Enhanced Faulty Node Detection with Interval Weighting Factor for Distributed Systems

Riesa Krisna Astuti Sakir, Sanjay Bhardwaj, and Dong-Seong Kim

Abstract: **This paper proposes an enhanced faulty node detection method using interval weighting factor, which monitors node behavior using pseudo-random Bose-Chaudhuri-Hocquenghem (BCH) code for distributed networked control systems. Master node collects the replacement of the cyclic redundancy check (CRC) codes by a single-bit BCH code of each slave node. However, BCH code can only obtain error position of the suspected faulty node without consideration of channel nodes. Hence, suspected error nodes are saved within the detected error interval and normalized using the weighting factor, which is carried out during sequential check, for interpretation. Fault judgement is carried out to adequately interpret the data, to guarantee detection accuracy of the detected error. The data is represented by statistical characteristic of the raw data and filtered data. This scheme can be applied to detect and prevent the severe damage by node failure. The simulation results prove the effectiveness of the interval faulty weighting factor, in obtaining raw data from the monitoring of the BCH code and filtered data, which are more accurate representation than the raw data. Moreover, the characteristics of the observed data were verified as the evaluation result of the suspected faulty node.**

Index Terms: **BCH, CRC, distributed networked systems, fault judgement, faulty node detection.**

I. INTRODUCTION

DISTRIBUTED networked control systems [1] have received significant attention with respect to their capabilities in field of communication networks, control and computer science. This system is based on distributed and parallel architecture, which is directly related to balance of the workload [2]. Moreover, critical impact in distributed systems is used to stabilize the system for redundancy and fault tolerance, [3]. Electric power systems [4], ship power systems [5], visual processing [6], and computer processors [7] are several applications that utilize the advantages of distributed systems.

In [8], [9], redundancy and fault tolerance also have critical impact on stabilizing systems based on a distributed system. Due to large variation in quality of components that are used in the construction of a distributed networked control system, there is a high probability that the component responsible for system's workload distribution will share the work unfairly. It is not necessary that all the system's components are identical in terms of quality. Instead, the performance ability is distributed to every node that is involved in the system. Therefore, when viewing the overall system, it is not easy to determine which part is performing well and which part is performing poorly.

A distributed networked control system is constructed using components with significantly varying qualities. There is high probability that components will show the biased workload. With workload distribution, channel quality of components are identical in respect to distributed networked control system, which node receives the performance capability instead of the biased workload. Therefore, significant amount of work is carried out by the system to determine and classify nodes as faulty or healthy. With conventional method, the system reliability can only be ensured using cyclic redundancy check (CRC). The data correctness can be identified using CRC code for the calculation of the message syndrome. However, the assumption related to CRC calculation is that the message is always correctly calculated by the node. Therefore, individual node performances should be evaluated using on data integrity checking.

Traditionally, CRC code is used to guarantee system reliability. Data correctness is identified by CRC code which is used to calculate the message syndrome in the receiver. However, there is an assumption that the node constantly calculates the message correctly when the CRC calculation is employed by the system. Therefore, the individual node performances should be evaluated using data integrity checking. The distributed manner of the node reliability effect is considered in this study. For example, the periodical occurrence of errors within an area was predicted. The authors in [10] found this constraint is reflected by a single bit, which represents the Bose-Chaudhuri-Hocquenghem (BCH) code fragment that is added to the CRC code. Thus, the findings of this study are applicable to industrial networks such as controller area networks (CAN).

In the field of industrial and military applications, distributed networked control systems employ Fieldbus systems. Fieldbus system is appropriate for distributed networked systems, to achieve real-time distributed control. Power distribution is an aspect of the system robustness that is improved by the use of industrial Fieldbus systems. In [11], a wireless Fieldbus based on MAC protocol is evaluated to achieve real-time transmission in the networked control systems. The wireless Fieldbus performed within desired time limits, to transmit three types of data: The periodic data, sporadic data, non real-time messages using transmission and bandwidth allocation scheme. The data transmission continuously in the node has to equipped with error detection to prevent the unexpected event, which can ruin

Manuscript received June 3, 2019; revised December 9, 2020; approved for publication by Ali Abedi, Division III Editor, December 28, 2020.

This research was supported by Kumoh National Institute of Technology (2019-104-131).

R. K. A. Sakir is with the Department of IT Convergence Engineering, Kumoh National Institute of Technology, Gumi, South Korea, email: riesakr@kumoh.ac.kr.

S. Bhardwaj and D.-S. Kim are with the ICT Convergence Research Center and Department of IT Convergence Engineering, Kumoh National Institute of Technology, Gumi, South Korea, email: sanjay.b1976@gmail.com, dskim@kumoh.ac.kr.

Digital Object Identifier: 10.23919/JCN.2021.000002

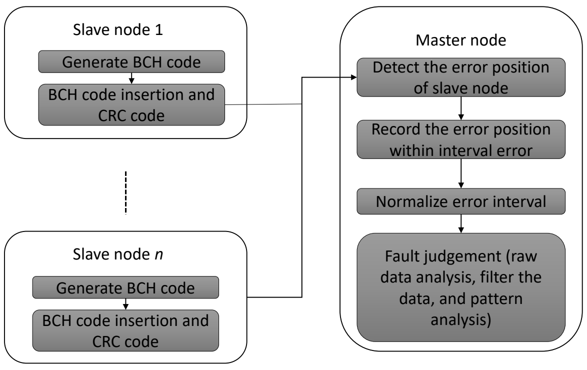

Slave node 1
Generate BCH code
BCH code insertion and CRC code
Slave node *n*
Generate BCH code
BCH code insertion and CRC code
Master node
Detect the error position of slave node
Record the error position within interval error
Normalize error interval
Fault judgement (raw data analysis, filter the data, and pattern analysis)

Fig. 1. System model of faulty node detection scheme.

the data received. In addition, the inserted BCH code is applied to observe the behavior of the nodes. To increase the detection accuracy, the BCH code is inserted into one-bit CRC code to maintain the overall faulty node and channel noise monitoring. From this observation, the incorporation of the detected time of the faulty node and the suspected faulty node channel was found to be the most critical factor to the application of interval weighting factor. This monitoring technique is composed of different sets of data interpretation referred to as fault judgements. It was assumed that system synchronicity, clock inconsistency, and page faults may occur regularly at a slave node of the proposed scheme. The earlier detection of the decrease in performance of an internal node can be carried out by observing the capability of the node to conduct sequential calculation within a given time. The calculation results, which consists of a single-bit BCH fragment at each node are collected and analyzed at the master node. Moreover, BCH code calculation at each slave node can be seen a test to detect which the node has failed.

