Moving-update Kalman Algorithm in Low-cost Node-Red IoT Network for Estimating Flood Water Level

Quang Dung NGUYEN[1], *Hoang Trung LE*[1], *Hoang Thien LE*[1], *Viet Hung TRAN*[2,*]

[1]Faculty of Electrical and Electronics Engineering, Ton Duc Thang University, Ho Chi Minh City, Vietnam
[2]Modeling Evolutionary Algorithms Simulation and Artificial Intelligence, Faculty of Electrical and Electronics Engineering, Ton Duc Thang University, Ho Chi Minh City, Vietnam.

*Corresponding Author: Viet Hung TRAN (email: tranviethung@tdtu.edu.vn)
(Received: 27-Jan-2022; accepted: 15-May-2022; published: 30-Sep-2022)
DOI: http://dx.doi.org/10.55579/jaec.202263.367

Abstract. *Flooding is one of the most common natural disasters in Vietnam. Although a hydrological monitoring system has been developed in Vietnam, the adoption of a Flood Warning and Monitoring System (FWMS) is still limited. A practical issue is that the river water levels is rarely flat, but undulating with flood water ripples, which makes the measurement inaccurate. In this paper, we will design a recursive Kalman estimation for fluctuating flood water level in the Node-Red IoT network. Indeed, the low complexity of the popular Kalman filter algorithm is very suitable for a low-cost IoT system like Node-Red. In our experiments, the accuracy of our Kalman algorithm is far superior to the standard Moving Average (MA) algorithm. To our knowledge, this is the first time that the Kalman filter has been used in a practical Node-Red IoT experiment. We will show that our novel Moving-update Kalman algorithm, which combines MA and Kalman methods, can track data recursively without prior knowledge of noise's variance. Our novel algorithm is of linear complexity and, hence, fast enough for low cost IoT and FWMS systems in developing countries like Vietnam. We also included the industrial Message Queuing Telemetry Transport (MQTT) protocol in IoT network in our Node-Red system, which means our designed Node-Red proposal is capable of transferring data to any FWMS network via internet.*

Keywords

Kalman filter, flood water level, Node-Red, IoT, MQTT.

1. Introduction

Vietnam is a coastal country, it suffers from many storms and floods every year [1]. Natural disasters and floods can strike at any time, adversely affect society, particularly those who live near the river [2]. As a result, a warning system of flood calamity is required. The warning system's goal is to reduce the number of victims and material harm to society while also providing data to authorities. Therefore, the application of Internet-of-Things (IoT) to water level measurement in lake and river is very important and necessary.

However, a difficulty of measuring river and lake water levels in IoT systems is fluctuating

noise because the water level is always undulating with flood water ripples. There are many methods for this problem in literature, but they are unsuitable for low-cost IoT system implementation (c.f. section 1.1. for details). Furthermore, the flooding places are mostly constrained with very low bandwidth communications. Hence, this paper will propose a solution for estimating flood water levels via our novel Moving-update Kalman filter (MKF) algorithm. Our MKF algorithm combines the traditional Kalman algorithm with moving-update estimate of noise's variance. Owing to our recursive design, this MKF method is suitable for a low-cost industrial IoT networks like Node-Red systems.

Fig. 1: A photo of central Vietnam flood, taken by us in August 2020.

1.1. Literature review

Many studies on flood warning systems have focussed on the estimation of water levels in a wide area in recent years. There are three main approaches for building estimation models [3]: traditional statistical methods, probability methods and machine learning (ML) methods [4, 5].

In statistical approaches, a Flood Warning and Monitoring System (FWMS) model based on Moving Average (MA) algorithm was proposed [6]. Their model used an average of recent real data to forecast the water level for the next periods. It is well suited for immediate prediction because of its recursive computation process. The Exponentially Weighted Moving Average (EWMA) model is proposed in [7]. Their idea is that different weighting factors of flood data in different historical periods have distinct effects on the prediction process. The EWMA model's prediction outcomes are smoother and more accurate than the MA model's. In FWMS, the Autoregressive Integrated Moving Average (ARIMA) model is widely used [8, 9, 10]. The data difference was used to filter the non-stationary components in the original sequence and yield better prediction results. Although the traditional statistical-based models perform better in terms of interpretability and low computational cost, it is constrained by its limited feature extraction and it cannot handle complex prediction problems with multiple constraints.

Hence, other studies have applied ML approaches or multi-mode fusion [11] to FWMS since the rise of ML era. Based on the decision tree method, the system in [12] can detect water level and monitor floods with possible humanitarian consequences before they occur. In [13], a prediction model of MIMO artificial neural network was applied to FWMS. A two-year data collection and a particular treatment of input data were used to obtain accurate forecasts, allowing a balance to be found between the spatial and temporal resolution of rainfall information and the model complexity. In [14], a convolutional neural network (CNN) was employed to predict fluvial flood inundation quickly. To estimate water depth, the CNN model was trained with outputs from a 2D hydraulic model.

Nonetheless, in practical FWMS, the prediction model is all limited with the storage capacity and processing power of low-cost IoT devices. Many ML models demand a large amount of historical data for training (storage problem), as well as a high temporal complexity during the training process (calculation problem) to attain higher prediction accuracy. As a result, although the ML model has a higher prediction accuracy, it cannot match the requirements of the lightweight model for the low-cost IoT computing node.

For low-cost scenarios of FWMS, the probability methods (mostly in Gaussian form) yield a good trade-off between the statistics and ML approaches. For example, a coupled hydraulic and Gaussian Kalman filter model was developed in [15] for a real-time flood forecast correction in the Yangtze river. This model is one of effective approaches for reducing errors in real-time flood

data under constraints of model structure, input data and calibrated parameters. In [16], the Extended Kalman Filter (EKF) technique was used to predict and estimate flood water level. Although the EKF can only solve the nonlinear problems of flood level via local Gaussian approximation, it is still one of good predictors of flood water level.

In summary, the above researches have studied both simple and complex models for FWMS. The advantages and disadvantages of the aforementioned strategies are summarized in Table 1. Note that, there are also Extended Kalman filters in [17, 18, 19] for tracking flood water levels, but their Taylor approximation is not fast enough for our low-cost IoT system in this paper because it requires a lot of memory and computation speed. Therefore, we has designed the Moving-update Kalman algorithm in order to balance these problems above.

