

# Intelligent Sensing with Multiagent-based Wireless Sensor Network for Bridge Condition Monitoring System

Putra, S. A., Trilaksono, B. R., Riyansyah, M., Laila, D. S., Harsoyo, A. & Kistijantoro, A. I.

Author post-print (accepted) deposited by Coventry University's Repository

**Original citation & hyperlink:**

Putra, SA, Trilaksono, BR, Riyansyah, M, Laila, DS, Harsoyo, A & Kistijantoro, AI 2019, 'Intelligent Sensing with Multiagent-based Wireless Sensor Network for Bridge Condition Monitoring System' IEEE Internet of Things Journal, vol. (In-Press), pp. (In-Press).

<https://dx.doi.org/10.1109/JIOT.2019.2901796>

DOI 10.1109/JIOT.2019.2901796

ESSN 2327-4662

Publisher: Institute of Electrical and Electronics Engineers (IEEE)

**© 2019 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.**

**Copyright © and Moral Rights are retained by the author(s) and/ or other copyright owners. A copy can be downloaded for personal non-commercial research or study, without prior permission or charge. This item cannot be reproduced or quoted extensively from without first obtaining permission in writing from the copyright holder(s). The content must not be changed in any way or sold commercially in any format or medium without the formal permission of the copyright holders.**

**This document is the author's post-print version, incorporating any revisions agreed during the peer-review process. Some differences between the published version and this version may remain and you are advised to consult the published version if you wish to cite from it.**

# Intelligent Sensing in Multiagent-based Wireless Sensor Network for Bridge Condition Monitoring System

Seno Adi Putra, Bambang Riyanto Trilaksono, *Senior Member IEEE*, Muhammad Riyansyah, Dina Shona Laila, *Senior Member IEEE*, Agung Harsoyo, Achmad Imam Kistijantoro

**Abstract**—This work proposes the development of an autonomous system for dynamic response based bridge condition assessment using Wireless Sensor Network (WSN). The assessment identifies the bridge’s fundamental frequency and uses the information to determine the bridge rating. Due to the computational capability in wireless sensor nodes, it is of practical interest to implement in-network processing in bridge condition monitoring system, in which data processing is conducted within the sensor networks to prevent data flooding in WSN. One of the promising in-network processing approaches is the agent-based processing that leverages the concept of system autonomy. However, uncontrolled in-network processing consumes a lot of energy. Thus, setting all sensors to wake up or sleep deterministically is often not a feasible solution. What is needed is for the system to perform in-network processing only in the event when the bridge is traversed by a single heavy truck, whereas this event occurs randomly. Thus, the two-player game and reinforcement learning algorithm are utilized to control the process. Simulation results show that the proposed control algorithm is able to effectively determine when the process should be executed. A case study, testing the algorithm using real measurements taken from a bridge, and then comparing the test results with the results generated from finite element analysis (FEA) is provided for validation purpose. Comparison of the proposed approach with earlier works, in terms of processing time and energy consumption, is also presented.

**Index Terms:** wireless sensor network (WSN), in-network processing, bridge rating, multiagent system, two-player game, and reinforcement learning.

## I. INTRODUCTION

Structural health monitoring systems (SHMS) plays an important part in bridge management during its service life. SHMS comprises on-site destructive and non-destructive sensing and testing of structural characteristics, including structure responses that indicate structural deformation. As non-destructive test for materials and structure have been more available, recent works have been focusing mainly on integrating these non-destructive methods to structural health monitoring (see [1] and references therein). However, research and development

related to the exploitation of autonomous systems for collecting, processing, evaluating, and disseminating information for bridge health conditions monitoring application is still limited.

Works related to structural health monitoring system are divided into three areas. The first emphasizes on methodology development for data processing that leverages the reliable data transmission approach [2], statistical classifier [3], signal analysis using machine learning [4], in-network processing method [5] and [6], and damage detection algorithms [7]. The second addresses architecture design that proposes the process and product of planning, designing, and constructing system from hardware to software development such as system development from hardware to application software with high time-synchronization accuracy [8], hardware development for wireless powering [9], system development for self-powered sensor [10], hardware development for WIM [11], accelerometer-based WIM development [12], and SHMS frameworks using IoT technology [13]. The third category focuses on the domain-specific application, in this case to develop an overall system for bridge condition assessment using the bridge dynamic response [14].

The work of this paper contributes to all the three areas. Some previous works which are closely related to our work are the bridge condition assessment using dynamic response proposed in [14] and online SHM algorithm proposed in [6]. Both have used frequency-based damage detection method. The drawback of this bridge condition assessment method is it is traditionally conducted manually, in which raw data collected from WSN are processed using an *Excel spreadsheet* to obtain the bridge’s fundamental frequency and its amplitude. The online SHM algorithm improves the traditional manual approach by incorporating an in-network processing method that processes data in each sensor node.

An obvious problem found in the traditional approach is data flooding. In this approach sensor nodes always transmit raw data to the sink node. It produces a large amount of data that must be managed in the WSN and the server. Those data transmitted to the sink node do not always represent the occurrence of significant events. In the bridge condition monitoring practice, we only need to record the bridge response when a heavy truck passes over the bridge, whereas other lighter vehicles may be neglected due to their insignificant effect on the bridge vibration. Therefore, we need to consider an effective sensing method in which sensing and measurement should be conducted only when a heavy moving truck is passing over the bridge.

While the online SHM algorithm has improved the traditional approach, it has not considered the effective sensing of different vehicles as mentioned above. Thus, large and inefficient energy consumption is still an issue in both approaches. When using the WSN, the limitations of

---

• S.A. Putra, B.R. Trilaksono, M. Riyansyah, A. Harsoyo and A.I. Kistijantoro are with Institute of Technology Bandung (ITB), Bandung 40132, Indonesia. S.A. Putra is also with Telkom University, Bandung 40257, Indonesia. D.S. Laila is with the School of Mechanical, Aerospace & Automotive Eng., Coventry University, Coventry CV1 5FB, UK.

Corresponding authors: seno\_ap@yahoo.com dina.laila@coventry.ac.uk.

• The authors thank the LPDP (Indonesia Endowment for Education) to sponsor S.A. Putra’s PhD project and the Royal Academy of Engineering for the Newton Funds award IAPP1\_100018 to support this research.

• Copyright (c) 2012 IEEE. Personal use of this material is permitted. However, permission to use this material for any other purposes must be obtained from the IEEE by sending a request to [pubs-permissions@ieee.org](mailto:pubs-permissions@ieee.org).

the available energy should be considered. In WSNs, each sensor contributes to the application as long as its battery has sufficient charge to keep the sensor active. Almost 60% of energy will be released when a sensor is in an *active mode* to communicate with other sensors [15].

Another problem faced by the traditional approach and online SHM algorithm is the time delay when the data sent by all sensor nodes are received by the sink node. The base station which is part of the sink node where all communication channels are opened could not handle messages or data sent by sensor nodes simultaneously. The queue of data occurs in the base station's buffer, causing the time delay. It is concluded that none has applied an intelligent sensing that operates autonomously, taking energy consumption and processing time into consideration. Therefore, we propose the use of intelligent sensing to overcome problems found in both earlier approaches.

The proposed intelligent sensing in this work leverages the conceptual multiagent architectures [16], [17], [18], [19], [20], the mobile agents approaches [21], [22], [23], [24], [25], [26], and [27], and some intelligent WSN behaviour [28] and [29] are combined to minimize the time delay. Works that put all these concepts together are still scarce. Thus, it is our novel idea in architectural design.

TABLE I  
SYSTEM COMPARISON

The Related Work	Description
The bridge condition assessment using dynamic response [14]	<ol style="list-style-type: none"> <li>1) Topology: centralized processing</li> <li>2) Sensor nodes always sense and send the raw data to sink node with 100 Hz sampling frequency.</li> <li>3) The FFT analysis and peak picking is performed manually using spread sheet.</li> <li>4) The application displays bridge properties and bridge rating and capacity.</li> <li>5) Limitation: energy consumption and latency</li> </ol>
Online SHM algorithms [6]	<ol style="list-style-type: none"> <li>1) Topology: cluster-based processing</li> <li>2) Each sensor node performs FFT analysis and mode shape assembling.</li> <li>3) Sensor nodes always sense and the data transmission to sink node is performed if the indication of possible damage occurs.</li> <li>4) The application displays bridge properties, fundamental frequencies, and mode shape.</li> <li>5) Limitation: energy consumption and latency</li> </ol>
Our proposed work	<ol style="list-style-type: none"> <li>1) Topology: cluster-based processing</li> <li>2) The sensor nodes perform FFT analysis and mode shape assembling.</li> <li>3) The data transmission to sink node is performed using mobile agent approach.</li> <li>4) Control mechanism is implemented in which the data processing and transmission in each sensor node will be sent if there is a heavy truck passing over the bridge. The node's sleep and wake up mode controlling is provided.</li> <li>5) The application displays bridge properties, bridge rating, and mode shape.</li> <li>6) Limitation: using accelerometer sensor only, bridge capacity determination is not covered.</li> </ol>

The main contribution of this work is to improve bridge condition assessment [14] and online SHM algorithm [5] by incorporating the intelligent system for autonomous bridge condition monitoring. The multiagent system leveraging the system autonomy and intelligent agent characteristics such as reactive, proactive and social [30] is presented. The bridge condition assessment requires these characteristics, especially to make decision when sensor nodes should be

set to *active mode* to capture the significant events and when they should be set to *sleep mode* as long as possible to save the energy while still maximizing the probability to capture important events. The comparison of our proposed system and the previous works is described in Table I.

The scopes of this work specifically include the multiagent system design for in-network processing implementation in bridge condition monitoring system that recontextualizes our previous works [31] and [32], the development of effective sensing method that employs agents in two WIM nodes in such a way to control when sensing and data processing should be conducted, and the two-player game and reinforcement learning [33] recontextualization in two WIMs interaction model.

The rest of this paper is organized as follows. Section 2 describes the bridge condition assessment. Section 3 discusses the proposed multiagent system, while Section 4 describes the proposed methods. Section 5 presents a case study and Section 6 provides some simulation results, comparing our propose approach with existing methods. Finally, Section 7 gives the conclusions and future works.