The Fieldbus system is a popular technology that is applied to distributed networked systems. Therefore, this proposed system is adjusted to implement the Fieldbus system. The CAN bus [12], Modbus [13], Flexray [14] or industrial Ethernet can also be adjusted to implement this method. This is due to the capability of the algorithm to prevent the system function failure caused by broken nodes, thus maintaining the system performance.

This paper presents an extended version of the previous work [15] wherein the main problem and solution were discussed. The introduction of fault judgement with respect to the weighting factor of suspected faulty node is the main contribution of this paper. Weighting factor is used to declare the suspected node at the time of received error positions. Error position is detected by analyzing the BCH code that has been inserted into the CRC code in the master node. In addition, error position was analyzed by incorporating the external factor. The remainder of this paper is organized as follows. The related work is discussed in the Section II. Thereafter, fault node detection using the faulty weighting factor is presented in the Section III. The fault judgement process and its application is presented in the Section IV. Thereafter, simulation results of the proposed algorithm are presented in Section V. Finally, Section VI presents the conclusion.

II. RELATED WORKS

Faulty nodes are disadvantageous to distributed networked systems that consist of million nodes. Several related papers have been motivated to propose a faulty node detection, which can prevent severe damage caused by the faulty node using different methods and algorithms.

A fault node detection scheme for distributed systems was proposed in [15] using BCH and CRC code. Code combination results in increased performance with respect to the detection of faulty nodes. However, error position detector still has not guaranteed to interpret the faulty node without checking channel quality of nodes. Method based on Markov chain model was developed for collective monitoring of node behavior to detect faulty nodes [16] for wireless sensor network. Data in distributed networked control systems is transmitted from the slave nodes to the master node simultaneously. Thus, this method is not able to detect the faulty node from the channel quality view.

A weighting factor can also be applied to long short-term memory (LSTM) recurrent neural networks for faulty node detection systems [17]. The fault detection method was developed for machines in smart factories. The use of weighting factor can detect and obtain the faulty node more accurately. Data aggregation for transport has been used for the detection of faulty nodes in wireless sensor networks. The capability of the sink was utilized to detect the faulty machine with using Markov chain controller (MCC). Moreover, delays and wireless network traffic are considered to increase the detection rate. However, it did not provide the classification of suspected faulty node. Therefore, it will take much time for worker to recover the faulty machine.

The node behavior in any system field is monitored to realize real-time system. Most of the data generated in distributed system environment should be used to monitor node behavior. It will be an effective method with decrease the time required by the monitoring process. The BCH code is an algorithm that can be employed for collective monitor. In [18], the Chien search is exploited for modeling the power estimation, which employs higher error correcting capability, namely BCH codes. However, the error detection method did not consider the number of nodes in their system. Thus, it can not be applied into distributed systems, which have huge number of nodes.

To ensure the data is fault, an approach using weighted data-driven is an effective way for large-scale environments [19]. The fault detection and isolation (FDI) is identified and modeled to enable the approach, which can show the accurate result without loss of sensitivity. However, the approach does not consider for industrial application, which has several requirements and important issues to meet with fault detection method, particularly in the distributed networked control systems.

The authors in [20] directed significant research attention toward early identification of node failures in distributed systems, which consist of hundreds and thousands of nodes. However, identification only considers identification time capability to accelerate the maintaining process in distributed storage systems. In [21], faulty node diagnosis is employed using coordinator-based adaptive fault diagnosis algorithm as innovative technique in distributed computer systems. Both related works cannot interpret faulty node due to existing researches of predicting faulty node have attention in the early identification and diagnosed node regularly execution. Those systems did not consider channel quality in interval data that represented by slave node.

III. Faulty Node Detection Scheme

The author in [22] defines a distributed system as a distributed system is a collection of autonomous computing elements that appears to its users as a single coherent system and based on this definition they gave a characteristic of distributed system i.e. it is a collection of computing elements each being able to behave independently of each other. Similarly, authors of [23] called a distributed system as a collection of autonomous processing nodes that appears to the user as a single processor system. Therefore, a master slave distributed configuration is considered for the proposed system to overcome the faulty node detection for which communication between master and slave nodes is exploited. There are data queues in communication from each slave node to the master node that are given by the distributed system. Along with this an assumption is made that master node is able to communicate with its slaves under all circumstances such that its performance does not act as a bottleneck and do not interfere with the performance of distributed system to complete the task of the specific node.

As mentioned above, this paper presents a method to observe suspected faulty nodes in distributed networked control systems using a communication approach between master node and slave nodes. Moreover, the method can be considered as a semi-parallel approach. The master node identifies the slave nodes, in addition to the channel quality from the slave nodes to master node as shown in Fig. 1. In the Fig. 2, the faulty node detection scheme is performed within pre-processing phase and decision phase, which is divided into the BCH code insertion, fault weighting factor, and decision phase.