1.2. Our contribution

Our paper's aim is to design a recursive Kalman algorithm and provide an IoT solution for estimating flood water levels in a low-cost Node-Red IoT network. To our knowledge, this is the first time that a Kalman filter was designed via Node-Red programming for a low-cost IoT system. Indeed, we only found one failed attempt implementing Kalman via the Node-RED programming in [28], and it said that their Kalman code didn't work. We designed the Kalman block for Node-Red following actual flow of computations.

Our Kalman method will be compared with the traditional Moving Average method (MA) in this paper. Our simulations and real-time experiments will show that estimating water level via the recursive Kalman algorithm considerably enhance the accuracy, for a relatively modest hardware cost. We also combine MA and Kalman methods so that our novel Moving-update Kalman algorithm can track data recursively without prior knowledge of noise's variance.

Also, in Vietnam, most of the places affected by natural disasters are in remote places with poor bandwidth and network access. Rapid upload of monitoring data and analysis of predictive results cannot be guaranteed, affecting decision-making time. Hence, for a real-time IoT system of FWMS, we combined industrial MQTT communication protocol [29] with recursive Kalman algorithm in our experiments of real-time processing, which is capable of solving the bandwidth bottleneck of FWMS's cloud communication systems.

1.3. Organization of paper

The following is the organization of the paper. In section 2. , the system model and methodology are presented. Section 3. introduces our Node-Red Programming. The simulation results are described in section 4. . Section 5. describes our practical experiments. Finally, section 6. gives some conclusions.

2. System model and methodology

Many FWMS solutions are now built on traditional IoT architecture to develop a remote monitoring system for FWMS [30], owing to the rise of IoT and the integration of various sensor devices and wireless technology.

In this paper, we use the industrial MQTT protocol to solve the IoT network bottleneck. With the ability to transmit data continuously in a low bandwidth environment, the MQTT data can be transmitted in real-time [31]. The data at each measurement point is processed by the our recursive Kalman filter algorithm. Finally, the data is transmitted in real-time to the server via the industrial MQTT protocol, as shown in Fig. 2.

A sensor network, which is made up of several types of sensors, can be used for real-time monitoring water levels, precipitation, flow rates, etc. The sensor network of low-cost ESP32 devices transmits data to cloud server, which, in turn, runs the Kalman Filter algorithm after receiving the data. The cloud stores the data, communicates with the client, and displays information

Algorithms	Advantages	Disadvantages
MA [20], EWMA [21], ARIMA [22]	well suited for immediate prediction because of its easy computation process	constrained by the limited feature extraction and its accuracy is low for complex scenarios.
ML [23]	accurately predict the trend of the observed object in advance of a long period of time	demanding a large amount of data and unsuitable for a low-cost IoT network
Kalman filters [24, 25, 26, 27]	good trade-off between speed and accuracy, suitable for real-time IoT network	very few researches of Kalman filters for a practical low-cost IoT network

Tab. 1: Pros and cons of the methods in section 1.1.

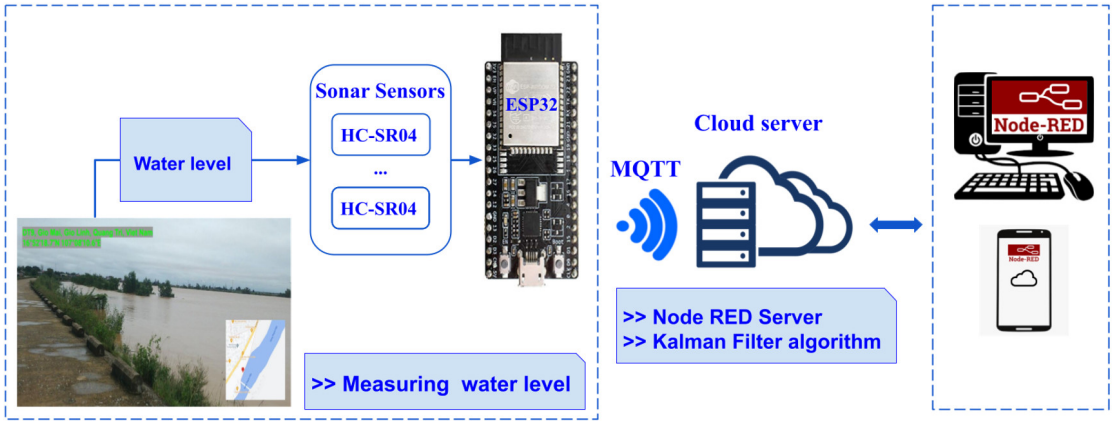

Fig. 2: IoT real-time data acquisition using industrial MQTT protocol. The ESP32 is a cheap wireless device in the market [32].

to the user, such as the current flood status and river water level.

2.1. Node-Red IoT platform

Node-Red is a programming tool for wiring together hardware devices, Application Programming Interface (APIs) and online services via drag-and-drop interface [33]. Node-Red is an open-source JavaScript environment based on Node.js. It was created by IBM experts and is best suited for IoT system development [34, 35]. It's a virtual programming environment that connects hardware and software to build "data streams" from the sensor to the cloud. It can be used to write data processes, making data processing easier. It can be used to construct data processing logic and quickly transport processed data to higher-level systems (SQL server, enterprise management system, central data collector, cloud-based service, etc.) [36, 37, 38].

Node-Red is made up of three fundamental components: a node panel, a flow panel and a Dashboard panel. Node-Red is a collection of nodes that perform various tasks (MQTT broker, Raspberry Pi, TCP, etc.). It's ideal for prototyping because of its versatility and ability to create applications in a short amount of time [39]. In this paper, we will use the Node-Red platform to build a IoT system for monitoring water level, as shown in Fig. 3.

2.2. Kalman filter algorithm

The popular Kalman filter (KF) algorithm differs from the conventional method of timing prediction [40]. It can estimate the system's state via a set of incomplete observations (i.e. miss-

Fig. 3: Our Node-Red Dashboard for monitoring water level via recursive Kalman algorithm.

ing time points in time-series data) or noisy observations (measurement error). The KF is a fast recursive filter model [41]. It takes up small memory and only needs to retain data for system's state at a time, rather than a long period of time. The actual measured data are used to correct the estimate results [42].

For a quick review, the one-dimensional Kalman filter model is defined as follows:

$$\begin{aligned} x(t) &= x(t-1) + n_v(t), \text{ with } n_v(t) \sim \mathcal{N}(0, \sigma_v^2), \\ z(t) &= x(t) + n_w(t), \text{ with } n_w(t) \sim \mathcal{N}(0, \sigma_w^2), \end{aligned} \tag{1}$$

in which $n_v(t)$ and $n_w(t)$ are additive white Gaussian noise (AWGN). The $x(t)$ is the actual water level over time t, $z(t)$ is the sensor's measurement of $x(t)$, water noise variance is σ_v^2 and sensor noise variance is σ_w^2 in this paper.