## II. BRIDGE CONDITION ASSESSMENT

In this work, the bridge is idealized as a single degree of freedom (SDOF) and configured as simply supported bridge. The bridge condition is indicated by a parameter, namely the *bridge rating*, which has a correlation with its dynamic response when passed by a heavy truck.

The procedure for a bridge assessment follows several steps [14]: 1) deploying a WSN on the bridge to collect the bridge's vibration data when it is passed over by a heavy truck; 2) performing Fast Fourier Transform (FFT) on the collected vibration data and identify the peak amplitude of the first fundamental frequency; 3) performing a Finite Element Analysis (FEA) from the bridge construction image representing the bridge ideal condition; 4) running a FEA simulation to identify the bridge vibration first fundamental frequency and amplitude; 5) identifying the difference between the frequency identified from the FEA and from the field test using real sensors; 6) calculate the bridge rating  $R$  using the formula

$$R = (int)9 - \frac{f_{FEA} - \bar{F}}{f_{FEA}} * \frac{1000}{123} \quad (1)$$

where  $f_{FEA}$  is the bridge fundamental frequency calculated from FEA,  $\bar{F}$  is the current bridge fundamental frequency measured by WSN. Table II describes the bridge's conditions and their corresponding required treatment.

TABLE II  
THE BRIDGE CONDITION AND TREATMENT

Rating	Description	Treatment
0	Failed condition	Rehabilitation or replacement
1	Imminent Failure Condition	
2	Critical Condition	
3	Serious Condition	Preventive
4	Poor Condition	maintenance or repairs
5	Fair Condition	
6	Satisfactory Condition	
7	Good Condition	Preventive maintenance
8	Very Good Condition	
9	Excellent Condition	

Based on theoretical analysis, the fundamental frequency of the bridge idealized as SDOF and simply supported configuration can be calculated as

$$f = 1.5708 \sqrt{\frac{EI}{ML^4}} \quad (2)$$

where  $EI$  is the effective flexural rigidity of the bridge,  $M$  is the total mass of the bridge, and  $L$  is the bridge length. The detailed description of bridge types is described in [14]. The numerical analysis of bridge fundamental frequency can be conducted using application software such as *CSiBridge*, which was used in this work.

In order to measure the bridge rating, collaboration between accelerometer-based WIM and accelerometer sensor nodes (ACC), which are deployed on two independent spans of the bridge as shown in Fig. 1, is proposed in this work. The WIM nodes identify truck type (II, III, and, IV) while the ACCs identify the bridge fundamental frequency. The WIMs are placed in the middle of the bridge and always be connected to power supply and the ACCs are placed on the main component of the bridge to obtain significant vibration data. The sink nodes are mini computers or single board computers that process and store data from the ACC nodes. They also coordinate the ACC nodes, monitor in-network processing and communicate with a cloud server. An agent performs such processes and a mobile agent dispatching is deployed in sink nodes.

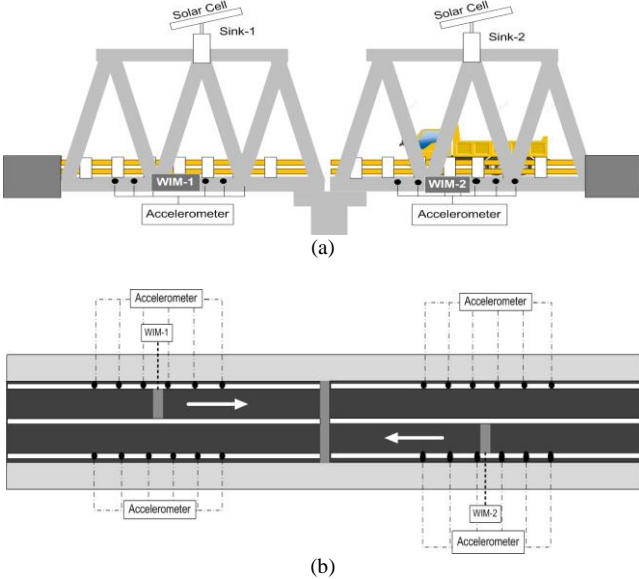


Fig. 1 The WIM, ACC, and sink deployment: (a) side and (b) top view

The proposed sensing strategy will work for two lanes of the bridge with different direction. In Fig. 1b vehicles are not allowed to overtake each other along the bridge. To prevent a complicated mode analysis, two-way driving situation is excluded. Only the case when the bridge is passed over by a heavy truck at one time is considered.

When a heavy truck approaches the bridge, e.g. from the right side as seen in Fig. 1a, the WIM deployed on the right side chooses an action from its available actions at the current time slot based on its action probability distribution. If it detects a heavy truck, it transmits a warning message to

its counterpart or the WIM on the left side to start sensing in the counterpart WSN area. If the counterpart WIM receives the warning message, it transmits a message to its ACC nodes to wake them up and start capturing data.

### III. THE PROPOSED MULTIAGENT SYSTEM

This work leverages a multiagent-based in-network processing protocol for bridge condition assessment application. The concept of a mobile agent-based protocol [34] was utilized. The communication protocols among sensor nodes, between sensor nodes and sink node, between WIMs and sensor nodes, and between two WIMs were also designed.

The developed agents are based on Belief Desire Intention (BDI) using Agent Factory Micro Edition (AFME). They consist of a perceptor, actuator, module, and service [21]. The perceptor generates beliefs, the actuator executes an action environment based on beliefs generated by the perceptor, the module is an information space that is used for data shared by perceptor and actuator, and service represents a space for data sharing among agents. The agent employment is illustrated in Fig 2. Here the system module is divided into two layers: *in-network processing* and its *control or scheduling*.

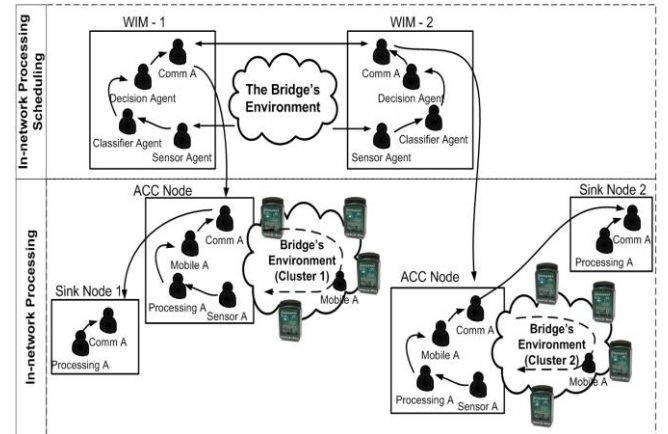


Fig. 2 The proposed multiagent system

ACC node contains four agents. The communication agent (CommA) is responsible for communication with the neighbour ACC node, WIM node, and the sink node. The processing agent (ProcessingA) performs data preprocessing. The Sensor A is responsible for capturing accelerometer data. The Mobile A migrates from the sensor node to other nodes to collect data.

The WIM nodes consist of four agents. The first agent is CommA, which accesses the transceiver of a sensor node to communicate with its counterpart WIM and ACC nodes within its area. The second agent is SensorA that senses the bridge vibration. The acceleration data of the bridge vibration are sent to the Classifier Agent to classify truck's type. Based on this classification, the Classifier Agent sends a message to the Decision Agent indicating the occurrence of a significant event. Then, the Decision Agent executes the reinforcement learning algorithm to choose the best action. The messages involved in the system are described in Table III.

TABLE III  
MESSAGE TYPES INVOLVED IN SYSTEM

Message	Description
IST	Internal State Report is a message initiated by a sensor node to report the sensor node's condition to sink node and its neighbours
SSC	Send Sensing Command is a message from WIM commanding all sensor nodes to broadcast its internal state and start capturing the sensed data
RMA	Request Mobile Agent is a message initiated by a sensor node to request a mobile agent dispatching
SRC	Send Registration Command is a message from a WIM to wake sensor nodes up and command them to register themselves to their neighbours and sink node
CCR	Cost Calculation Report is a message from a sensor node to the sink node so that the sink node executes mobile agent migration plan
RSN	Register Sensor Node is a message from a sensor node to inform its condition to the sink and its neighbours
RSA	Receive Sensor Data is a message submitted by a mobile agent to request the sink node to accept data
SE	Significant Event is a warn message from a WIM to another WIM indicating the arrival of a truck on the bridge
IPR	In-network Processing is Running is a message from a WIM to inform its counterpart that in-network processing in its area is running
ND	Network is Dead is a message from a WIM sent to its counterpart indicating that in-network processing failed or is incomplete within a certain time duration
RIP	Reject in Network Processing is a response message sent by a WIM to inform its counterpart that in-network processing cannot be executed
IPC	In-network Processing is Completed is a message sent by a WIM to its counterpart indicating the in-network processing is successful in its area
COS	Check Counterpart's State is a request submitted by a WIM to check its counterpart's state
CSI	Counterpart State Info is a message from a WIM to its counterpart informing its current environmental state

In the sink node `CommA` and `ProcessingA` are defined. `CommA` plays a role in accessing the transceiver so that the sink node can communicate with the ACC and the WIM nodes. `ProcessingA` handles in in-network processing including the ACC nodes condition identification, mobile agent migration plan, mobile agent dispatching, and communication with the server.

#### IV. THE PROPOSED METHODS

This section describes the algorithms that are employed in agents of ACC sensor nodes, sink nodes, and WIMs. The description starts from the first layer called *in-network processing* layer and then the upper layer called *in-network processing control* or *scheduling* as illustrated in Fig. 2.

##### 4.1 In-network processing algorithm

In-network processing is a computational process conducted at each ACC node. Here, the process includes:

- 1) Each ACC node placed in a cluster announces itself to the sink node and neighbour nodes after receiving the SRC command from a WIM (see Table III). Registration is conducted by ACC node's `CommA` that transmits IST messages to their neighbour and the sink node. Registration includes its signal strength  $p$  and remaining energy level  $e$ . The sink and sensor nodes then identify active nodes. Sensors with energy level lower than 30% cannot be involved.