A. BCH Code Insertion

The BCH codes are employed as detector to determine the faulty node in the distributed networked control systems. They are effective with respect to a parallel framework. Moreover, BCH codes are generated by the master node using input from 15 pseudo-random bits. The input is the polynomial standard of CAN based on the CRC specification. The master distributes the BCH codes among the slave nodes. Thereafter, the master node collects all the data from the slave nodes to specify the faulty node based on the collected and calculated BCH codes. In particular, BCH code calculation is achieved using two communication models, which involves the encoding and decoding of BCH. This determines the error position of the faulty node using Chien search algorithm [18].

Slave nodes consist of the main data packet and parity code. The parity code is CRC code. Each slave node transforms one bit of the BCH code, into one CRC code, and then combines it with main data packet (e.g., Node 1 only consider the first bit of the BCH code and then combines it with the main data packet. Whereas, node 2 only consider the second bit of the BCH code and combines it with the main data packet, and so on). Data is then transmitted to the master node. Moreover, each node should independently update the BCH code by incrementing one of its values to obtain new BCH sequence, and then transmitting it to master node. The incremented BCH code is required for the specification of the error position at each slave node. Therefore, the master node only check the current and previous BCH code. Through the BCH codes from each slave node, the master node checks if there is an error of the slave node.

B. Faulty Weighting Factor

This step is carried out by the master node for the evaluation of the faulty factor of the suspected node, which is detected by its performance with respect to the calculation of the BCH code sequence. However, through the collection of BCH code error in the master node, the channel error between two faulty nodes is detected. This channel error represents the interval data error if two BCH code error is detected in the master node. Thus, faulty weighting factor is an effective method for the normalization of the observed data.

In previous studies, random weighting was used to achieve smaller mean square errors from multi-sensor observation data [24]. Moreover, the weighting technique was incorporated into the weighted least error squares algorithm to achieve a fast and reliable response from the relay operation [25]. Based on these techniques, the detected faulty error can be weighted by its channel quality.

To evaluate the suspected faulty node behavior, its CRC code are observed. If there is an error due to BCH insertion code, the master node should evaluate the channel quality of the suspected faulty node by initiating the trapping process to observe the CRC code performance. Given that the CRC code is directly related to the channel capacity, the CRC code performance of the suspected node can be periodically observed.

The master node able to detect the error position of slave nodes by collecting the BCH code continuously. The detected error position represents the position of BCH code error occurs and time of the errors are also recorded. By doing this detection with two errors, the error interval can be figure out through each detection error representation. The error interval is the main objective of the data interpretation of the BCH insertion code result. Therefore, it can be conjectured that with a decrease in the error interval in each process, there is an increase in the probability that the node is suspected as the faulty node.

To apply a weighting factor to the error interval of the suspected faulty node, the following analysis can be considered to enhance the error data. Fig. 3 shows an example of behaviors of two nodes. Node A and B represent the collection of the error intervals from the suspected nodes, which is defined by $\{\alpha_1, \alpha_2, \cdots, \alpha_n\}$ and $\{\beta_1, \beta_2, \cdots, \beta_n\}$, respectively.

In the Fig. 3 presents the application of five sampling operations to the CRC code immediately after the detection of the error positions from the BCH insertion code. In summary, the error interval data can be updated as follows,

$$\alpha'_i = \begin{cases} 2\alpha_i & \text{if } \epsilon_i^{CRC} = 0 \\ \alpha_i/\epsilon_i^{CRC} & \text{if } \epsilon_i^{CRC} \neq 0 \end{cases}, \tag{1}$$

where α' is the updated error interval and ϵ_i^{CRC} is the total number of detected CRC code errors at the suspected node. The same equation can be applied to the other error interval data.

If no CRC code errors are detected, the channel condition of

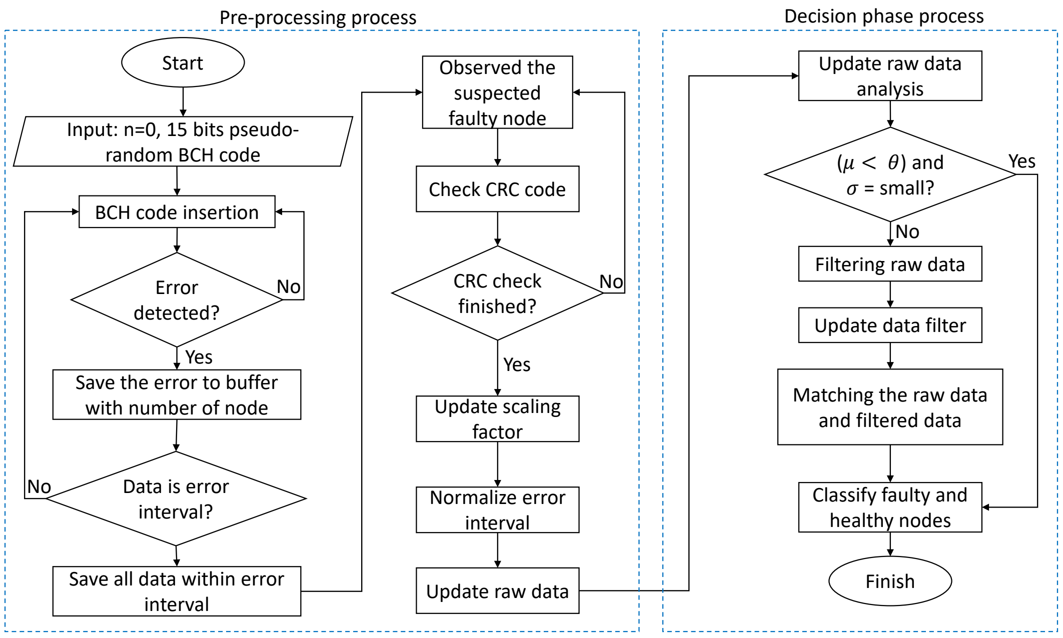


Fig. 2. Flowchart of proposed faulty node detection scheme.