We assume that the water level $x(t)$ is mostly stable, only affected by noise caused by flood water ripples, as shown in Fig. 5. The Kalman estimate $\hat{x}(t)$ of the water level $x(t)$ can be calculated recursively, as follows [43]:

$$\hat{x}(t) = \hat{x}(t-1) + K(t)\left[z(t) - \hat{x}(t-1)\right], \tag{2}$$

$$\frac{1}{\hat{\sigma}_x^2(t)} = \frac{1}{\hat{\sigma}_x^2(t-1) + \sigma_v^2} + \frac{1}{\sigma_w^2}, \tag{3}$$

$$K(t) \triangleq \frac{\hat{\sigma}_x^2(t)}{\sigma_w^2}, \tag{4}$$

in which $\hat{\sigma}_x^2(t)$ is the estimate's variance. Note that, from Eq. (3), we can see that the estimate variance $\hat{\sigma}_x^2(t)$ is always bounded, as follows:

$$\frac{1}{\frac{1}{\sigma_v^2} + \frac{1}{\sigma_w^2}} < \hat{\sigma}_x^2(t) < \sigma_w^2, \quad \forall t. \tag{5}$$

Also, substituting $\hat{\sigma}_x(t) = \hat{\sigma}_x(t-1) = \tilde{\sigma}_x$ at convergence to Eq. (3) yields [44]:

$$\tilde{\sigma}_x^2 \triangleq \lim_{t\to\infty} \hat{\sigma}_x(t) = \sigma_v^2 \frac{\sqrt{1 + 4\frac{\sigma_w^2}{\sigma_v^2}} - 1}{2}. \tag{6}$$

The above equations are illustrated in Fig. 4 and Fig. 5.

2.3. Moving average method

The well-known moving average (MA) method, also known as a rolling average or running average, is a statistical computation that analyzes data points by creating a series of averages of data subsets. Given a series of values and a fixed subset size, the first element of the moving average is obtained by taking the average of the initial fixed subset of the number series. The subset is then "shifted forward", which means the first number in the series is excluded and includes the next value in the subset.

In this paper, a moving average is the mean of the previous k data-points, as shown in Fig. 6, i.e.:

$$\begin{aligned} \tilde{z}_k &\triangleq \frac{z_{n-k+1} + z_{n-k+2} + \ldots + z_n}{k} \\ &= \frac{\sum_{i=n-k+1}^{n} z_i}{k} \end{aligned} \tag{7}$$

in which $\tilde{z}_k$ is moving average of the previous k data-points and z_i is the measured data at time-point i.

2.4. Computational complexity

From Eq. (2-4) and Eq. (7), we can see that both one-dimensional Kalman algorithm and Moving Average method has the same linear computational complexity $O(n)$, in which n is number of time-points. This low complexity is totally suitable to low-cost IoT networks.

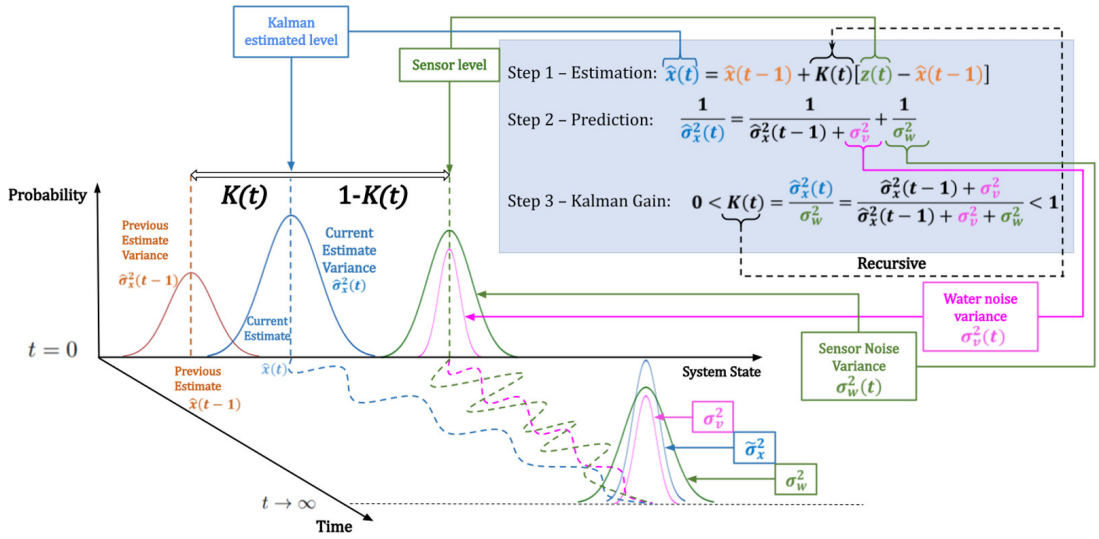

Fig. 4: Illustration of one-dimensional Kalman filter algorithm, as explained in section 2.2.

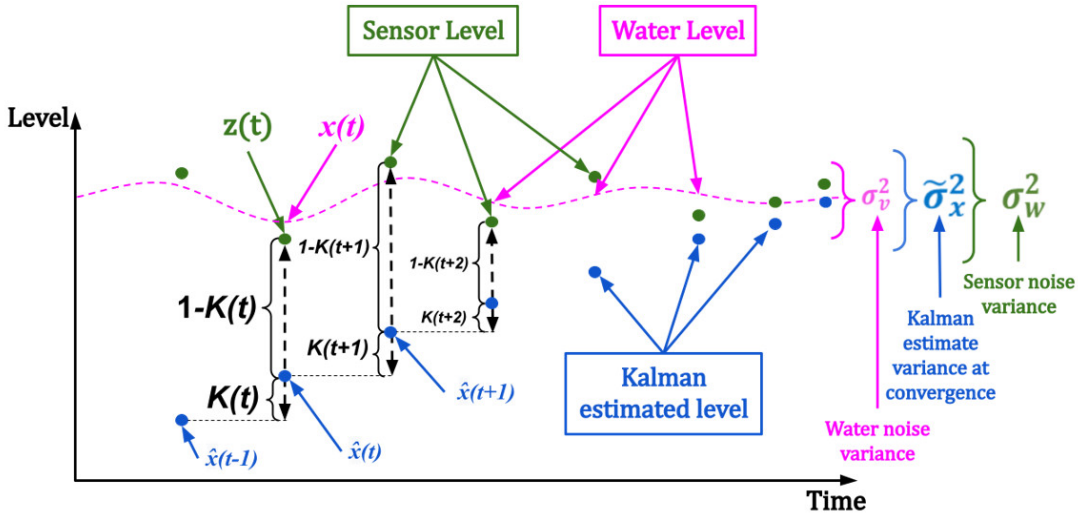

Fig. 5: Illustration of one-dimensional Kalman filter estimate at convergence, as explained in Fig. 4.