- 2) The sink node's `ProcessingA` performs mobile agent migration plan using the optimization function

$$\max O(p, e) = \sum_{i=1}^N \sum_{j=1}^N c_{ij} (w_p p_{ij} + w_e e_{ij}) \quad (3)$$

where  $c_{ij} = 1$  if the  $i^{\text{th}}$  and  $j^{\text{th}}$  sensor nodes are part of a mobile agent route and  $c_{ij} = 0$  if the  $i^{\text{th}}$  and  $j^{\text{th}}$  nodes are not a part of a route.  $w_p$  and  $w_e$  are the weight of the signal strength and the energy level, respectively, while  $p_{ij}$  and  $e_{ij}$  are the signal strength and the energy level between the  $i^{\text{th}}$  and  $j^{\text{th}}$  sensor node, respectively. The rules of this function are as follows: the mobile agent dispatches from a sensor node to another sensor node; a sensor node can only be visited from another sensor node; and the mobile agent cannot visit the  $i^{\text{th}}$  to the  $j^{\text{th}}$  sensor nodes and cannot go back from the  $j^{\text{th}}$  to the  $i^{\text{th}}$  sensor node. To find the optimal route, genetic algorithm is used.

- 3) After receiving a SSC from a WIM, all ACC nodes' `ProcessingA` calculate the fitness of their neighbour and all ACC nodes' `SensorA` start to sense the bridge vibration. `ProcessingA` then transforms the acceleration data of the bridge vibration to frequency domain to obtain the first fundamental frequency and its amplitude as shown in *Algorithm 1*.

##### Algorithm 1 Data Processing in Each Sensor Node

```

1: Input: Given two WSNs deployed on a bridge structure;
2: Output: Bridge rating;
3: for each individual sensor node  $S_i$  in a cluster do
4:   capture  $2^N$  data of bridge vibration  $A = \{a_1, a_2, \dots, a_n\}$ ;
5:   perform FFT analysis on data  $A$ ;
6:   identify first fundamental frequency of the structure  $f_i^1$ ;
7:   identify peak amplitude  $\Phi_i^1$  of first fundamental frequency
8:     of the structure  $f_i^1$ ;
9:   request mobile agent to sink node;
10: end

```

- 4) The sensor node's `CommA` then requests a mobile agent to the sink node by sending RMA message. After receiving mobile agent requests from all sensor nodes, the sink node dispatches a mobile agent to ACC's network. The mobile agent task when arrives in first,  $i^{\text{th}}$ , and last node is described in *Algorithm 2*.

##### Algorithm 2 Data Processing in Mobile Agent

```

1: Input: bridge reference from finite element analysis  $f_{new}$ ;
2: for each individual sensor node  $S_i$  in a cluster do
3:   if current node is first or  $i^{\text{th}}$  node then
4:     get  $f_i^1$  and  $\Phi_i^1$  from node's local memory;
5:     add  $f_i^1$  and  $\Phi_i^1$  in mobile agent's vector of  $F_i^1$  and  $\Phi_i^1$ ;
6:     identify next hop sensor node address;
7:     take mobile agent off;
8:   else current node is last node
9:     get  $f_N^1$  and  $\Phi_N^1$  in node's local memory;
10:    add  $f_N^1$  and  $\Phi_N^1$  to mobile agent's vector of  $F_i^1$  and  $\Phi_i^1$ ;
11:    calculate average frequency data  $\bar{F}$ ;
12:    calculate bridge rating  $R$ ;
13:    send  $R$  and vector data of  $\Phi_i^1$  to sink node;
14:    make mobile agent sleep;
15:   end
16: end

```

#### 4.2 In-network processing scheduling algorithm

To prevent uncontrolled in-network processing that potentially depletes the energy of the ACC sensor node, the in-network processing control is proposed. The control function maintains the time ratio between when the ACC nodes are active and when they are inactive. The Decision Agent of WIM node is taken into account in making that decision. Here, the Decision Agent's modes such as In-network processing, Sleep, Wake, and Warn are defined.

Mode 1 is In-network processing that refers to a mechanism of activating the ACC nodes from sleep to sense bridge vibration, perform FFT and peak picking, and transmit the results to the sink node using mobile agent. Mode 2 is Sleep during which a WIM can neither execute in-network processing nor send a request to its counterpart for executing in-network processing. For example, if WIM-1 detects a heavy truck and WIM-2 is in sleep mode, WIM-2 can neither accept WIM-1's request nor send a request message to WIM-1. Mode 3 is Wake that is required for the WIM to accept the request from its counterpart for in-network processing execution. In this mode, a WIM cannot request its counterpart to execute in-network processing even when it detects a heavy truck. Mode 4 is Warn in which if a WIM detects a heavy truck, it transmits an in-network processing request message to its counterpart.

The states No truck, One truck, and Many trucks are also defined. No truck means in the current time slot the WIM detects no truck. One truck and Many trucks indicate that the WIM detects one and many trucks respectively in the current time slot. Ideally, in-network processing should be run when a single truck is passing over the bridge.

##### 4.2.1 Two-player game approach

The interaction between both WIMs is modelled according to a two-player game approach with three actions: Sleep, Wake and Warn. The game is modelled as reward or punishment matrices, as given by

$$R_{WIM}^1 = \begin{bmatrix} r_{11}^1 & r_{12}^1 & r_{13}^1 \\ r_{21}^1 & r_{22}^1 & r_{23}^1 \\ r_{31}^1 & r_{32}^1 & r_{33}^1 \end{bmatrix}, R_{WIM}^2 = \begin{bmatrix} r_{11}^2 & r_{12}^2 & r_{13}^2 \\ r_{21}^2 & r_{22}^2 & r_{23}^2 \\ r_{31}^2 & r_{32}^2 & r_{33}^2 \end{bmatrix} \quad (4)$$

where the superscripts 1 and 2 refer to WIM-1 and WIM-2, respectively and the subscripts 1, 2, and 3 refer to the action Sleep, Wake, and Warn, respectively. Each WIM will select an action according to their available actions. The joint action of both WIMs determines the rewards they receive according to their reward matrices. If WIM-1 selects action  $x$  and WIM-2 selects action  $y$ , then WIM-1 receives the reward  $r_{xy}^1$  and WIM-2 receives the reward  $r_{yx}^2$ . The actions can be selected based on the policy of their actions, which are updated using a reinforcement learning algorithm. The expected rewards for WIM-1 and WIM-2 are calculated respectively by

$$X_{WIM}^1 = \sum_{x=1}^N \sum_{y=1}^N r_{xy}^1 \pi_x^1 \pi_y^2 \quad (5)$$

$$X_{WIM}^2 = \sum_{y=1}^N \sum_{x=1}^N r_{yx}^2 \pi_y^2 \pi_x^1 \quad (6)$$

where  $\pi_x^1$  is the probability of action  $x$  and  $\pi_y^2$  is the probability of action  $y$  selected by the first and second WIM respectively. In each time slot, the bridge is under one of nine states as illustrated in Table IV.

TABLE IV  
LIST OF AVAILABLE STATE

State	WIM 1	WIM 2	Illustration
1	No truck	No truck	
2	No truck	One truck	
3	No truck	Many trucks	
4	One truck	No truck	
5	One truck	One truck	
6	One truck	Many trucks	
7	Many trucks	No truck	
8	Many trucks	One truck	
9	Many trucks	Many trucks	

The *reward* is defined as the energy consumption used by a WIM plus an additional bonus corresponding to the current state. The parameter bonus includes the detected truck type, identified by WIM's Classifier Agent in current time slot, times average remaining energy of ACC nodes. For example, if the state is number 4 (see Table III) and WIM-1 selects the action Warn and WIM-2 selects the action Wake, then in-network processing will be executed in WIM-2's area and the *rewards* are  $r_{32}^1 = (\epsilon' - \epsilon) + \bar{\xi}m$  for WIM-1 and  $r_{23}^2 = (\epsilon' - \epsilon) - a$  for WIM-2.  $\epsilon'$  and  $\epsilon$  are the current energy level and the previous energy level of both WIMs, respectively, when performing an action.  $m$  is the truck type, which takes the value of 2, 3 or 4 indicating Type II, Type III, or Type IV respectively.  $\bar{\xi}$  is the average remaining energy of ACC sensors ( $0 \leq \bar{\xi} \leq 100$ ).

The reward matrices can be set according to developer preferences as long as the reward fits within particular state. Each state has its own reward matrices as shown in Table V that defines  $-\Delta\epsilon_{xy}$  as the energy consumption when a WIM performs action  $x$  and its counterpart performs action  $y$ .

Notation  $a$  is a constant that makes sure the reward matrix has an inverse.  $\xi m > a$  is the bonus when in-network processing is executed in a counterpart's area.

TABLE V  
LIST OF REWARD MATRICES

State	Reward Matrices
1, 3, 7, 9	$\begin{bmatrix} -\Delta\epsilon_{11} & -\Delta\epsilon_{12} & -\Delta\epsilon_{13} \\ -\Delta\epsilon_{21} & -\Delta\epsilon_{22} & -\Delta\epsilon_{23} - a \\ -\Delta\epsilon_{31} & -\Delta\epsilon_{32} - a & -\Delta\epsilon_{33} \end{bmatrix}$
2, 8	$\begin{bmatrix} -\Delta\epsilon_{11} & -\Delta\epsilon_{12} & -\Delta\epsilon_{13} \\ -\Delta\epsilon_{21} & -\Delta\epsilon_{22} & -\Delta\epsilon_{23} + \xi m \\ -\Delta\epsilon_{31} & -\Delta\epsilon_{32} - a & -\Delta\epsilon_{33} \end{bmatrix}$
4, 6	$\begin{bmatrix} -\Delta\epsilon_{11} & -\Delta\epsilon_{12} & -\Delta\epsilon_{13} \\ -\Delta\epsilon_{21} & -\Delta\epsilon_{22} & -\Delta\epsilon_{23} - a \\ -\Delta\epsilon_{31} & -\Delta\epsilon_{32} + \xi m & -\Delta\epsilon_{33} \end{bmatrix}$
5	$\begin{bmatrix} -\Delta\epsilon_{11} & -\Delta\epsilon_{12} & -\Delta\epsilon_{13} \\ -\Delta\epsilon_{21} & -\Delta\epsilon_{22} & -\Delta\epsilon_{23} + \xi m \\ -\Delta\epsilon_{31} & -\Delta\epsilon_{32} + \xi m & -\Delta\epsilon_{33} \end{bmatrix}$

#### 4.2.2 Reinforcement learning algorithm (RL)

The objective of the WIM nodes is to find an optimal action probability distribution, also called policy  $\pi(s, a)$ , that directs an action in a state. Policy  $\pi$  is described as a two dimensional matrix given by

$$\pi(s, a) = \begin{bmatrix} \pi(s_1, a_1) & \pi(s_1, a_2) & \pi(s_1, a_3) \\ \pi(s_2, a_1) & \pi(s_2, a_2) & \pi(s_2, a_3) \\ \dots & \dots & \dots \\ \pi(s_9, a_1) & \pi(s_9, a_2) & \pi(s_9, a_3) \end{bmatrix} \quad (7)$$

Here, the utilization of the RL algorithm is described in detail. The  $Q$ -learning algorithm is employed in Decision Agent of WIM to find optimal policies or actions. The Decision Agent chooses an action in a current state according to its action probability distribution. It selects the action with the highest probability, evaluate the effect on  $Q$ -value, summarize the  $Q$ -value of the state, and approximate its counterpart's policy. Thus, Decision Agent learns which the best action should be selected in the current state and time slot.