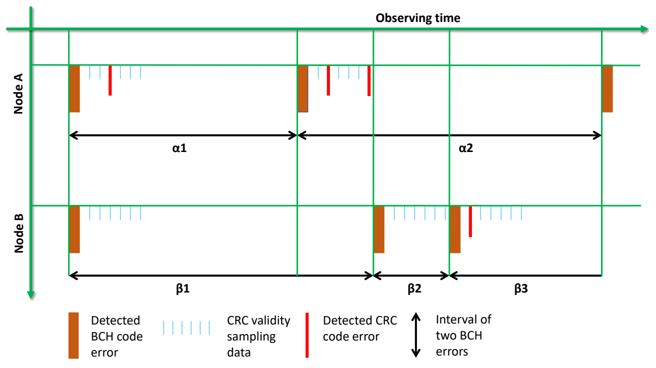


Fig. 3. Data interpretation of the suspected faulty nodes.

the suspected faulty node can be assumed to be clear. Therefore, the only suspected fault is related to the hardware. In this case, if the fault is directly related to the hardware, the channel condition is good, subsequent hardware fault can be accurately detected.

Conversely, if the error from the CRC code is detected, there is a fault in the channel condition of the suspected faulty node. Then, the error that related to the hardware or channel shows the different of the pure channel condition. Therefore, to prevent an undesirable assumption, it is better to increase the awareness of the suspected faulty node by increasing its error interval value by shortening it according to the detected CRC code errors. Based on the pre-processing phase for the building of raw data, followed by the decision phase, as shown in Fig. 2. If the fault monitoring of the master node detects the error positions from the received BCH fragments, the detected time is recorded in the memory. This new detected time is combined with the previous detected error time. Based on this, the error interval data can be generated.

Thereafter, the system initiates a thread at the suspected faulty node. This thread monitors the CRC code behavior within a given time period for a given iteration. During this time period, the total detected CRC code error is recorded. This data is used for the scaling factor of the previous error interval data. When the CRC code monitoring process is complete, then the raw data is updated. After this process, the raw data is then transmitted to the decision phase.

IV. FAULT JUDGEMENT

Fault judgement is a method used to interpret data correctly after the implementation of the BCH code insertion scheme. Fault judgement considers the degree of faultiness that represents the suspected node. The error data received by the master node reflects the pure error position. The error data is collected using the sampling data. To ensure the consistency of the data refinement was employed in this study. It supported by outside factor of BCH insertion code that defined as the performance of suspected node (with respect to the channel quality). Therefore, the nodes and channel performance were combined via the BCH and CRC codes to produced raw data. This step is referred to as the faulty weighting factor or pre-processing phase.

This raw data is interpreted in the decision phase to calculate the fault judgement. The decision phase is divided into three parts, namely, the raw data analysis, filtering data, and pattern analysis.

A. Raw Data Analysis

The raw data is obtained from the faulty weighting factor process. Moreover, the raw data analysis involves basic evaluation of the statistical characteristics of the raw data. The suspected node is concluded while the interval (μ) less than error interval limit (θ) and value of standard deviation (σ) is very small.

Based on this, the judgement of the suspected node can be carried out immediately, i.e., the raw data is sufficient for decision to be reached about the faultiness. Hence, the remainder of the procedure can be disregarded. If the raw data does not meet this requirement, it is processed and analyzed in the filtering phase.

B. *Filtering Phase*

Data from the distributed system is represented by the characteristics of the raw and filtered data. Raw data contains the data that is accumulated by the faulty node as well. The corrupt data generated by faulty nodes can be because of interference, noise as a well due of the inherent fault in the node itself. For the correct identification of the faulty nodes, filtration of the data is carried out using HPF. Along with this, HPF also segregates the frequencies, which results in the enhancement of components of the raw data so that their characteristics can be analyzed as well examined independently. Although, the sharpening and enhancing process of the images using HPF based on a finite impulse response (FIR) [26] does not provide a direct co-relation with the filtering phase that is being used in the proposed method. But, the ultimate aim is same, i.e. segregation of the corrupt noisy data (faulty node) from the raw data.

Thus, the filtering phase is implemented to evaluate the extent of faultiness of the suspected node. The operation of the FIR filter can be expressed as follows,

$$y(t) = \sum_{i=0}^{T-1} h(i)x(t-i), \tag{2}$$

where $y(t)$ is the output data, $x(t)$ is the input data, $h(i)$ is ith coefficients of FIR filter, and T is the total member of FIR filter's coefficients.

In this scheme, we use nine data sequences to represent the FIR filter coefficient, which neglect the affection of the data sequence quantity. The following demonstrates the simplicity of the calculation required to obtain the HPF coefficients for the filtering of the raw data. It is necessary to select the cut-off frequency f_c to realize such a filter. Along with f_c, the normalized transition frequency ω_n is represented as the critical frequency of scaled specification with value 0.25π in the MATLAB filter design function [27]. This normalized transition frequency is defined as $\omega_n = 2\pi f_c/f_s$, where f_s is the frequency sampling.