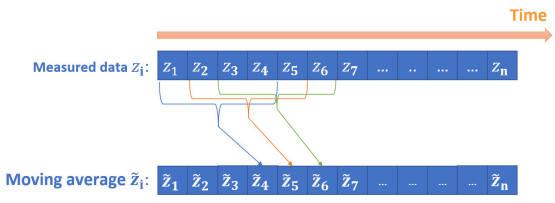

Fig. 6: Simple moving average method, as shown in Eq. (7) with $k = 4$.

3. Node-Red Programming

In this section, we will use the Node-Red platform to design the one-dimensional Kalman filter in a low-cost IoT network, as shown in Fig. 7.

Fig. 7: Recursive Kalman algorithm for IoT Network in Node-Red, as illustrated in Fig. 5.

3.1. Recursive Kalman algorithm

Fig. 8 demonstrates the flowchart of our recursive Kalman algorithm in Node-Red. Measurement data is the input of "Kalman 1D" block. For each coming data, the Kalman estimate is updated again until convergence occurs.

Our recursive Kalman algorithm is given in "Kalman 1D" block of Fig. 7. We designed "Kalman 1D" block in Fig. 9 based on the Eq. (2-4) and Fig. 4.

3.2. Moving average

In Fig. 25, we created block "Measured Data" for our simulation in Node-Red. This block has the function of simulating noisy measured data. The data is passed through block "Moving Average" and the results are displayed on the dashboard.

3.3. Moving-update Kalman filter (MKF)

In Fig. 26, we have designed the "Moving-update Kalman" block based on Eq. 8. Results are automatically updated recursively. This block combines the traditional Kalman algorithm and the Moving Average. Experimentally, the results are better and have the advantage of good data tracking without knowing the noise's variance in advance.

In our novel algorithm, we apply MA method in Eq. (7) and estimate the unknown noise's variance σ_w over time in Kalman, as follows:

$$\tilde{\sigma}_w(k) \triangleq \sqrt{\frac{\sum_{i=n-k+1}^{n}(z_i - \tilde{z}_k)^2}{k}}. \qquad (8)$$

Based on the measured data, we estimate the unknown noise's variance $\tilde{\sigma}_w$ via Eq. (8) and update the value $\tilde{\sigma}_w$ of Kalman formula in Eq. (3-4). Because the recursive measurement estimate in Eq. 8 is consistent and leads to the sensor noise's variance asymptotically, our novel Moving-update Kalman algorithm can track data recursively without knowing noise's variance in advance. Our Moving-update Kalman method is more suitable in practice because the noise's variance is often unknown.

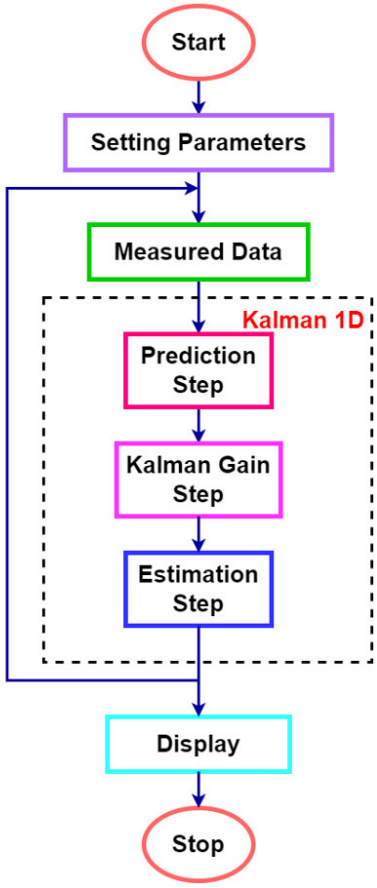

Fig. 8: Flowchart of recursive Kalman algorithm in Fig. 7.

Fig. 9: Inside of "Kalman 1D" block in Fig. 7. Notation σ^* is equivalent to notation $\hat{\sigma}$ in the paper.

4. Simulation results

In our Node-Red simulation, the input data and necessary parameters are set up in the "Setting Parameters" block in Fig. 23. We generate random values that are similar to actual flood levels, which simulate undulating flood levels caused by water ripples.

Fig. 23 shows our setting parameters with true value of water level, the initial estimate $\hat{x}(0)$, initial estimate variance $\hat{\sigma}_x^2(t)$, water noise variance σ_v^2 and sensor noise variance σ_w^2. Our Kalman parameters in Fig. 23 are set up as follows: $\hat{x}(0) = 10$, $\sigma_x(0) = 0.1$, $\sigma_v = 0.01$ and $\sigma_w = 0.36$, as shown in Node-Red dashboard in Fig. 3 and Fig. 24. In this section, all x-axis is of time in second, while y-axis is of centimeters (cm).

4.1. Recursive Kalman estimates

In our simulation, we use the recursive Kalman algorithm to estimate the true value of water level. Fig. 10 depicts the algorithm's convergence process. The "Measured Data" line is the random observation with 200 samples in 10 minutes. The "Kalman Filter" line represents the recursive Kalman estimates. The "True Value" line is the true value of water level. At convergence, we can see that the Kalman estimate is very close to the true value.