The RL algorithm described in *Algorithm 3* is employed in the WIM's Decision Agent. Firstly, the Decision Agent selects one of the available actions based on the current probability distribution in the current state and time slot. Then, it checks its counterpart's state. Secondly, it executes a selected action, receives a reward, and moves on to the next state. Finally, it updates its action probability distribution in the current state based on the obtained rewards and the counterpart's probability distribution.

In *Algorithm 3*, lines 3 to 21 represent a time slot. On line 1, learning rates  $\zeta$  and  $\delta$  and discount factor  $\gamma$  are initialized from 0 to 1. The WIM must select the action in the current time slot (line 4). It selects the action according to the probability distribution in the current state. If WIM selects the Warn action, it sends in-network processing request to its counterpart. The WIM then obtains a reward, moves on to the next state, and updates the  $Q$ -value (line 8).

#### Algorithm 3 In-Network Processing Decision Algorithm

```

1: Input:  $\zeta = 0.8, \delta = 0.4,$  and  $\gamma = 0.9$ ;
2: Set the  $Q$  value function of each state to 0.0 and the action probabilities  $\pi$  to the expected value in each state;
3: while  $Q$  has not converged do
4:   based on action probability in  $\pi(s, a)$ , select an action;
5:   if action = warn then
6:     the WIM request in-network processing in the counterpart's area;
7:   end
8:   obtain reward  $r$ , observe next states  $s'$ , then update  $Q(s, a) \leftarrow Q(s, a) (1 - \zeta) + \zeta(r + \gamma \max_a Q(s', a'))$ ;
9:   if my action = wake or warn then
10:     based on the current  $Q(s, a)$ , estimate the probability distribution of the neighbour WIM;
11:     according to the estimation, update the WIM's probability distribution  $\pi(s, a)$  for all actions in the current time slot and state;
12:   else
13:     calculate the average reward  $\mathfrak{R}(s) \leftarrow \sum Q(s, a) \pi(s, a)$ ;
14:     for all my actions do
15:        $\pi(s, a) \leftarrow \delta(Q(s, a) - \mathfrak{R}(s)) + \pi(s, a)$ ;
16:     end
17:   end
18:    $\pi(s) \leftarrow$  Normalize probability distribution( $\pi(s)$ );
19:    $\xi \leftarrow$  (time slot / time slot + 1) *  $\xi$ ;
20:    $s \leftarrow s'$ ;
21:   time slot++;
end

```

If the Decision Agent selects Wake or Warn, it estimates the counterpart's action probability distribution in the current time slot. According to the estimation, it updates its own probability distribution  $\pi(s, a)$  for each action in the current state. If it selects Sleep (lines 13 - 15), it updates its own probability distribution of actions  $\pi(s, a)$  for all actions based on the average reward that is calculated using  $Q$ -value times action probability (line 13). Then, the action probability distribution is updated (line 14) using the current action probability added by the gap between the  $Q$ -value and the average reward. The probabilities are normalized to obtain a valid distribution (total probability of actions is equal to 1 and each value is in 0 to 1) (line 17). The learning rate  $\zeta$  is decreased for convergence.

To estimate the counterpart's action distribution probability, the Decision Agent calculates the limit to one of the expected reward, showing the probability for infinite time and action as

$$Q_{WIM}^1(s, 1) = \lim_{\pi_1 \rightarrow 1} \sum_{x=1}^N \sum_{y=1}^N r_{xy}^1 \pi_x^1 \pi_y^2 = \sum_{y=1}^N r_{1y}^1 \pi_y^2 \quad (8)$$

$$Q_{WIM}^1(s, 2) = \lim_{\pi_2 \rightarrow 1} \sum_{x=1}^N \sum_{y=1}^N r_{xy}^1 \pi_x^1 \pi_y^2 = \sum_{y=1}^N r_{2y}^1 \pi_y^2 \quad (9)$$

$$Q_{WIM}^1(s, 3) = \lim_{\pi_3 \rightarrow 1} \sum_{x=1}^N \sum_{y=1}^N r_{xy}^1 \pi_x^1 \pi_y^2 = \sum_{y=1}^N r_{3y}^1 \pi_y^2 \quad (10)$$

Using Gaussian Elimination [35] the counterpart's action probability distributions  $\pi_1^1$  to  $\pi_3^1$  from Eqs. (8) - (10) can be estimated by WIM's Decision Agent. Then, it increases the expected reward and updates the action probability using

$$\pi_1^1(k+1) = \pi_1^1(k) + \alpha \left( \sum_{y=1}^N r_{1y}^1 \pi_y^2 \right)_k \quad (11)$$

$$\pi_2^1(k+1) = \pi_2^1(k) + \alpha \left( \sum_{y=1}^N r_{2y}^1 \pi_y^2 \right)_k \quad (12)$$

$$\pi_3^1(k+1) = \pi_3^1(k) + \alpha \left( \sum_{y=1}^N r_{3y}^1 \pi_y^2 \right)_k \quad (13)$$

where  $k \geq 0$  is the current time slot and  $\alpha > 0$  is the gradient step with a small value, e.g. 0.0001.

The probability distribution obtained from Eqs. (11) - (13) is probably not in the range of 0 to 1. Here, the normalization of the values between 0 and 1 is done using the *Softmax function* as given in *Algorithm 4*. Note that the subscript  $x$  refers to the action  $x$  of WIM-1's Decision Agent, and it is substituted by  $y$  for WIM-2.





**Algorithm 4** Normalizing probability distribution in WIM

- 1: **Input:** Given the vector of probability distribution  $\pi$ ;
- 2: **Output:** normalized probability distribution  $\pi'$ ;
- 3: set sum of probability  $\sigma = 0$ ;
- 4: create vector of  $Z$ ;
- 5: **for each probability distribution element  $\pi_x$  do**
- 6:      $z_x = e^{\pi_x}$ ;
- 7:      $\sigma = \sigma + z_x$ ;
- 8: **end for**
- 9: create vector of normalized probability distribution  $\pi'$ ;
- 10: **for each element of  $Z$  do**
- 11:      $\pi'_x = z_x / \sigma$ ;
- 12: **end for**

#### 4.2.3 Truck classifier

A neural network to classify the type of trucks passing over the bridge is employed in WIM's Classifier Agent. It classifies trucks into three categories: Type II, Type III, and Type IV as shown in Table VI. Trucks belong to those types are recommended by the Classifier Agent to request in-network processing in the counterpart's area.

TABLE VI  
VEHICLE CLASSIFICATION

I	II	III	IV
			

The neural network consists of 4 inputs, 3 outputs, 2 hidden layers with 20 neurons each, and a sigmoid activation function. Three outputs represent 001, 010, and 100 for truck types II, III, and IV respectively. The features extraction is conducted in WIM's Sensor Agent and the output data from it are sent to Classifier Agent as the input to the neural network, as can be seen in Fig. 3.

## V. THE CASE STUDY

Cisomang Underpass Bridge, Purbaleunyi Toll, West Java, Indonesia was used as the case study bridge. The dataset for in-network processing simulation in Section VI is taken from the vibration of this bridge.

The bridge was built using concrete materials of 41.5 MPa for girder boxes and 30 MPa for concrete plate. The bridge length and width are 30 m and 12.6 m, respectively.

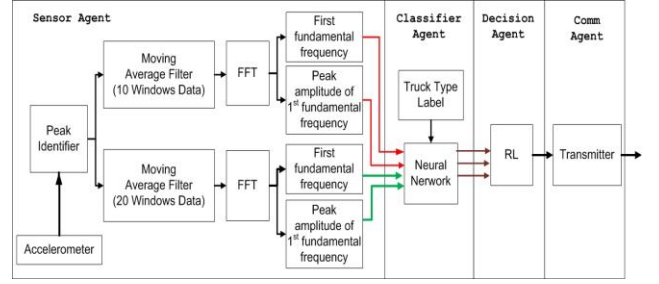


Fig. 3 WIM's agents

It comprises a concrete floor plate with thickness 0.36 m and a precast pre-stressed girder. The bridge views and properties are given in Fig.4 and Table VII.



Fig. 4 The Cisomang underpass bridge

TABLE VII  
CISOMANG UNDERPASS BRIDGE PROPERTIES

Element	Material	Properties			
		Density (kg/m <sup>3</sup> )	Modulus of Elasticity (MPa)	Quality (MPa)	Poisson's Ratio
Girder Box	Concrete	2,400	30,277	41.5	0.2
Concrete Plate	Concrete	2,400	25,741	30	0.2
Tendon	Steel	7,850	200,000	1,860	

#### 5.1 Finite element analysis of the bridge

FEA was conducted using *CSiBridge 2017* application. The bridge load test includes static load and dynamic load tests. Static load is the bridge weight, including girder boxes and concrete plate, and pre-stressed load. The FEA model of the bridge is shown in Fig. 5.