Moreover, a trade off needs to be made between sampling frequency, cut off frequency and the number of poles. Therefore, the authors of [28], suggested that a factor of 8 is required between the sampling frequency and the filter cut off frequency. Thus, the frequency sampling in this paper was defined as $f_s = 8f_c$. Hence, if the HPF has basic characteristic such as $H(\omega) = \lim_{x\to\infty} 1_{(-x,x)} - 1_{(-\omega_n,\omega_n)}$, then the following equation can be used to calculate coefficients function of the filter.

$$h(t) = \frac{1}{2\pi}\left(\lim_{x\to\infty}\int_{-x}^{x} e^{j\omega t}d\omega - \frac{1}{jt}\left[e^{jw_n t} - e^{-jw_n t}\right]\right). \tag{3}$$

Based on the above equation, the FIR coefficients can be computed as follows,

$$h(t) = \begin{cases} 1 - \frac{\omega_n}{\pi} & \text{if } t \neq T_{1/2} \\ -\frac{\sin(\omega_n(t-T_{1/2}))}{\pi(t-T_{1/2})} & \text{if } t = T_{1/2} \end{cases}, \tag{4}$$

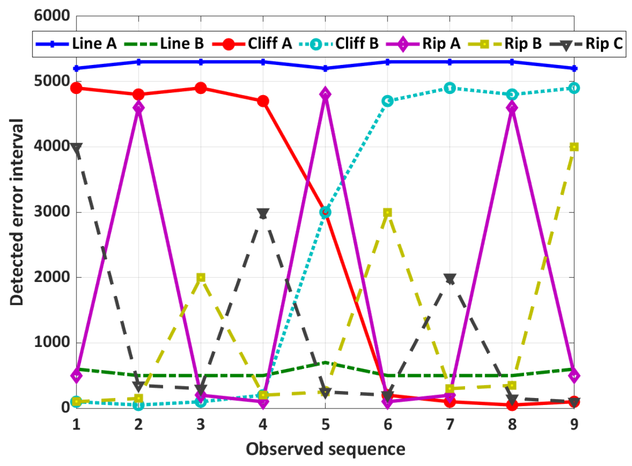


Fig. 4. Faulty decision pattern.

Table 1. State change characteristic.

State type	Parameter	Value	Case
Stable	θ_R, θ_F	$0 \leq \lvert\theta\rvert \leq \theta_{\text{lim}}$	
	ρ_R	$d_v \leq 1$	$\rho_{\text{lim}} \leq \rho_R < \infty$
		$d_v > 1$	$0 < \rho_R$
Destructive	θ_R, θ_F	$\theta_R < -\theta_{\text{lim}}, \theta_F < -\theta_{\text{lim}}$	
	d_v	$d_v > 1$	
Constructive	θ_R, θ_F	$\theta_{\text{lim}} \leq \theta_R, \theta_{\text{lim}} \leq \theta_F$	
	d_v	$d_v > 1$	

where $T_{1/2} = T - 1/2$ and the frequency response is symmetric about $T_{1/2}^{th}$ sample. Therefore, the complete HPF from the previous specification can be formed as follows,

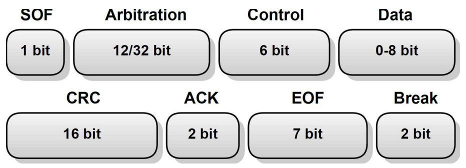
$$h = \{1.038, 1.1, 0.96, 0.684, 0.54, 0.684, 0.96, 1.1, 1.038\}. \tag{5}$$

From the output of this filter, its statistical characteristics are extracted, which is used to evaluate the suspected node. Finally, the final part of the judgement phase is carried out during the faulty pattern analysis.

C. *Pattern Analysis*

Raw data R and filtered data F are matched by their destructive pattern. Fault judgement can be issued when their pattern is sufficiently similar with the decision pattern. This decision pattern uses error interval limit. Moreover, limit is used to determine the alarming status of the node. When, detected error interval is below a certain value, the pattern analysis is carried out. Seven distinctive patterns are employed as the matching pattern. These patterns are classified by their three distinct behavioural features as follows: Line, Cliff, and Ripple patterns. Fig. 4 shows the appointed pattern used to determine faulty node.

The pattern behavioural feature for Line A and B represent stable error interval limit. Line A indicates that the interval limit is higher than the decision limit, whereas line B is lower than decision limit. Cliff A and B represent sudden state changes of interval errors. Cliff A represents sudden state change toward the destructive state, wherein error intervals are shortened. Conversely, cliff B represents the state change toward a better state, wherein error intervals are lengthened. Moreover, Rips A, B,



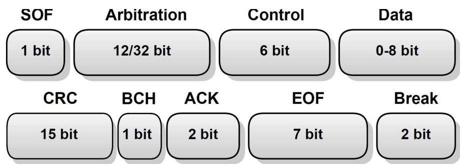
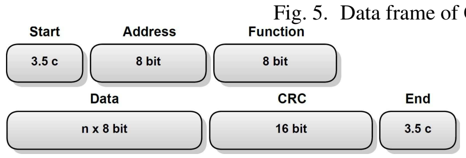


Fig. 5. Data frame of CAN 2.0B: CRC transmission (left) and CRC transmission with BCH (right).



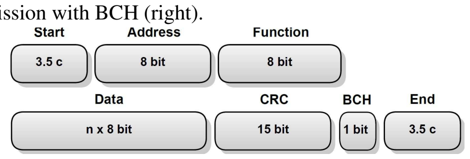
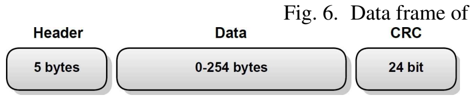


Fig. 6. Data frame of Modbus: CRC transmission (left) and CRC transmission with BCH (right).



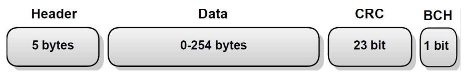


Fig. 7. Data frame of Flexray: CRC transmission (left) and CRC transmission with BCH (right).

Table 2. Description of the notations used in Table 1.

Parameter	Description
d_v	Decision parameter (variable)
θ_R	Error interval for raw data
θ_F	Error interval for filtered data
ρ_R	Statistical parameters for raw data
ρ_F	Statistical parameters for filtered data

and C represent the ripple states, where in the error intervals are both short and long at different instances. Rip A represents the constant ripple state. However, Rips B and C represent the ripple tendency toward longer and shorter error intervals respectively.