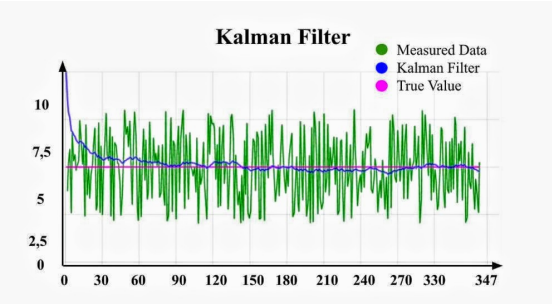

Fig. 10: The convergence simulation of recursive Kalman algorithm in Fig. 5

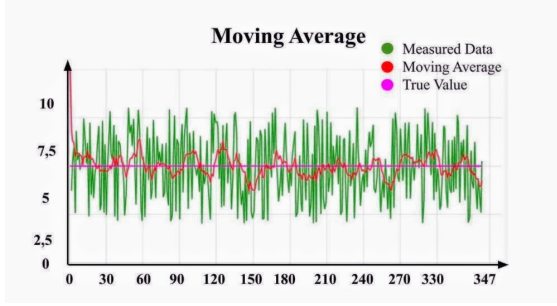

Fig. 11: The convergence simulation of Moving Average method in Fig. 10.

4.2. Moving Average estimates

Our simulation in Fig. 11 shows that the Moving Average estimates have lower variance than that of measured data, but much higher than that of Kalman estimates in Fig. 10.

4.3. Error comparison

In Fig. 12-13, we plot a comparative chart of error in order to compare the accuracy of the recursive Kalman algorithm and that of Moving Average method. We can see that the error's variance of Kalman estimate is much lower than that of Moving Average estimates.

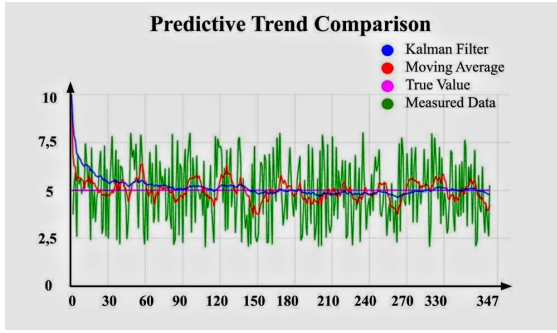

Fig. 12: Comparison of Kalman and Moving Average estimates in Fig. 10-11.

4.4. Moving-update Kalman algorithm:

The result of our novel MKF algorithm is shown in Fig. 15. This result is similar to the Kalman method with known noise's variance in Fig. 10.

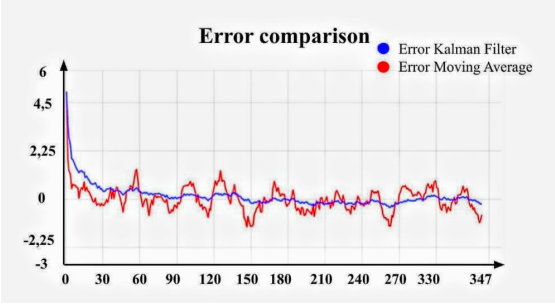

Fig. 13: Error comparison between Kalman and Moving Average estimates. The error value is the difference between estimates and true value in Fig. 12.

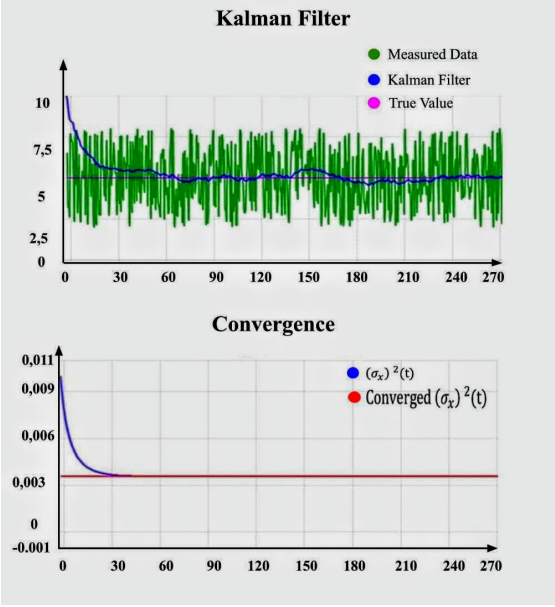

Fig. 14: Converged $\bar{\sigma}_x^2$ in equation 6.

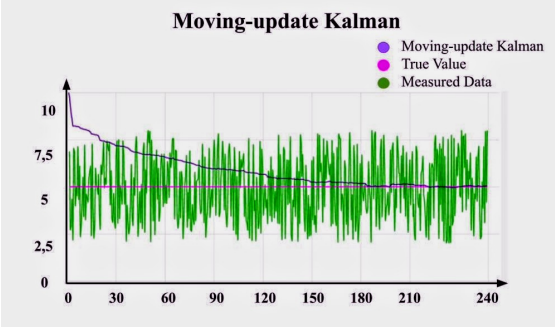

Fig. 15: Moving-update Kalman algorithm.

4.5. Computational complexity

Because all algorithms in this paper are recursive for one-dimensional data, their computational complexity is linear with O(n), where n is the number of data. Hence our Moving-update Kalman algorithm is fast enough for low-cost ESP32 devices and Node-Red programming in IoT systems.

A comparison of computational complexity and requirement between our proposed method and standard Kalman filters is shown in Table 2.

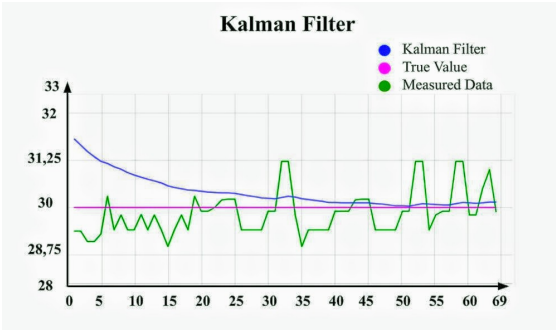

Fig. 16: Recursive Kalman estimates for our experiment in Fig. 19. The y-axis is of centimeters.

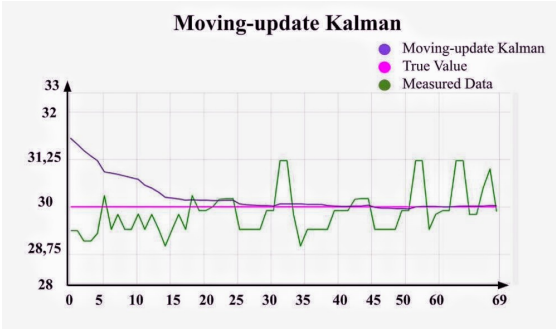

Fig. 17: Moving-update Kalman estimates for our experiment in Fig. 19. The y-axis is of centimeters.

