International Journal of Informatics Society

12/16 Vol. 8 No.3 ISSN 1883-4566

Informatics Society

Aims and Scope

The purpose of this journal is to provide an open forum to publish high quality research papers in the areas of informatics and related fields to promote the exchange of research ideas, experiences and results.

Informatics is the systematic study of Information and the application of research methods to study Information systems and services. It deals primarily with human aspects of information, such as its qu ality and value as a resource. Informatics also referred to as Information science, studies t he structure, algorithms, behavior, and interactions of natural and a rtificial systems that store, process, access and communicate information. It also develops its own conceptual and theoretical foundations and utilizes foundations developed in other fields. The advent of computers, its ubiquity and ease to use has led to th e study of info rmatics that has computational, cognitive and social aspects, including study of the social impact of information technologies.

The characteristic of informatics' context is amalgamation of technologies. For creating an informatics product, it is necessary to integrate many technologies, such as mathematics, linguistics, engineering and other emerging new fields.

Guest Editor's Message

Yoshia Saito

Guest Editor of Twenty-fourth Issue of International Journal of Informatics Society

We are delighted to have the twenty-fourth issue of the International Journal of Informatics Society (IJIS) published. This issue includes selected papers from the Ninth International Workshop on Informatics (IWIN2015), which was held at Amsterdam, Netherlands, Sep. 6-9, 2015. The workshop was the ninth event for the Informatics Society, and was intended to bring together researchers and practitioners to share and exchange their experiences, discuss challenges and present original ideas in all aspects of informatics and computer networks. In the workshop 24 papers were presented in five technical sessions. The workshop was successfully finished with precious experiences provided to the participants. It highlighted the latest research results in the area of informatics and its applications that include networking, mobile ubiquitous systems, data analytics, business systems, education systems, design methodology, intelligent systems, groupware and social systems.

Each paper submitted IWIN2015 was reviewed in terms of technical content, scientific rigor, novelty, originality and quality of presentation by at least two reviewers. Through those reviews 14 papers were selected for publication candidates of IJIS Journal, and they were further reviewed as a Journal paper. This volume includes five papers among the accepted papers, which have been improved through the workshop discussion and the reviewers' comments.

We publish the journal in print as well as in an electronic form over the Internet. We hope that the issue would be of interest to many researchers as well as engineers and practitioners over the world.

Yoshia Saito received his Ph.D. degree from Shizuoka University, Japan, in 2006. He had been an expert researcher of National Institute of Information and Communications Technology (NICT) from 2004 to 2007, Yokosuka, Japan. He was a lecturer from 2007 to 2011 at Iwate Prefectural University and He is currently an associate professor at the University. His research interests include computer networks and Internet broadcasting. He is a member of IPSJ, ACM, and IEEE.

Real-time and Seamless WYSIWYAS Navigation System for Smart Device

Yusuke Takatori [*], Shohei Iwasaki [**], Tatsuya Henmi [**], and Hideya Takeo [*]
[*, **]Kanagawa Institute of Technology, Japan
[*]{takatori,takeo}@ele.kanagawa-it.ac.jp, [**]{s1112063,1112040}@ccy.kanagawa-it.ac.jp

Abstract - This paper proposes a real-time and seamless WYSIWYAS (RS-WYSIWYAS) navigation system for smart devices, which is to be used by a pedestrian while walking on a corridor. WYSIWYAS is a navigation concept that provides a user an intuitive guidance that do not need to interpret provided guidance information. The proposed system is implemented as a prototype application of RS-WYSIWYAS navigation system for Android smart device that has a rear camera. To provide WYSIWYAS navigation for the user, a smart device captures consecutive some M-CubITS marker elements assigned to one line by M-sequence on a corridor from a scenic movie. Then the application obtains the location of the captured marker elements in a building and the heading of the smart device. After that, it shows the user the direction to his/her destination in real time and seamlessly. Moreover, in order for the prototype application to become practical, its response and the performance of marker elements recognition are improved. The response of the application is improved by resizing of the captured movie and using native code programming. At the same time, marker recognition performance is improved by pre-masking of a captured movie. The results of experiments indicate that the response and marker recognition performance are improved to the level that a user could accept for practical use.