Based on these decision patterns, state changes can be further divided into several states as follow: stable state, destructive change state, and constructive change state. The stable state does not indicate that the faulty state did not occur, however, it indicates that the state change was not significant during the observation. Therefore this could indicate that the state is faulty with stable and unchanged error interval characteristics. Hence, the judgement for stable state is determined based on its error intervals whether it is higher or lower than the error interval limit. For the remaining, if the data is in the destructive change state the fault judgement is issued. As conversely, for every constructive change state is consider as a healthy node. Based on these parameters and the previously set decision patterns, the decision characteristic shown in Table 1 were obtained by determining the statistical characteristics of the raw R and filtered F data.

The notation of Table 1 is explained in Table 2, where d_v is defined as $d_v = \rho_F/\rho_R$, where F and R are filtered and raw data respectively and $\rho = \mu/\sigma$. It can be seen that the classification is significantly dependent on the θ value, with the exception of rip A, which has unique characteristics. Based on Table 1, Lines A and B are represented as stable states, and Rip A also represented as a stable state when $d_v > 1$ and $0 < \rho_R$. Given that Rip A also represents a combination of the destructive states, the pattern can be considered as fragile state. Cliff A and Rip C are classified as destructive state, whereas Cliff B and Rip B are classified as constructive state.

V. APPLICATION

This algorithm can be implemented in a Fieldbus system which is a potential technology for distributed networked control systems. Moreover, a low transmission network is required for its application; industrial networked system. There are three candidates: CAN, Modbus, and Flexray. Radio-frequency identification (RFID) technology is comparison between those standards that have longer generator polynomial.

Fig. 5 showed the example of CAN 2.0B's data frame. The left figure is the data frame of CAN 2.0B using CRC transmission, whereas the right figure is the data frame of CAN 2.0B using CRC transmission with BCH insertion. The data frame is significantly protected by coordination code, which contains data that is managed according to the priority. The CRC polynomial standard can be used for false error detection, which is lower than 4.7×10^{-11} times of error rate. The standard polynomial is $x^{15} + x^{14} + x^{10} + x^8 + x^7 + x^4 + x^3 + 1$.

Fig. 6 showed the Modbus's data frame. The left figure is data frame of Modbus using CRC transmission, whereas the right figure is the data frame of Modbus using CRC transmission with BCH insertion. In this field, the interconnection between CAN and Modbus is widely employed. The Modbus and CAN 2.0B have same lengths, but their generator polynomials are different. $x^{16} + x^{15} + x^2 + 1$ is the generator polynomial of Modbus.

Fig. 7 showed the data frame of Flexray. The left figure is the data frame of Flexray using CRC transmission, whereas the right figure is the data frame of Flexray using CRC transmission with BCH insertion. The Flexray generator polynomial length is longer than the CAN and Modbus polynomials, and it can be expressed as follows: $x^{24} + x^{22} + x^{20} + x^{19} + x^{18} + x^{16} + x^{14} + x^{13} + x^{11} + x^{10} + x^8 + x^7 + x^6 + x^3 + x + 1$. The data is constantly monitored by this polynomial, although the length is significantly greater than those of the two other protocols.

As the characteristics of these systems, the faulty node detection scheme can manage more than 15 node in accordance with an increase in the Galois field number of polynomial. Furthermore, the proposed system can be applied for large distributed systems. The Chien search algorithm is involved in the system when more than two errors are detected by calculation. The main objective of the inserted BCH code is to increase the real-

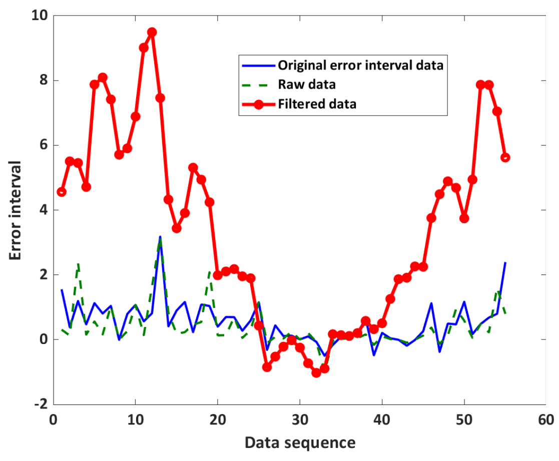


Fig. 8. Data treatment comparison.

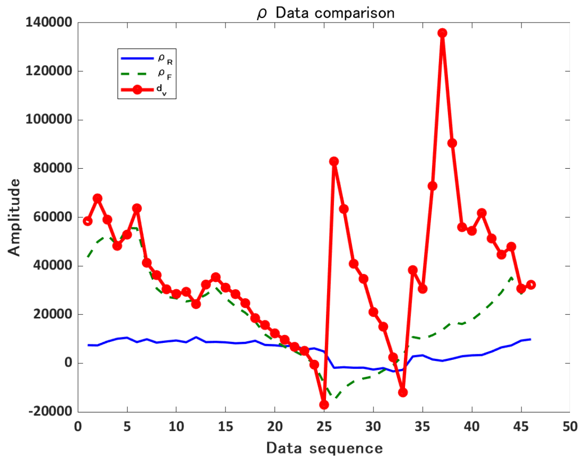


Fig. 9. Statistical characteristic comparison of the observed data: ρ_R, ρ_F and d_v.

time monitoring of faulty node without diminishing the overall system performance for industrial networked control systems.

VI. SIMULATION

A fault judgement simulation represents the error nodes in the monitoring BCH code using the interval error. A fault judgement simulation refers to the mis-detection, as reported in previous work [15]. The mis-detection simulation reveals the calculated error positions due to the BCH fragments. As a direct correlation, this mis-detection simulation for the faulty judgement simulation is the consistencies calculation of the observed node under the various channel qualities. Moreover, the mis-detection of faulty nodes is assumed to be nearly and non-existent, whereas the channel quality is used in the main decision calculation of the faulty node. The simulation results are presented using MATLAB.