5. Experiment results

Following traditional IoT model in Fig. 2, we built a simple experiment to measure the water level in Fig. 19. We dropped pebbles into the water in order to simulate the ripple process of flood water waves. In our system, the ESP32

Algorithms	Complexity	Estimation	Suitable scenarios
Kalman filter	$O(n)$	Optimal, linear	Known noise's variance
Extended Kalman	$O(K^2n)$	Sub-optimal, non-linear	Known noise's variance
Moving-update Kalman	$O(n)$	Asymptotically optimal, linear	Unknown noise's variance

Tab. 2: A comparison of computational complexity and suitable scenarios [45], in which n is the number of data and K is the number of time-points for gradient computation in Taylor's approximation.

Time(s)	Value	Time(s)	Value	Time(s)	Value	Time(s)	Value	Time(s)	Value
0	29.380	10	29.820	20	29.8	30	29.9	40	30.2
1	29.380	11	29.410	21	29.9	31	29.9	41	30.3
2	29.410	12	29.380	22	29.9	32	29.9	42	30.3
3	29.000	13	29.410	23	29.9	33	29.9	43	30.31
4	29.410	14	28.970	24	29.93	34	29.9	44	30.31
5	29.410	15	29.410	25	30	35	30.2	45	30.33
6	29.410	16	29.410	26	29.9	36	30.23	46	30.33
7	29.410	17	29.410	27	29.9	37	30.22	47	30.3
8	29.410	18	28.2	28	29.9	38	30.22	48	30.33
9	29.410	19	29.8	29	29.9	39	30.2	49	30.33

Fig. 18: Measured data from sonar sensor of our experiment in Fig. 19.

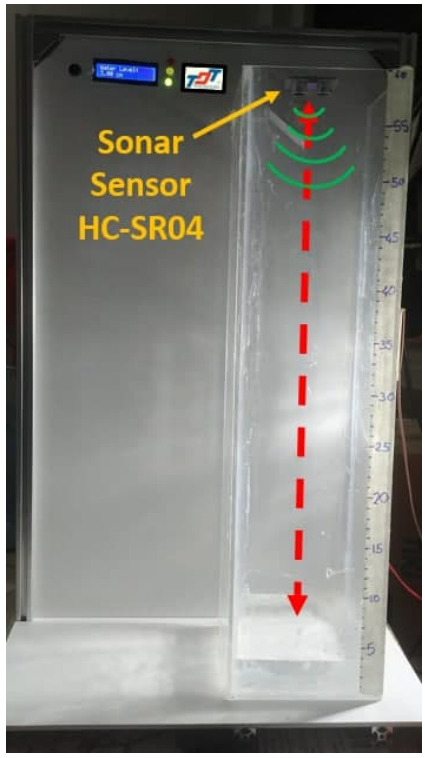

Fig. 19: Our water level measurement experiment, which follows traditional IoT model in Fig. 2.

module is used to collect water level data from a sonar sensor. The data obtained is shown in Fig. 18. We set the actual water level to $30cm$ and intentionally created noise around the true value, as shown in Fig. 18. The initial prediction value of the algorithm is $31.5cm$. After a while, the results converge to the true value $30cm$, as shown in Figs. 16-17.

Via recursive and moving-update Kalman algorithms, our Node-Red system can eliminate the noise caused by water ripples and predict the water level with high accuracy, as shown in Figs. 16-17. Indeed, in our MKF algorithm, we do not know the measurement noise variance in advance. Furthermore, our recursive estimations are updated continuously and asymptotically converged to the true value. For practical illustration, we have applied our MKF to a low-cost industrial Node-Red IoT system as a demo for measuring water levels and its potential application in floods. Therefore, experimentally we can see that our MKF algorithm works well even on low-cost devices.

6. Conclusions

This study has proposed a recursive Kalman algorithm in Node-RED platform for estimating flood water levels. By using industrial MQTT protocol in Node-RED program, our simulations and experiments have demonstrated that the recursive Kalman algorithm is fast and suitable for a low-cost IoT and Flood Warning and Monitoring System (FWMS). We also showed that our novel Kalman recursion is far superior to the traditional Moving Average method, even with similar computational complexity in a low-cost IoT system.

In future work, we can feasibly estimate more flood-related information (such as flow rate, temperature, humidity, etc.) via our Kalman algorithm in Node-Red. Our recursive Kalman algorithm can be embedded feasibly in practical systems.

Appendix

Node-Red Programming

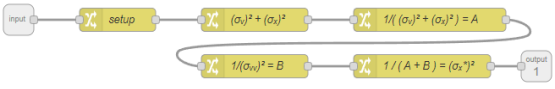

Fig. 20: Inside of "Prediction: $(\sigma_x^*)^2$" block in Fig. 9.

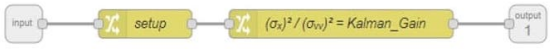

Fig. 21: Inside of "Kalman Gain" block in Fig. 9.

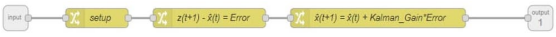

Fig. 22: Inside of "Estimation: $\hat{x}(t+1)$" block in Fig. 9.

We designed "Kalman 1D" block in Fig. 9 with three steps:

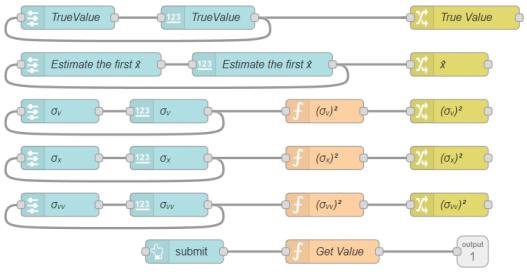

Fig. 23: Inside of "Setting Parameters" block in Fig. 7.

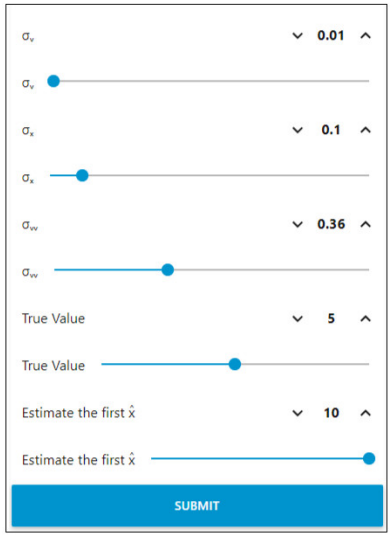

Fig. 24: The Node-Red dashboard frame of "Setting Parameters" block in Fig. 23.