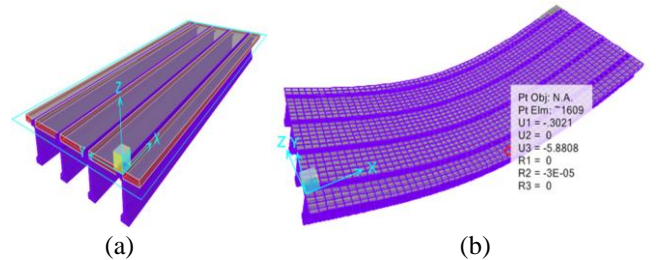


Fig. 5 The FEA of Cisomang Underpass Bridge: (a) the bridge structure and (b) deflection combining bridge's weight and pre-stressed load

From the static analysis, the middle of the bridge is able to lift its weight with maximum deflection 14.85 mm. With additional pre-stressed force, the bridge deflection is 5.88 mm. A dynamic analysis was conducted to measure the dynamic response of the bridge when a heavy moving truck is passing over. The vibration of the bridge is depicted in Fig. 6. Here, the first fundamental frequency identified in



FEA is **4.732 Hz**. This value will be used as the ideal bridge fundamental frequency,  $f_{FEA}$ , or in other words it is use as the reference for the bridge's excellent condition.

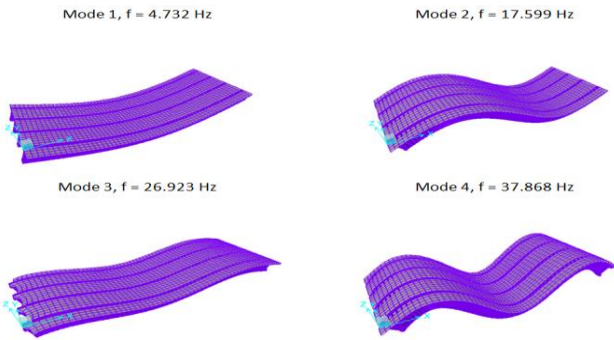


Fig. 6 Vibration data of the bridge through FEA

### 5.2 The bridge vibration dataset

To confirm the fundamental frequency measured from the FEA, the field measurement in the real bridge was conducted. Through this measurement, the vibration data for in-network processing and truck classification dataset were obtained. The vibration data when the bridge is passed over by moving heavy truck were captured using SunSPOT wireless sensor node as illustrated in Fig. 7.



Fig. 7 Bridge vibration data collection using SunSPOT node

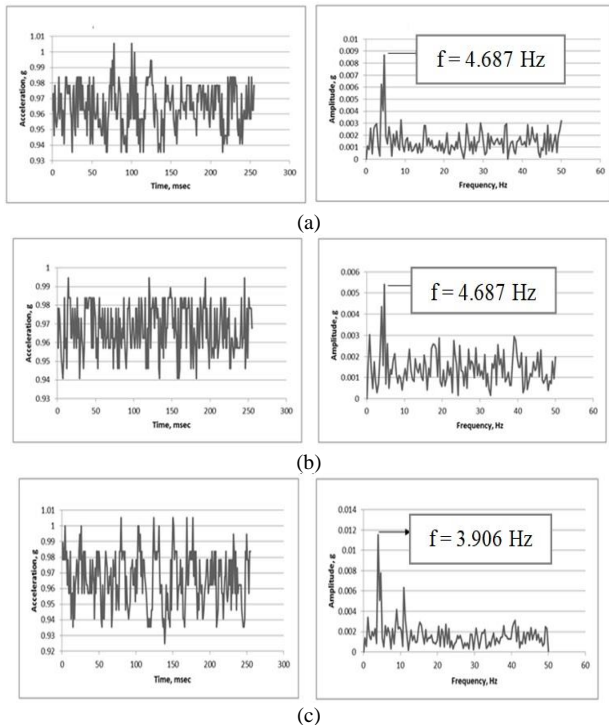


Fig. 8 The examples of Cisomang Underpass Bridge's vibration captured by SunSPOT wireless sensor nodes when it was passed over by truck: (a) Type II, (b) Type III, and (c) Type IV

The average fundamental frequency detected by our sensor node is about **4.39 Hz**. Computation using Eq. (1) yields the bridge rating of level 8, indicating that the bridge is in a very good condition. Fig. 8 shows the examples of the bridge vibration and its fundamental frequency when it was passed over by three different truck types, and Fig. 9 shows the interface displaying the bridge's rating and mode shape

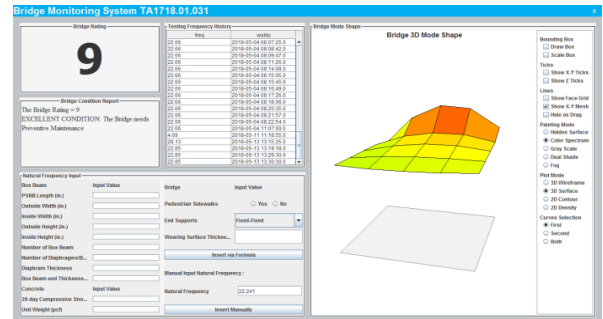


Fig. 9 Bridge rating monitoring graphical user interface

### 5.3 The truck inter-arrival time on the bridge

In order to train the two-player game and reinforcement learning for the in-network processing control, the trucks inter-arrival time or headway data are required. The density of the truck headway on the Cisomang Underpass Bridge between 09.00am and 12.00pm follows the distribution as shown in Fig. 10. The truck headway collected from Cisomang Underpass Bridge was used to confirm that the algorithm for the headway generation for simulation also follows a similar distribution.

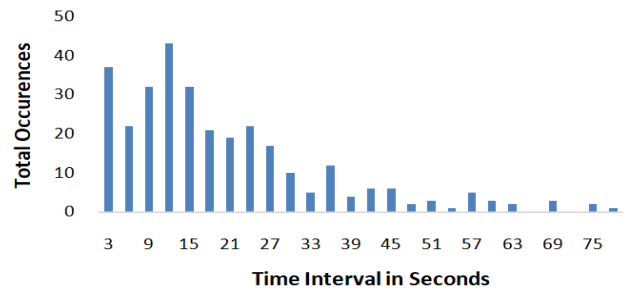


Fig. 10 Truck headway distribution on Cisomang Underpass Bridge

As can be seen in Fig. 10, the trucks inter-arrival time or headway follows exponential distribution. Hence, it can be used to model the vehicle arrival and the random process of a number of vehicles that arrive in a given time period following that distribution. Here, we model how many heavy trucks arrive in a given interval of time or what the time interval between the successive trucks is. The arrival of trucks at a section is illustrated in Fig. 11. The notations  $h1, h2, \dots$ , etc. indicate the headways with some real values, while  $t1, t2$ , and  $t3$  are time intervals.

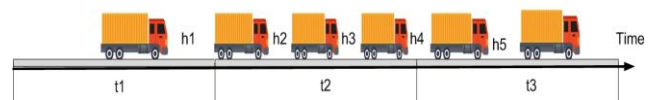


Fig. 11 The illustration of the truck arrival

Gathering headway data on real bridge is not easy task. Fortunately, trucks headway can be generated using *Algorithm 5*, which also follows an exponential distribution. Using this algorithm, the various numbers of trucks in an hour can be set to simulate different traffic scenario. Here, the algorithm was deployed on two WIMs to simulate the truck arrival detection following that distribution.

---

**Algorithm 5** Inter-arrival time or Headways Generation
 

---

- 1: Set the number of vehicle per hours or flow rate  $\hat{b}$ ;
  - 2: Calculate mean headway  $\mu = 1/\hat{b}$ ;
  - 3: Set stopping criteria;
  - 4: Set initial headway = 0;
  - 5: **while** initial headway  $\leq$  stopping criteria **do**
  - 6:     generate random value  $x$ ;
  - 7:     generate headway,  $t = \mu * (-\log(x))$ ;
  - 8:     sleep( $t * 1000$ );
  - 9:     send warning message to Decision Agent of WIM
  - 10:    initial headway = initial headway +  $t$ ;
  - 11: **end while**
- 

## VI. SIMULATION AND EXPERIMENTAL RESULTS

To overcome the difficulties of system testing in the real bridge, the development of simulation environment in computer was carried out. SunSPOT Solarium Emulator was used for simulation environment as illustrated in Fig. 12. The description of the SunSPOT sensor nodes and SunSPOT Solarium Emulator can be found in [36] and [37]. The Java codes implementing our proposed method were deployed on each sensor node. These Java codes can be installed on real SunSPOT sensor nodes directly. The emulator role is also very important for code debugging and application testing.

As can be seen in Fig. 12, two nodes acting as the WIM run *Algorithm 5* as if they detect truck arrival on the bridge and execute the two-player game reinforcement learning while the nodes in each cluster execute in-network processing. It was assumed that the WIMs were installed on each span of the two-span bridge and each span has the same property as the Cisomang Underpass Bridge. Our simulation objectives were to prove that in-network processing algorithm run in each WSN cluster successfully and the two-player game and reinforcement learning works properly, in the sense that two WIMs are able to adjust their action probabilities and approximate their counterpart's action probability distribution for each state in a time slot. The experiment in real SunSPOT sensor nodes was also conducted to study the processing time and the effect of experiment when implementing our proposed work.

### 6.1 In-network processing simulation

In-network processing in different number of sensor nodes was tested, in which the signal strength and the remaining battery level weights were set to 0.4 and 0.6 respectively. Here, processing time such as FFT and peak picking, mobile agent migration plan, mobile agent migration, and the whole in-network processing that runs both in emulator (simulation) and real SunSPOT nodes (experiment) were measured. The ACC nodes in each cluster were tested, performing 100 Hz sampling frequency and capturing 512 data. These processing time are shown in Fig. 13.

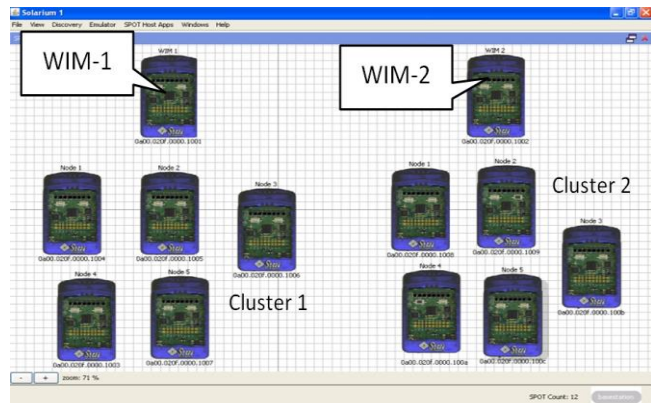


Fig. 12 SunSPOT Solarium emulator

In-network processing time refers to time duration required by a WSN to complete the process starting from registration, capturing vibration data by accelerometer sensor, until receiving pre-processing data by sink node. Mobile agent migration time refers to duration time required by mobile agent to travel hop by hop from the first node to the last node. As can be seen in Fig. 13, the average in-network processing time using six SunSPOT nodes is 13 seconds and mobile agent migration time is 9.4 seconds.