Fig. 8 presents comparison of three datasets: Original error, raw data, and filtered data. The original error is interval error of the detected error position during the monitoring of BCH fragments, which is obtained by error interval calculation. The raw data is considered as a wider interpretation of the interval error and channel quality of the suspected faulty node. Moreover, the filtered data represents the raw data after processing using the HPF. As it can be seen, from the graph, suspected node encountered a problem at approximately the 30th data sequence. Moreover, the filtered data was sufficiently amplified for the problem.

Figs. 9 and 10 presents the overall characteristics of the observed data. The d_v, which is already defined as $d_v = \rho_F/\rho_R$, where $\rho_F = \mu_F/\sigma_R$, and $\rho_R = \mu_R/\sigma_R$, and ρ_F and ρ_R represents statistical parameters of filter and raw data, respectively, are shown in Fig. 9. In Fig. 10, the two straight lines denotes the θ_{lim} value, the dynamic solid line denotes the slope of the regressed raw data, and the dashed line denotes the slope of the filtered data. Based on this figure, the angle data comparison suggests that the observed data was mostly under the stable state, with the exception of a significant peak, which was outside of the limit line. Moreover, it was also suggested that at the first 5th data sequence and the last 45th data sequence: the data was in destructive state, although the error interval was significantly longer. However, if the data presented in Fig. 9 is included in the decision making, the observed data exhibited a Rip A characteristic at approximately the 30th sequence of data, where the angle data was within the limit line. There was a relative peak in the value.

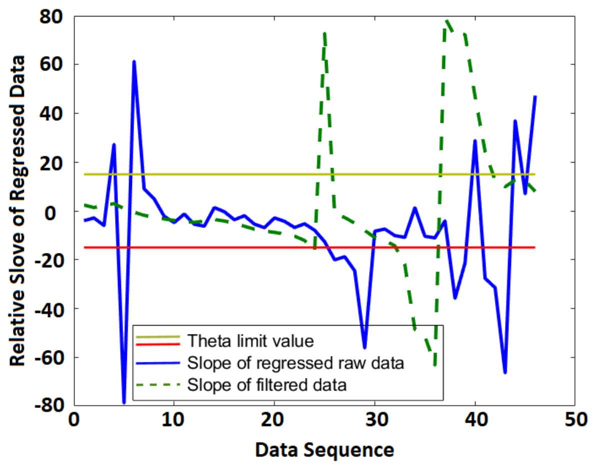


Fig. 10. Statistical characteristic comparison of the observed data: Slope value of regressed data.

VII. CONCLUSION

This paper presents an enhanced faulty node detection method, in which the possibilities of an data errors are investigated through additional checking function. The performance evaluation of distributed networked control systems involves the monitoring of single-bit BCH code within at given time-period. In the faultiness degrees, the output BCH code monitoring and insertion of the CRC code are applied to the normalized data using faulty weighting factor.

There are three datasets that were presented in the simulation result, namely, the original error, raw data, and filtered data.

The original data was obtained by previous work, whereas the raw data and filtered data were obtained by a decision phase process. This detection scheme using weighting factor and decision phase perform that suspected faulty node can be detected in a more transparent way. The final process of the decision phase is the pattern analysis, which provides a detailed representation of the suspected faulty nodes by interpretation into three different state changes. This scheme can be simply applied to distributed networked control systems, to maintain the original error of suspected faulty nodes using the CRC performance and one-bit BCH insertion process. Then, the multi-label classes of faulty node can be considered as future work instead of faulty or healthy node.

REFERENCES

[1] X. Ge, F. Yang, and Q.-L. Han, "distributed networked control systems: A brief overview," *Information Sciences*, vol. 380, pp. 117 – 131, 2017.

[2] M. Nithya and N. U. Maheshwari, "Load rebalancing for Hadoop Distributed File System using distributed hash table," in *Proc. ICISS*, Dec. 2017.

[3] T. A. Courtade and R. D. Wesel, "Optimal Allocation of Redundancy Between Packet-Level Erasure Coding and Physical-Layer Channel Coding in Fading Channels," *IEEE Trans. Commun.*, vol. 59, pp. 2101–2109, Aug. 2011.

[4] Y. Wei and D.-S. Kim, "Enhanced Network Recovery Scheme on Real-time Switched Ethernet for Naval Combat System," *Int. J. Commun. Netw. Distrib. Syst.*, vol. 14, pp. 145–163, Feb. 2015.

[5] T. V. Vu, D. Gonsoulin, D. Perkins, B. Papari, H. Vahedi, and C. S. Edrington, "Distributed control implementation for zonal MVDC ship power systems," in *Proc. IEEE ESTS*, Aug. 2017.

[6] C. Kyrkou and T. Theocharides, "A parallel hardware architecture for real-time object detection with support vector machines," *IEEE Trans. Comput.*, vol. 61, no. 6, pp. 831–842, June 2011.

[7] I. Burlachenko, "Management of energy efficient distributed computer systems with self-contained remote modules using multi-agent system," in *Proc. IEEE ELNANO*, Apr. 2015.

[8] S. Zug, A. Dietrich, and J. Kaiser, "An architecture for a dependable distributed sensor system," *IEEE Trans. Instrum. Meas.*, vol. 60, no. 2, pp. 408–419, Feb. 2010.

[9] T. A. Courtade and R. D. Wesel, "Optimal allocation of redundancy between packet-level erasure coding and physical-layer channel coding in fading channels," *IEEE Trans. Commun.*, vol. 59, no. 8, pp. 2101–2109, Aug. 2011.