1. Block "Prediction: $(\sigma_x^*)^2$" in Fig. 20 computes estimate variance of the system to obtain the next estimate $\hat{\sigma}_x(t+1)^2$ for the next time step $t+1$, based on Eq. 3.

2. Block "Kalman Gain" in Fig. 21 performs the function of calculating the correction factor to estimate the next state of the system, as shown in Fig. 21 and Eq. 4.

3. Block "Estimation: $\hat{x}(t+1)$" in Fig. 22 estimates the water level $\hat{x}(t+1)$ at the time $t+1$ based on previously calculated estimate variance results, as shown in Eq. 2.

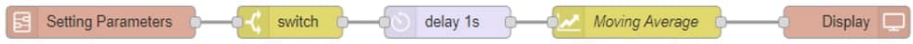

Fig. 25: Moving Average algorithm for IoT Network in Node-Red [46].

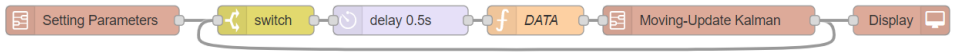

Fig. 26: Moving-update Kalman algorithm for IoT Network in Node-Red.

References

[1] Staff-reporters/VnExpress, Central Vietnam: a month in tragedies. https://e.vnexpress.net/news/news/central-vietnam-a-month-in-tragedies-4185767.html, published on November 3, 2020.

[2] Anh-Kiet, L.N., Historic flooding in central Vietnam: 102 dead and 26 missing. http://hanoitimes.vn/historic-flooding-in-central-vietnam-102-dead-and-26-missing-314576.html, published on October 20, 2020.

[3] Tran, V.H. (2014). *Variational Bayes inference in digital receivers*. Ph.D. thesis, Trinity College Dublin, Ireland.

[4] Xiao-ling Wu, C.h.W., Xi-Chen, X.h.X., & Zhou, Q. (2008). Kalman Filtering Correction in Real-Time Forecasting with Hydrodynamic Model. *Journal of Hydrodynamics*, *20*, 391—397.

[5] Sankaranarayanan, S., Malavika Prabhakar, P.J., Sreesta Satish, A.R., & Krishnan, A. (2019). Flood prediction based on weather parameters using deep learning. *Journal of Water and Climate Change*, *11*, 1766—1783.

[6] AlissonSilva Souza, F.L.d.S.S., André Márciode Lima Curvello & Silva, H.J. (2017). A flood warning system to critical region. *Procedia Computer Science*, *109*, 1104–1109.

[7] Bae, I. & Ji, U. (2019). Outlier Detection and Smoothing Process for Water Level Data Measured by Ultrasonic Sensor in Stream Flows. *Water*, *11*(5).

[8] Ab Razak, N., Aris, A., Ramli, M., Looi, L., & Juahir, H. (2018). Temporal flood incidence forecasting for Segamat River (Malaysia) using autoregressive integrated moving average modelling. *Journal of Flood Risk Management*, *11*(S2), S794–S804.

[9] Wong, W.M., Subramaniam, S.K., Feroz, F.S., Subramaniam, I.D., & Rose, L.A.F. (2020). Flood Prediction using ARIMA Model in Sungai Melaka, Malaysia. *International Journal of Advanced Trends in Computer Science and Engineering*, *9*(4), 5287–5295.

[10] Wong, W.M., Lee, M.Y., Azman, A.S., & Rose, L.A.F. (2021). Development of Short-term Flood Forecast Using ARIMA. *International Journal of Mathematical Models and Methods in Applied Sciences*, *15*, 68–75.

[11] Noymanee, J. & Theeramunkong, T. (2019). Flood Forecasting with Machine Learning Technique on Hydrological Modeling. *Procedia Computer Science*, *156*, 377–386, 8th International Young Scientists Conference on Computational Science, YSC2019, 24-28 June 2019, Heraklion, Greece.

[12] Abdirahman Osman Hashi, M.A.E., Abdullahi Ahmed Abdirahman & Hashim,

S.Z.M. (2021). A Real Time Flood Detection System Based on Machine Learning Algorithms. *International Conference of Reliable Information and Communication Technology*, *72*.

[13] Campolo, M., Soldati, A., & Andreussi, P. (2003). Artificial neural network approach to flood forecasting in the River Arno. *Hydrological Sciences Journal*, *48*(3), 381–398.

[14] Kabir, S., Patidar, S., Xia, X., Liang, Q., Neal, J., & Pender, G. (2020). A deep convolutional neural network model for rapid prediction of fluvial flood inundation. *Journal of Hydrology*, *590*, 125481.

[15] Wu, X.L., Xiang, X.H., Wang, C.H., Chen, X., Xu, C.Y., & Yu, Z. (2013). Coupled Hydraulic and Kalman Filter Model for Real-Time Correction of Flood Forecast in the Three Gorges Interzone of Yangtze River, China. *Journal of Hydrologic Engineering*, *18*(11), 1416–1425.

[16] Adnan, R., Ruslan, F.A., Samad, A.M., & Md Zain, Z. (2012). Extended Kalman Filter (EKF) prediction of flood water level. In *2012 IEEE Control and System Graduate Research Colloquium*, 171–174.

[17] Adnan, R., Ruslan, F.A., Samad, A.M., & Md Zain, Z. (2012). Extended Kalman Filter (EKF) prediction of flood water level. In *2012 IEEE Control and System Graduate Research Colloquium*, 171–174.

[18] Adnan, R., Ruslan, F.A., Samad, A.M., & Zain, Z.M. (2013). New Artificial Neural Network and Extended Kalman Filter hybrid model of flood prediction system. In *2013 IEEE 9th International Colloquium on Signal Processing and its Applications*, 252–257.

[19] Rachmawati, V., Arif, D.K., & Adzkiya, D. (2018). Implementation of Kalman filter algorithm on models reduced using singular pertubation approximation method and its application to measurement of water level. *Journal of Physics: Conference Series*, *974*, 012018.

[20] Gongshen, L., Jianhua, L., & Shenghong, L. (2006). New multi-pattern matching algorithm. *Journal of Systems Engineering and Electronics*, *17*(2), 437–442.

[21] Crowder, S.V. & Hamilton, M.D. (1992). An EWMA for Monitoring a Process Standard Deviation. *Journal of Quality Technology*, *24*(1), 12–21.

[22] Nie, H., Liu, G., Liu, X., & Wang, Y. (2012). Hybrid of ARIMA and SVMs for Short-Term Load Forecasting. *Energy Procedia*, *16*, 1455–1460, 2012 International Conference on Future Energy, Environment, and Materials.