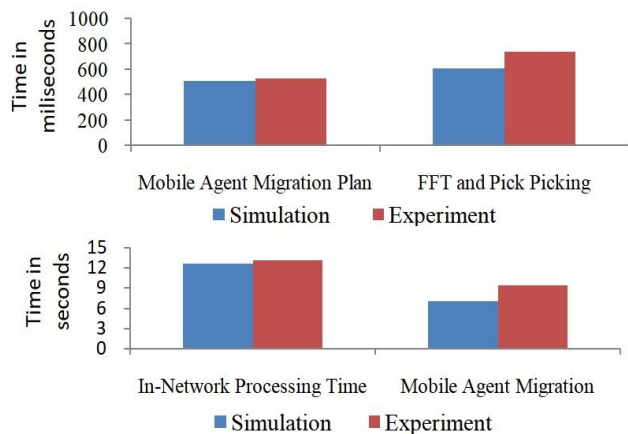


Fig. 13 The average processing time

### 6.2 Simulation of in-network processing scheduling

The WIM node interactions using the two-player game and reinforcement learning were also simulated using the SunSPOT Solarium emulator. In this simulation, it has been proven whether each WIM is able to estimate the counterpart's probability distribution of actions for each time slot and the WIM nodes considered that only a single heavy truck passing on the bridge as the best moment to request in-network processing.

The most interesting states being analyzed were the pairs of states number 2 and number 4 or states number 4 and number 2. These states are the ideal condition to execute in-network processing. Fig. 14a shows that the  $Q$ -value of WIM-1 when performing action Wake in the state No truck vs. One truck (State 2) and the  $Q$ -value of WIM-2 when performing action Warn in the state One truck vs. No truck (State 4) are the highest  $Q$ -value out of their available actions. In this situation, WIM-1 accepts WIM-2's request to execute in-network processing

in the WIM-1 area. The optimal policies of both players are described in Fig. 14b. When a WIM chooses action *Wake*, it approximates its counterpart's probability distribution and based on it a WIM updates its own probability distribution. The approximation will converge to the counterpart's probability distribution.

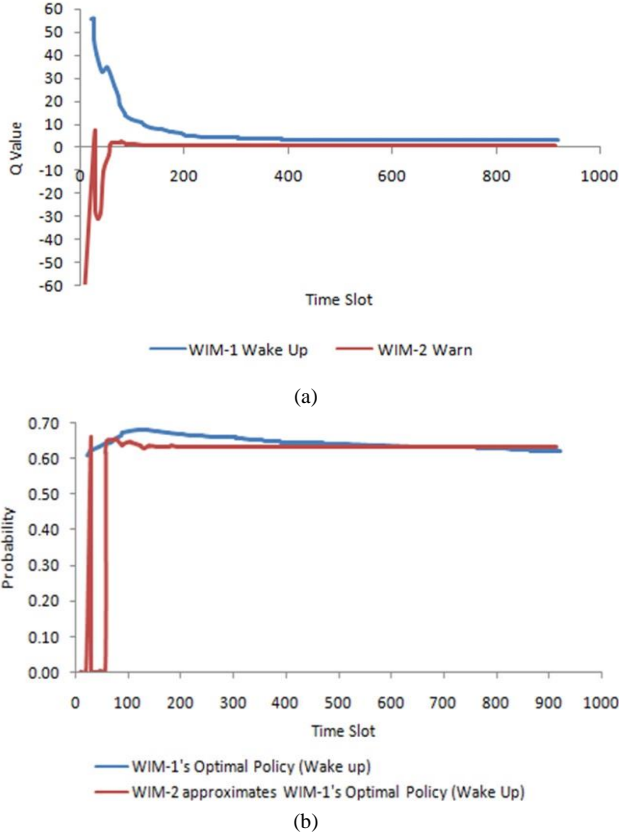


Fig. 14(a) WIM-1's  $Q$ -value in State 2 and WIM-2's  $Q$ -Value in State 4 and (b) WIM-1 approximates WIM-2

Let's look at the state number 4 from the perspective of WIM-1 or the state number 2 from the perspective of WIM-2. Here, WIM-1 detects One truck while WIM-2 detects No truck. The  $Q$ -value of both WIMs is shown in Fig. 15a. The  $Q$ -value of WIM-1 when performing action *Warn* in the state number 4 and the  $Q$ -value of WIM-2 when performing action *Wake* in the state number 2 are the highest  $Q$ -values for the available actions. It can be concluded that WIM-2 accepts WIM-1's request to execute in-network processing in WIM-2's WSN area. Fig. 15b shows how they approximate counterpart's probabilities.

The optimal policies for other states are displayed in Fig. 16. Here, we summarize the probability density of each action at the end of the learning time period. Although the occurrence of states 1, 3, 6, 7, 8, and 9 are rare, the WIMs are still able to approximate the current WIM probability distribution. For example, we see the state 6 in Fig. 16a and the state 8 in Fig. 16b. In this case, WIM-1 detects One truck and WIM-2 detects Many trucks. The optimal policy of WIM-1 is *Warn* and it approximates WIM-2's optimal policy as *Wake*. At the same time, the optimal policy of WIM-2 is *Wake* and it approximates *Warn* for WIM-1. In this case, WIM-2 accepts WIM-1's request to execute the in-network processing in WIM-2's WSN area.

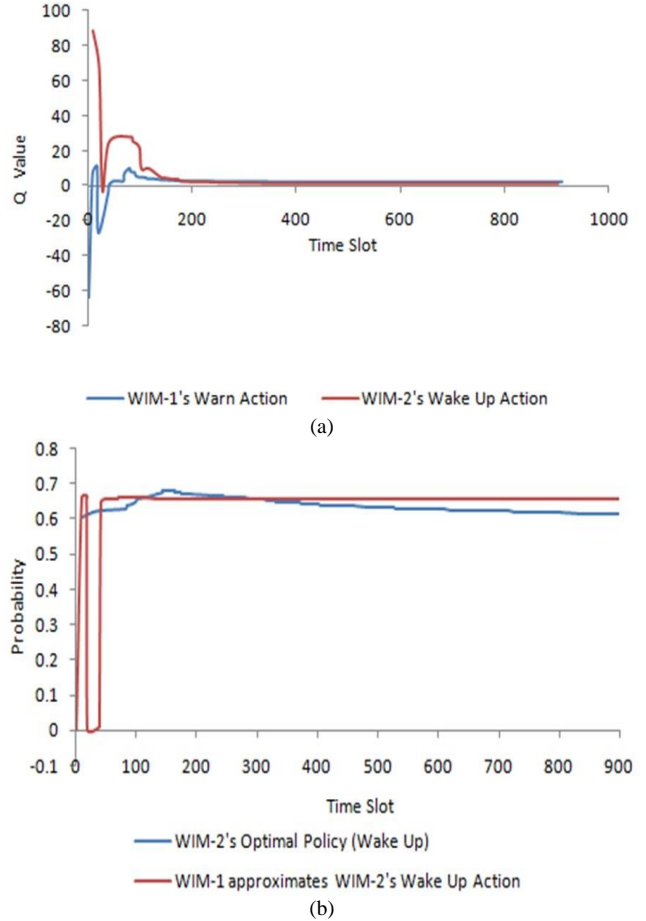


Fig. 15 (a) WIM-1's  $Q$ -value in State 4 and WIM-2's  $Q$ -Value in State 2 and (b) WIM-1 approximates WIM-2.

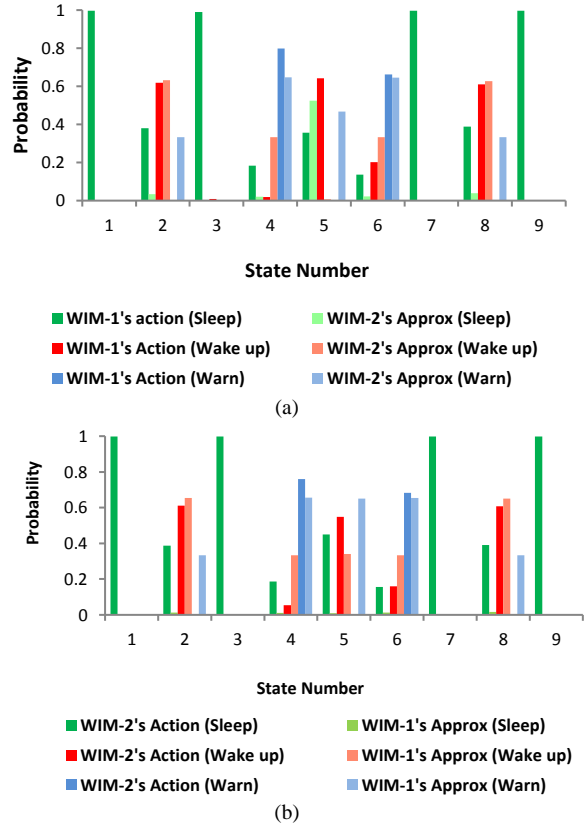


Fig. 16 Probability distribution for all states in the Decision Agent of (a) WIM-1 and (b) WIM-2

As can be seen in Fig. 16, the in-network processing was executed in the state 8 in WIM-1's perspective and the state 6 in WIM-2's perspective. If WIM chooses action `Sleep`, then it does not approximate its counterpart's distribution probability, but only updates its own distribution. Thus, states 1, 3, 7, and 9 are expected to be in `Sleep` mode.

Only in state 5 both WIMs cannot approximate their counterpart's probability distribution correctly. It is called a conflict between two WIMs in which both WIMs select the same action. For example, both WIMs select action `Wake` and they expect their counterpart to select action `Warn`. In another situation both WIMs could select action `Warn` and expect their counterpart to select action `Wake`. Therefore, when both WIMs in state 5, it cannot be predicted as well as other states. However, in-network processing probably could be executed in this state if one of the WIMs selects action `Warn` and another selects action `Wake`.

