hybrid ARIMA and neural network model,” *Neurocomputing*, vol. 50, pp. 159–175, 2003.

[4] H. Zhou et al., “Informer: Beyond efficient transformer for long sequence time-series forecasting,” in *Proc. AAAI Conf. Artif. Intell.*, 2021, pp. 11106–11115.

[5] N. Kitaev, L. Kaiser, and A. Levskaya, “Reformer: The efficient transformer,” in *Proc. Int. Conf. Learn. Representations*, Addis Ababa, Ethiopia, 2020.

[6] Y. Liu et al., “itransformer: Inverted transformers are effective for time series forecasting,” in *Proc. 12th Int. Conf. Learn. Representations*, 2024. [Online]. Available: https://openreview.net/forum?id=JePfAI8fah

[7] Y. Nie, N. H. Nguyen, P. Sinthong, and J. Kalagnanam, “A time series is worth 64 words: Long-term forecasting with transformers,” in *Proc. 11th Int. Conf. Learn. Representations*, 2023. [Online]. Available: https://openreview.net/forum?id=Jbdc0vTOcol

[8] A. Zeng, M. Chen, L. Zhang, and Q. Xu, “Are transformers effective for time series forecasting?,” in *Proc. AAAI Conf. Artif. Intell.*, 2023, pp. 11121–11128.

[9] K. Yi et al., “Frequency-domain MLPs are more effective learners in time series forecasting,” in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2024.

[10] A. Gu, K. Goel, and C. Ré, “Efficiently modeling long sequences with structured state spaces,” in *Proc. Int. Conf. Learn. Representations*, 2022.

[11] A. Gu et al., “Combining recurrent, convolutional, and continuous-time models with linear state space layers,” in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2021, pp. 572–585.

[12] R. Yamashita, M. Nishio, R. K. Do, and K. Togashi, “Convolutional neural networks: An overview and application in radiology,” *Insights Imag.*, vol. 9, pp. 611–629, 2018.

[13] R. E. Kalman, “A new approach to linear filtering and prediction problems,” *J. Basic Eng.*, vol. 82, no. 1, pp. 35–45, 1960.

[14] A. Ribeiro, I. D. Schizas, S. I. Roumeliotis, and G. Giannakis, “Kalman filtering in wireless sensor networks,” *IEEE Control Syst. Mag.*, vol. 30, no. 2, pp. 66–86, Apr. 2010.

[15] L. Hostetler and R. Andreas, “Nonlinear Kalman filtering techniques for terrain-aided navigation,” *IEEE Trans. Autom. Control*, vol. 28, no. 3, pp. 315–323, Mar. 1983.

[16] T. I. Fossen and T. Perez, “Kalman filtering for positioning and heading control of ships and offshore rigs,” *IEEE Control Syst. Mag.*, vol. 29, no. 6, pp. 32–46, Dec. 2009.

[17] Z. Qiu, S. Wang, P. Hu, and L. Guo, “Outlier-robust extended Kalman filtering for bioinspired integrated navigation system,” *IEEE Trans. Automat. Sci. Eng.*, early access, Oct. 05, 2023, doi: 10.1109/TASE.2023.3319508.

[18] Z. Qiu and L. Guo, “Improved cubature Kalman filter for spacecraft attitude estimation,” *IEEE Trans. Instrum. Meas.*, vol. 70, 2020, Art. no. 9504213.

[19] W. Yang, S. Li, Z. Li, and X. Luo, “Highly accurate manipulator calibration via extended Kalman filter-incorporated residual neural network,” *IEEE Trans. Ind. Informat.*, vol. 19, no. 11, pp. 10831–10841, Nov. 2023.

[20] W. Yang, S. Li, and X. Luo, “Data driven vibration control: A review,” in *Proc. IEEE/CAA J. Automatica Sinica*, pp. 1898–1917, 2024.

[21] Y. Yuan, X. Luo, M. Shang, and Z. Wang, “A Kalman-filter-incorporated latent factor analysis model for temporally dynamic sparse data,” *IEEE Trans. Cybern.*, vol. 53, no. 9, pp. 5788–5801, Sep. 2023.

[22] Z. Li, S. Li, and X. Luo, “Efficient industrial robot calibration via a novel unscented Kalman filter-incorporated variable step-size Levenberg–Marquardt algorithm,” *IEEE Trans. Instrum. Meas.*, vol. 72, 2023, Art. no. 2510012.

[23] H. Wu, J. Xu, J. Wang, and M. Long, “Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting,” in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2021, pp. 22419–22430.

[24] Z. Xu, A. Zeng, and Q. Xu, “Fits: Modeling time series with $10k$ parameters,” in *Proc. 12th Int. Conf. Learn. Representations*, 2024. [Online]. Available: https://openreview.net/forum?id=bWcnvZ3qMb

[25] D. M. Allen, “Mean square error of prediction as a criterion for selecting variables,” “*Technometrics*, vol. 13, pp. 469–475, 1971.

[26] C. J. Willmott and K. Matsuura, “Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance,” *Climate Res.*, vol. 30, pp. 79–82, 2005.

[27] A. Paszke et al., “PyTorch: An imperative style, high-performance deep learning library,” in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2019, pp. 8026–8037.