[10] Y. Wei and D.-S. Kim, "Enhanced network recovery scheme on real-time switched ethernet for naval combat system," *Int. J. Commun. Netw. Distrib. Syst.*, vol. 14, no. 2, pp. 145–163, Feb. 2015.

[11] D.-H. Choi and D.-S. Kim, "Wireless fieldbus for networked control systems using LR-WPAN," *Int. J. Control Autom. Syst. 2008*, pp. 119–125, Jan. 2008.

[12] M. Barranco, J. Proenza, and L. Almeida, "Quantitative comparison of the error-containment capabilities of a bus and a star topology in CAN networks," *IEEE Trans. Ind. Electron.*, vol. 58, no. 3, pp. 802–813, Nov. 2009.

[13] H. T. Mouftah, M. Erol-Kantarci, and M. H. Rehmani, *Transportation and Power Grid in Smart Cities: Communication Networks and Services*. John Wiley & Sons, 2018.

[14] H. Zeng, M. Di Natale, A. Ghosal, and A. Sangiovanni-Vincentelli, "Schedule optimization of time-triggered systems communicating over the FlexRay static segment," *IEEE Trans. Ind. Informat.*, vol. 7, no. 1, pp. 1–17, Feb. 2010.

[15] A. Prasetiadi and D. Kim, "Faulty Node Detection in Distributed Systems Using BCH Code," *IEEE Commun. Lett.*, vol. 17, pp. 620–623, Mar. 2013.

[16] M. Royyan, J.-H. Cha, J. M. Lee, and D.-S. Kim, "Data-driven faulty node detection scheme for wireless sensor networks," in *Proc. WD*, Mar. 2017.

[17] D. Park, S. Kim, Y. An, and J.-Y. Jung, "LiReD: A Light-Weight Real-Time Fault Detection System for Edge Computing Using LSTM Recurrent Neural Networks," *Sensors*, vol. 18, no. 7, June 2018.

[18] S.-Y. Wong, C. Chen, and Q. J. Wu, "Low power Chien search for BCH decoder using RT-level power management," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 19, no. 2, pp. 338–341, Oct. 2009.

[19] Z. Chen, H. Fang, and Y. Chang, "Weighted Data-Driven Fault Detection and Isolation: A Subspace-Based Approach and Algorithms," *IEEE Trans. Ind. Electron.*, vol. 63, no. 5, pp. 3290–3298, May 2016.

[20] J. Fang, S. Wan, P. Huang, C. Xie, and X. He, "Early Identification of Critical Blocks: Making Replicated Distributed Storage Systems Reliable Against Node Failures," *IEEE Trans. Parallel Distrib. Syst.*, vol. 29, no. 11, pp. 2446–2459, Nov. 2018.

[21] S. Kelkar, D. G. Yeole, M. B. Sinkar, P. B. Jagtap, and D. S. Zagade, "Coordinator-based adaptive fault diagnosis algorithm for distributed computing systems," in *Proc. ICACCI*, Sept. 2017.

[22] M. van Steen and A. S. Tanenbaum, "A brief introduction to distributed systems," *Computing*, vol. 98, p. 967–1009, Oct. 2016.

[23] A. Deyasi, S. Mukherjee, P. Debnath, and A. K. Bhattacharjee, *Computational Science and Engineering*. EH Leiden, The Netherlands: CRC Press/Balkema, 2016.

[24] S. Gao, Y. Zhong, and W. Li, "Random Weighting Method for Multisensor Data Fusion," *IEEE Sensors J.*, vol. 11, pp. 1955–1961, Sept. 2011.

[25] P. Jafarian and M. Sanaye-Pasand, "Weighted least error squares based variable window phasor estimator for distance relaying application," *IET Generation, Transmission Distribution*, vol. 5, pp. 298–306, Mar. 2011.

[26] C. Ciulla and G. Agyapong, "Intensity-curvature functional based digital high pass filter of the bivariate cubic B-spline model polynomial function," *Visual Comput. Ind. Biomedicine Art*, vol. 2, no. 1, p. 9, 2019.

[27] W. Alexander and C. Williams, "chapter 4 - design of digital filters," in *Digital Signal Processing*, W. Alexander and C. Williams, Eds. Boston: Academic Press, 2017, pp. 205 – 275.

[28] J. D. Irwin, *The industrial electronics handbook*. CRC press, 1997.

Riesa Krisna Astuti Sakir received her Master's degree from the Department of IT Convergence Engineering, Kumoh National Institute of Technology, South Korea in 2020. Her research interests are in the area of fog computing, real time systems, and industrial IoT.

Sanjay Bhardwaj received his Ph.D. degree from the Department of IT convergence Engineering, Kumoh National Institute of Technology, Gumi, South Korea in 2020. From 2012 to 2018 he worked as Assistant Professor in the Department of Electronics and Communication at Shoolini University, India. He is currently Postdoctoral Researcher at ICT Convergence Research Center, Kumoh National Institute of Technology, Gumi, South Korea. His research areas of interest are bio-inspired CRNs, IoT, IIoT and URLLC in the industrial wireless network.

Prof. Dong-Seong Kim received his Ph.D. degree in Electrical and Computer Engineering from the Seoul National University, Seoul, Korea, in 2003. From 1994 to 2003, he worked as a full-time Researcher in ERC-ACI at Seoul National University, Seoul, Korea. From March 2003 to February 2005, he worked as a Postdoctoral Researcher at the Wireless Network Laboratory in the School of Electrical and Computer Engineering at Cornell University, NY. From 2007 to 2009, he was a Visiting Professor with Department of Computer Science, University of California, Davis, CA. He is currently a Dean of industrial academic cooperation foundation and ICT Convergence Research Center (ITRC and NRF advanced research center program) supported by Korean government at Kumoh National Institute of Technology. He is a senior member of IEEE and ACM. His current main research interests are real-time IoT and smart platform, industrial wireless control network, networked embedded system and Fieldbus.