### 6.3 Simulation of truck classification

This subsection describes the trucks classification. Training dataset obtained from Cisomang Underpass Bridge's vibration was used to find the optimal neural network architecture. The variations of the neural network's total hidden layer, neuron in each layer, and learning rate are examined to find the most optimal configuration. We found that the configuration with 4 inputs, 2 hidden layers, 20 neurons for each hidden layer, 3 outputs, and learning rate 0.01 is the best as shown in Table VIII.

The classifier was then tested to make sure that it was able to classify the truck type correctly. The Java code implementing the WIM's `Classifier Agent` was deployed to the virtual SunSPOT sensor node of the Solarium Emulator. Here, the test dataset, which is outside the training dataset, was tested and the 82.6% of accuracy was achieved.

TABLE VIII  
THE NEURAL NETWORK'S CONFIGURATIONS

No	Hidden Layer	Total Neuron	Learning Rate	Accuracy	MSE
1	2	2, 2	0.01	51.20%	0.0971
2	2	5, 5	0.01	63.63%	0.0774
3	2	10, 10	0.01	79.90%	0.0453
4	2	15, 15	0.01	87.60%	0.0276
5	2	20, 20	0.01	95.69%	0.0079
6	2	35, 35	0.01	94.73%	0.012
7	3	5, 5, 5	0.01	63.63%	0.0735
8	3	10, 10, 10	0.01	82.29%	0.0358
9	3	20, 20, 20	0.01	93.33%	0.0152
10	2	20, 20	0.1	74.64%	0.0502
11	2	20, 20	0.001	77.03%	0.0561

### 6.4 Comparison with related works

The first performance analysis is the energy consumption. Here, the energy consumption in an hour is examined using SunSPOT sensor nodes that have a built-in lithium battery 720 mA-hours and the voltage 5 volt. Thus, the battery energy in an hour is 12,960 J. When the nodes are in mode `awake`, actively perform calculation and the radio communication is on, the average current draw is about 104 mA. When the nodes are in mode `sleep`, no computation activity while the radio communication is still on, the current draw is 46 mA [38].

With the first approach [14], the energy consumption for sensing and transmitting acceleration data of bridge vibration with 100 Hz sampling frequency is the same as the node when it is in mode `awake` in  $t$  seconds. Thus, energy consumption per hour using this approach is  $E = v. i. t = 5 \text{ volt} \times 104 \text{ mA} \times 3,600 \text{ second} \approx 1,872 \text{ J}$ .

Applying the second approach [5], FFT analysis and peak picking are always executed if a WIM detects a truck passing over the bridge without considering when sensing and data processing are at the best moment. Here we simulated the truck headway with flow rate 360 trucks per hours. The headway that was generated by *Algorithm 5* produced total truck occurrences of 42, 161, and 37 for *n truck*, *one truck*, and *many trucks* respectively, during the first hour. The WIM has to observe the state of the bridge every 15 seconds for each time slot. We found that 5.965 seconds were spent by sensor nodes to perform sensing of 512 data with 100 Hz sampling, FFT and peak picking, and transmit data to the sink node. Otherwise, they were in `sleep` mode. The energy consumption using this approach was  $E = ((161 + 37) \times 5.965 \text{ s} \times 5 \text{ volt} \times 104 \text{ mA}) + ((3,600 \text{ s} - ((161 + 37) \times 5.965 \text{ s})) \times 5 \text{ volt} \times 46 \text{ mA}) \approx 1,170.5 \text{ J}$ .

Using the third, which is our proposed approach, in-network processing is executed when a WIM chooses the action `Wake` while the other chooses the action `Warn`. According to the truck arrivals simulation implemented to the second approach, the total state occurrences in an hour when a WIM selected action `Wake` and the other selected `Warn` was 55 times, or in-network processing was executed 55 times. Otherwise, all ACC nodes were in `Sleep` mode. As described in Fig. 13, the average duration for in-network processing was 13 seconds, thus the total energy consumed in this situation was  $E = (55 \times 13 \text{ s} \times 5 \text{ volt} \times 104 \text{ mA}) + ((3600 \text{ s} - (55 \times 13 \text{ s})) \times 5 \text{ volt} \times 46 \text{ mA}) \approx 1,035.36 \text{ J}$ .

Comparing the three approaches, we can confirm that our proposed approach requires the lowest total energy consumption. Fig. 17 shows the energy level reduction at an ACC sensor node that performs in-network processing in an hour when using three different approaches. It can be seen that the propose method shows the best performance in saving battery energy when the traffic on the bridge is 360 truck per hours. In this case, the time ratio when a node in wake and sleep mode is 1:4. The lower the truck flow rate, the more energy will be saved, as the optimal sensor node's policy will choose the sleep mode more often.

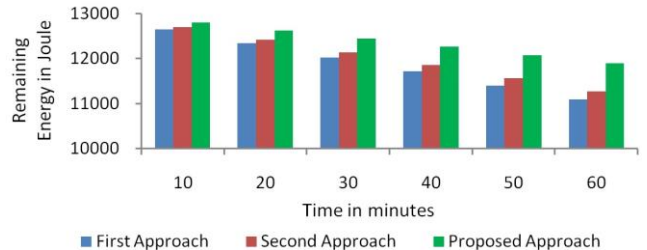


Fig.17 Energy consumption in an hour when running three approaches

The second performance analysis is to examine the processing time using six SunSPOT sensor nodes. The processing time refers to the time required by the system to complete the bridge rating assessment, from sensing the

vibration data until the fundamental frequency is converted to the value of the bridge rating in the sink node. The processing time of the three approaches is shown in Fig. 18.

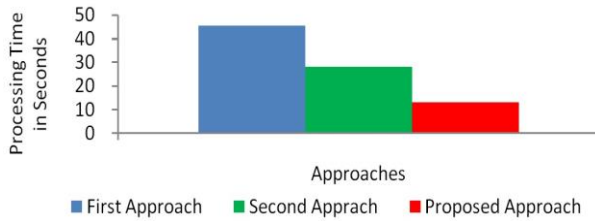


Fig. 18 Method comparison in term of processing time

As can be seen in Fig. 18, the first approach requires the longest time. In the first approach all sensor nodes simultaneously transmit the raw data of bridge vibrations to the sink node. FFT, peak picking, and bridge rating calculation are conducted in the sink node. The problem found with this approach lies in the base station node that is connected via a cable to the sink node. The base station node is the communication gateway where all messages and data packets from all sensor nodes are passed before entering the sink node. The drawback of this base station is the inability to handle the data reception from all sensor nodes simultaneously so that the queues of handling messages or data are carried out in the base station's buffer. Therefore, it causes the time delay and scalability issues. In the experiment, this approach works with four nodes only.

To overcome the problem of the first approach, the second approach was proposed. In this approach, data processing such as FFT processing and peak picking are conducted in each sensor node. However, the data reception that causes the delay in base station node is still not solved with this approach, as the base station node still has to handle simultaneous data reception from all sensor nodes.

Our proposed approach attempts to minimize the time delay in base station by preventing each sensor node to send data simultaneously. Here the mobile agent is dispatched hop by hop from the first node to the last node and then it transmits a single data packet wrapping all sensor nodes data to the sink node. As can be seen in Fig. 18, this approach reduces the total processing time.

## VII. CONCLUSIONS

This paper describes multiagent system employment in WSNs for bridge condition assessment using the dynamic response in which a bridge's fundamental frequency is measured. The main focus was the development of an autonomous system performing in-network processing. The issues such as large energy consumption to execute in-network processing and time delay are taken into account by control mechanism in the process. The process should be conducted only if a heavy vehicle passes over the bridge.

When making a decision about when the wireless sensor network should perform in-network processing, there is a trade-off between keeping the WSN resources alive as long as possible, by optimizing the sleeping time, and capturing significant events, by optimizing the waking time. Two-player game and reinforcement learning have been proposed to calculate the optimal policy and adjust the best action probability distribution in a state over a certain time period.

This process is conducted by agents deployed on two WIMs and includes a WIM sending warning message to its counterpart for in-network processing execution, approximating the counterpart's probability distribution and updating its own probability distribution. The ideal states expected or recommended for in-network processing are the state when WIMs are under the states number 2 and number 4 or under states number 4 and number 2. Simulation results show the *effectiveness* of the proposed method in which both WIMs can approximate their counterpart's probability distribution and update their probability distribution properly.

Based on simulation and experimental results, the strategy proposed in this paper shows the best results in terms of energy consumption and processing time, compared to previous similar works. This confirms the *efficiency* of the proposed method. In addition, the proposed system has implemented the system autonomy.

To overcome the difficulties when testing the system on a real bridge, the development of laboratory-based test-bed bridge is necessary. Using a test-bed bridge the simulation of several damage scenarios and intelligent sensing algorithms can be conducted in a more flexible way.

While this paper focuses on the bridge rating as the main parameter to determine, the determination of other important parameters such as the bridge capacity, which has not been considered, the use of additional sensors such as piezoelectric sensors, will be the subject of the future work. More testing and measurement in real bridge environment allowing various vehicles and traffic scenarios will also be done to further validate the results.