[23] Wuest, T., Weimer, D., Irgens, C., & Thoben, K.D. (2016). Machine learning in manufacturing: advantages, challenges, and applications. *Production & Manufacturing Research*, *4*(1), 23–45.

[24] Brown, S.D. (1986). The Kalman filter in analytical chemistry. *Analytica Chimica Acta*, *181*, 1–26.

[25] Sabat, N.K., Pati, U.C., Senapati, B.R., & Das, S.K. (2019). An IoT Concept for Region Based Human Detection Using PIR Sensors and FRED Cloud. In *2019 IEEE 1st International Conference on Energy, Systems and Information Processing (ICESIP)*, 1–4.

[26] Mapili, M.G.A., Rodriguez, K.A.D., & Sese, J.T. (2021). Smart Air Filtration System Using IoT and Kalman Filter Algorithm for Indoor Air Quality and Plant Monitoring. In *2021 IEEE 11th International Conference on System Engineering and Technology (ICSET)*, 309–314.

[27] Lai, X., Yang, T., Wang, Z., & Chen, P. (2019). IoT Implementation of Kalman Filter to Improve Accuracy of Air Quality Monitoring and Prediction. *Applied Sciences*, *9*(9).

[28] Node-RED (2019). Kalman filter not working. *https://discoursenoderedorg/t/kalman-filter-not-working*.

[29] Naik, N. (2017). Choice of effective messaging protocols for IoT systems: MQTT, CoAP, AMQP and HTTP. In *2017 IEEE International Systems Engineering Symposium (ISSE)*, 1–7.

[30] Sunkpho, J. & Oottamakorn., C. (2011). Real-time flood monitoring and warning system. *Songklanakarin Journal*, *33*, 227–235.

[31] Atmoko, R.A., Riantini, R., & Hasin, M.K. (2017). IoT real time data acquisition using MQTT protocol. *Journal of Physics Conference Series*.

[32] Babiuch, M., Foltýnek, P., & Smutný, P. (2019). Using the ESP32 Microcontroller for Data Processing. In *2019 20th International Carpathian Control Conference (ICCC)*, 1–6.

[33] Lekić, M. & Gardašević, G. (2018). IoT sensor integration to Node-RED platform. In *2018 17th International Symposium INFOTEH-JAHORINA (INFOTEH)*, 1–5.

[34] Chaczko, Z. & Braun, R. (2017). Learning data engineering: Creating IoT apps using the node-RED and the RPI technologies. In *2017 16th International Conference on Information Technology Based Higher Education and Training (ITHET)*, 1–8.

[35] Shinde, V.R., Tasgaonkar, P.P., & Garg, R.D. (2018). Environment Monitoring System through Internet of Things(IOT). In *2018 International Conference on Information , Communication, Engineering and Technology (ICICET)*, 1–4.

[36] Tabaa, M., Chouri, B., Saadaoui, S., & Alami, K. (2018). Industrial Communication based on Modbus and Node-RED. *Procedia Computer Science*, *130*, 583–588.

[37] Badii, C., Bellini, P., Cenni, D., Mitolo, N., Nesi, P., Pantaleo, G., & Soderi, M. (2020). Industry 4.0 Synoptics Controlled by IoT Applications in Node-RED. In *2020 International Conferences on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData) and IEEE Congress on Cybermatics (Cybermatics)*, 54–61.

[38] Rattanapoka, C., Chanthakit, S., Chimchai, A., & Sookkeaw, A. (2019). An MQTT-based IoT Cloud Platform with Flow Design by Node-RED. In *2019 Research, Invention, and Innovation Congress (RI2C)*, 1–6.

[39] Rajalakshmi, A. & Shahnasser, H. (2017). Internet of Things using Node-Red and alexa. In *2017 17th International Symposium on Communications and Information Technologies (ISCIT)*, 1–4.

[40] Fildes, R. (1991). Forecasting, Structural Time Series Models and the Kalman Filter: Bayesian Forecasting and Dynamic Models. *Journal of the Operational Research Society*, *42*(11), 1031–1033.

[41] Harvey, A.C. (2017). Forecasting, Structural Time Series Models and the Kalman Filter. *The Journal of the Operational Research Society*, *42*, 1031–1033.

[42] Welch, G.F. (2020). *Kalman Filter*. Cham: Springer International Publishing, 1–3.

[43] Wilson, R. & Finkel, L. (2009). A neural implementation of the Kalman filter. *Advances in neural information processing systems*, *22*, 2062–2070.

[44] Loh, P.S. (2019), A Simple Proof of the Quadratic Formula.

[45] Raitoharju, M. & Piché, R. (2019). On Computational Complexity Reduction Methods for Kalman Filter Extensions. *IEEE Aerospace and Electronic Systems Magazine*, *34*(10), 2–19.

[46] Nodered.org, Node RED contrib moving average. https://flows.nodered.org/node/node-red-contrib-moving-average, published on 2021.

About Authors

Quang Dung NGUYEN received his MSc degree from Tomsk Polytechnic University, Tomsk City, Russia in 2012. He is now a PhD Student at Faculty of Electrical and Electronics Engineering (FEEE), Ton Duc Thang University (TDTU), Vietnam. His research interests include applications of Kalman Filter in control systems; identication and synthesis in fractional, distributed parameter systems; non-linear system; PLC and SCADA system, industrial communication networks, image processing.

Hoang Trung LE is an undergraduate student in Automation and Control Engineering at Ton Duc Thang University, Vietnam. His research interest is signal processing and control theory. He is keen on pursuing a PhD degree and contributing to the development of his hometown in Central of Vietnam.

Hoang Thien LE is an undergraduate student in Automation and Control Engineering at Ton Duc Thang University, Vietnam. His research interest is Internet of Things.

Viet Hung TRAN received the B.Eng. degree from Hochiminh City University of Technology, Vietnam, in 2008, the MSc. degree from ENS Paris-Saclay, France, in 2009, and the Ph.D. degree from the Trinity College Dublin, Ireland, in 2014. From 2014 to 2016, he held a post-doctoral position with Telecom ParisTech, France. From 2017 to 2018, he was a Research Fellow at University of Surrey, U.K. He is currently a researcher at Ton Duc Thang University, Vietnam. His research interest is artificial intelligence and information theory. He was awarded the best mathematical paper prize at Irish Signals and Systems Conference, 2011.