## REFERENCES

- [1] S. K. U. Rehman, Z. Ibrahim, S. A. Memon, M. Jameel, "Nondestructive Test Methods for Concrete Bridges: A Review", *Construction and Building Materials*, vol. 107, pp. 58-86, 2016.
- [2] Z. Zou, Y. Bao, F. Deng, and H. Li, "An Approach of Reliable Data Transmission With Random Redundancy for Wireless Sensors in Structural Health Monitoring", *IEEE Sensors Journal*, vol. 15, no. 2, pp. 809-818, 2015.
- [3] C. Tschope and M. Wolff, "Statistical Classifiers for Structural Health Monitoring", *IEEE Sensors Journal*, pp. 1567-1576, 2009.
- [4] S. Kim, J. Lee, M.-S. park, and B.-W. Jo, "Vehicle Signal Analysis Using Artificial Neural Networks for a Bridge Weigh-in-Motion System", *Sensors*, vol. 9, pp. 7943-7956, Oct. 2009.
- [5] P. Guo, J. Cao, and X. Liu, "Lossless In-Network Processing in WSNs for Domain-Specific Monitoring Applications", *IEEE Trans. Industrial Informatics*, vol. 13, no. 5, pp. 2130-2139, 2017.
- [6] Z. Alam, G. Wang, J. Cao, and J. Wu, "Deploying Wireless Sensor Network with Fault-Tolerance for Structural Health Monitoring", *IEEE Transaction on Computer*, pp. 382-395, 2015.
- [7] A. B. Noe, A. Abdaoui, T. Elfouly, M. H. Ahmed, and A. Badawy, "Structural Health Monitoring Using Wireless Sensor Networks: A Comprehensive Survey", *IEEE Communication Surveys and Tutorials*, vol. 19, no. 3, pp. 1403-1423, 2017.
- [8] A. Araujo, J. García-Palacios, J. Blesa, F. Tirado, and E. Romero, "Wireless Measurement System for Structural Health Monitoring With High Time Synchronization Accuracy", *IEEE Trans. Instrumentation and Measurement*, vol. 61, no. 2, pp. 801-810, 2012.
- [9] D. Mascareñas, E. Flynn, C. Farrar, G. Park, and M. Todd, "A Mobile Host Approach for Wireless Powering and Interrogation of Structural Health Monitoring Sensor Networks", *IEEE Sensors Journal*, vol. 9, no. 12, pp. 1719-1726, Dec. 2009.
- [10] E. Sazonov, H. Li, D. Curry, and P. Pillay, "Self-powered Sensor for Monitoring of Highway Bridge", *IEEE Sensor Journal*, pp. 1422-

1429, 2009.

- [11] M. Lydon, S. E. Taylor, D. Robinson, P. Callender, and C. Doherty, "Development of a Bridge Weighted-in-Motion Sensor: Performance Comparison Using Fiber Optic and Electric Resistance Strain Sensor System", *IEEE Sensors Journal*, pp. 4284-4296, 2014.
- [12] R. Bajwa, "Wireless Weigh-in-Motion: Using Road Vibration to Estimate Truck Weights", PhD Thesis, Electrical Engineering and Computer Sciences University of California, Berkeley, 2013.
- [13] C. J. A. Tokognon, B. Gao, G. Y. Tian, and Y. Yan, "Structural Health Monitoring Framework Based on Internet of Things: A Survey", *IEEE IoT Journal*, vol. 4, no. 4, pp. 619-635, 2017.
- [14] A. A. Islam, F. Li, H. Hamid, and A. Jaroo, "Bridge Condition Assessment and Load Rating Using Dynamic Response", YoungsTown State University, Ohio, Final Report 134695, 2014.
- [15] T. Nanayakkara, M. N. Halgamuge, P. Sridhar, and A. M. Madni, "Intelligent Sensing in Dynamic Environments Using Markov Decision Process", *Sensors*, vol. 11, pp. 1229-1242, Jan. 2011.
- [16] K. H. S. Hla, Y. Choi, and J. S. Park, "The Multiagent System Solutions for Wireless Sensor Network Applications", in *KES International Symposium on Agent and Multi-Agent Systems: Technologies and Applications*, Incheon, Korea, 2008.
- [17] R. Tynan, G. M. P. O'Hare, D. Marsh, and D. O'Kane, "Multiagent System Architectures for Wireless Sensor Networks", in *5th International Conference*, Atlanta, GA, USA, 2005.
- [18] M. Guijarro, R. Fuentes-Fernandez, and G. Pajares, "A Multiagent System Architecture for Sensor Network", in *Multiagent Systems Modelling, Control, Programming, Simulations, and Applications*. Croatia: InTech, 2011, pp. 23-40.
- [19] A. Montoya, D. C. Restrepo, and D. A. Ovalle, "Artificial Intelligence for Wireless Sensor Network Enhancement", *Smart Wireless Sensor Networks*, pp. 73-81, 2010.
- [20] P. Gil, A. Santos, and A. C., "Dealing with Outliers in Wireless Sensor Networks: an Oil Refinery Application", *IEEE Transactions on Control Systems Technology*, vol. 22, no. 4, pp. 1589-1596, 2014.
- [21] F. Aiello, A. Carbone, G. Fortino, and S. Galzarano, "Java-based Mobile Agent Platform for Wireless Network Sensor", in *The International Multiconference on Computer Science and Information Technology*, 2010, pp. 165-172.
- [22] M. Chen, "Mobile Agent-based Wireless Sensor Network", *Journal Computing*, vol. 1, no. 1, pp. 14-21, 2006.
- [23] M. Chen, S. Gonzalez, and V. C. M. Leung, "Application and Design Issues for Mobile Agent in Wireless Sensor Networks", *IEEE Wireless Communication*, vol. 14, no. 6, pp. 20-26, 2007.
- [24] M. Chen, L. T. Yang, T. Kwon, L. Zhaou, and M. Jo, "Itinerary Planning for Energy Efficient Agent Communication in Wireless Sensor Network", *IEEE Transactions on Vehicular Technology*, vol. 60, no. 7, pp. 3290-3299, 2011.
- [25] C. Konstantopoulos, A. Mpitziopoulos, D. Gavalas, and G. Pantziou, "Effective Determination of Mobile Agent Itineraries for Data Aggregation on Sensor Networks", *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 12, pp. 1679-1693, Dec. 2010.
- [26] X. Wang, M. Chen, T. Kwon, and H. C. Chao, "Multiple Mobile Agents' Itinerary Planning in Wireless Sensor Networks: Survey and Evaluation", *IET Communicat.*, vol. 5, no. 12, pp. 1769-1776, 2011.
- [27] Q. Wu, "On Computing Mobile Agent Routes for Data Fusion in Distributed Sensor Networks", *IEEE Transaction Knowledge and Data Engineering*, vol. 16, no. 6, pp. 740-753, 2004.
- [28] M. Sharawi, I. A. Saroit, H. El-Mahdy, and E. Emary, "Routing Wireless Sensor Networks Based on Soft Computing Paradigms: Survey", *International Journal on Soft Computing, Artificial Intelligence and Applications*, vol. 2, no. 4, pp. 21-36, 2013.
- [29] R. V. Kulkarni, F. Anna, and G. K. Venayagamoorthy, "Computational Intelligence in Wireless Sensor Networks: A Survey," *IEEE Comm. Surveys & Tutorials*, vol. 13, no. 1, pp. 68-96, 2011.
- [30] M. Wooldridge, *An Introduction to Multiagent Systems*, 2nd ed. Chichester, United Kingdom: John Wiley and Sons Ltd, 2009.
- [31] S. A. Putra, B. R. Trilaksono, A. Harsoyo, and A. Imam, "Agent-based Structural Health Monitoring System on Single Degree of Freedom Bridge: A Preliminary Study", *International Conference on Information Technology Systems and Innovation*, Bandung, 2015.
- [32] S. A. Putra, B. R. Trilaksono, A. Harsoyo, and A. I. Kistijantoro, "Multiagent System In-network Processing in Wireless Sensor Network", *International Journal on Electrical Engineering and Informatics*, vol. 10, no. 1, pp. 94-107, Mar. 2018.
- [33] D. Ye and M. Zhang, "A Self-Adaptive Sleep/Wake-Up Scheduling Approach for Wireless Sensor Network", *IEEE Transactions on Cybernetics*, vol. 48, no. 3, pp. 1-14, Mar. 2017.
- [34] N. A. Pantazis, S. A. Nikolidakis, and D. D. Vergados, "Energy-Efficient Routing Protocol in Wireless Sensor Network: A Survey", *IEEE Comm. Surveys and Tutorials*, vol. 15, no. 2, pp. 551-591, 2013.
- [35] A. K. a. E. E. Kalu, "Gaussian Elimination", *Numerical Methods with Applications*. South Florida: autarkaw, 2010, ch. 04.06.
- [36] H.-d. Shih, X. Zhang, D. S. L. Wei, K. Naik, and R.-C. Chen, "Design and Implementation of Mobile Sensor Network Test-bed Using SunSPOTs", *International Journal of Future Computer and Communication*, vol. 2, no. 2, pp. 115-120, 2013.
- [37] R. Lopes and F. Assis, "MASPOT: A Mobile Agent System for SunSPOT", in *Tenth International Symposium on Autonomous Decentralized Systems*, Tokyo and Hiroshima, 2011.
- [38] S. Labs. (2010, Nov.) Sun Spot World. [Online]. [www.sunspotdev.org](http://www.sunspotdev.org)



**Seno Adi Putra** obtained a Bachelor degree in Physics Science and a Masters degree in Electrical Engineering from the Institute of Technology Bandung (ITB), Indonesia. Currently, he is a PhD student in the same institution. His research interest includes wireless sensor network, intelligent and multiagent system, and machine learning. .



**Bambang Riyanto Trilaksono** obtained his first degree in Electrical Engineering from Electrical Engineering in the Institute of Technology Bandung (ITB), and a Masters and PhD degree from Electrical Engineering, Waseda University, Japan. He is currently a Professor in Control and Computer System Research Group in ITB. His research interest includes robust control, intelligent control & intelligent systems, control applications, robotics, and embedded control systems.



**Muhammad Riyansyah** obtained his Bachelor and Masters degrees in Civil Engineering from the Institute of Technology Bandung (ITB), Indonesia. He obtained his PhD degree in Civil Engineering, from National University of Singapore. Currently, he is a Lecturer within the Structural Engineering Group, Faculty of Civil and Environmental Engineering, ITB, Indonesia.



**Dina Shona Laila** obtained her BEng and MEng degrees from ITB, Indonesia, and her PhD degree in Control Engineering from the University of Melbourne, Australia. Currently, she is a Reader (Associate Professor) in Control and Instrumentation, at the Faculty of Engineering, Computing and Environment, Coventry University, UK. Her research interest includes nonlinear control and embedded control systems for various applications.



**Agung Harsoyo** obtained his BEng degree in Electrical Engineering from ITB, a Masters degree in Telecommunication Engineering from ENSTB, France and a PhD degree in Telecommunication Engineering from UBS. He is a researcher in Control and Computer System Research Group ITB. His research interests include wireless sensor network and control system.



**Achmad Imam Kistijantoro** obtained his BEng degree in Informatics from ITB and a Masters degree from TU Delft, Netherlands. He obtained his PhD from the University of Newcastle upon Tyne, UK. His research interests includes distributed system, parallel computation, and high performance computation.