Keywords: ITS, Indoor navigation system, WYSIWYAS navigation, M-CubITS

1 INTRODUCTION

Because of the spread of smart devices, the demand for pedestrian navigation systems has increased. Most smart devices use GPS for positioning. However, if these devices are used inside a building or underground areas where an electric wave of the GPS does not reach, it is difficult for such devices to pinpoint their location. In addition, it is known that GPS precision and accuracy decreases if a smart device is used in areas dominated by tall buildings [1]. For these reasons, pedestrian navigation systems not dependent on GPS have attracted research attention. In around 2000, many indoor navigation systems have been developed. Indoor navigation system using wireless technology such as radio wave [2], ultrasonic wave [3], [4] or infrared ray [5] uses position of a transmitter. However these system need installation of transmitters and the user needs a dedicated receiver. Recently, a positioning technique using an electric wave emitted by Wi-Fi access points attracts attention [6]. Though these technique are available with the smartphone which is equipped with Wi-Fi device as standard equipment, some problems are left in the precision for this system because the RSSI (Received Signal Strength Indicator) fingerprinting observed by a receiver often includes some vagueness and an estimated position may not be stable. On the other hand, many positioning technologies that recognize a tag located indoors and pinpoint the location of the terminal have been conducted. Tag-based pedestrian navigation system using ID-tag like RFID or two dimension code have been studied [7]-[10]. In most cases of tag-based navigation system, tags are input their location information, then a mobile device reads these information and recognizes its location, and the acquired location information is pointed on a 2D map. However, the guidance is conducted only at tag-installed places and uses 2D map. That is, the user have to look for the ID-tag and understand his/her location on a 2D map. Besides he has to judge the way to go. Because the operation load of the tag-based navigation systems on a user is high, tag-based 2D navigation system is difficult for persons who cannot read a 2D map well. To realize a more intuitive navigation system, Kurihara proposed an indoor navigation system using augmented reality (AR) marker [11]. The AR marker proposed by Kato [12] is a two-dimensional symbol that allows a digital camera to determine its position and rotation relative to the surface of the marker. A smart device with camera reads the AR marker and overlays an arrow on the AR marker shown in the captured frame. Therefore, this navigation system does not need to interpret a two-dimensional map. However, its navigation concept is still a tag-based navigation. Like fixed guidepost, AR markers are located at each turning point. Therefore, similar to the conventional tag-based navigation systems, a user must perceive the position of the AR markers during use. To reach a destination, the user must first locate the nearest AR marker.

Furthermore, Hasegawa proposed a navigation concept called "WYSIWYAS" (What you see is what you are suggested) [13]-[15]. WYSIWYAS is a fundamental design concept of human-machine interface (HMI) for personalized and intuitive navigation. WYSIWYAS navigation is designed to provide intuitive navigation information to the user without awareness of navigation infrastructures. Then, it overlays a direction arrow to the destination on a scenic image that is captured by the camera device. Additionally, Hasegawa proposed the positioning system M-CubITS, which allows a mobile device with a camera function to determine its position and orientation. This system uses multimodal markers with 1-bit information (0 or 1), which are disposed in a line along a passage in accordance with an m-sequence. When a user takes a picture of a passage that includes a number of disposed M-CubITS markers, the bit sequence of markers is extracted. At this time, the user do not have to consider the arrangement or the position of the markers. Following this, the extracted bit sequence is checked against a database, and the position and the orientation of the mobile device is obtained. By applying the M-CubITS positioning system to the WYSIWYAS navigation concept, intuitive

navigation without awareness of navigation infrastructures is achieved. Yamashita and Manabe have built pedestrian navigation systems based on the "WYSIWYAS navigation concept with M-CubITS positioning system" on a mobile phone with a camera device [16]-[18]. Masuda has built navigation systems based on the "WYSIWYAS navigation concept with M-CubITS positioning system" for vehicles [19].

Nevertheless, since each conventional WYSIWYAS navigation application running on a mobile phone captures a still picture [16]-[18], they cannot provide real-time and seamless navigation for a user. They urge a user to stop and to take a still picture when he/she wants to know the direction of the destination. It requires a series of operations (it impairs the real-time response of the system) and it is repeated every time the user wants the navigation information (it impairs the seamless of the system). It is insufficient from the point of view of the design concept of WYSIWYAS (What you see is what you are suggested) because if a mobile device has moved, the current scene is different from the captured image before moving. Further, series of operation (Stopping and taking a still image) is a big burden on the user when he/she uses it in a building that has a lot of branch points or entrances of rooms. This may lead to some problems that effect on the usability of it. First, this could impede other pedestrian traffic because users stay on the passage way. Second, when marker recognition was failed, it's necessary to do the same operation once again. Third, for a person using a wheelchair the photography operation is troublesome. This is because it is not available during movement. Fourth, the intuition of the navigation is spoiled because the conventional system cannot support the posture change of the user. To realize more intuitive and high usability WYSIWYAS navigation, it is necessary to guide the user seamlessly in real-time to remove these problems.

In this paper, a prototype system of a real-time and seamless WYSIWYAS (RS-WYSIWYAS) navigation system that does not require a user to know the direction of the destination has been developed. To achieve real-time and seamless WYSIWYAS navigation, our proposed system processes a captured camera-preview movie, and superimposes the navigation information on it. An Android tablets with rear camera are used for the proposed system. A smart tablet captures a scene movie and extracts consecutive some M-CubITS marker elements (classified by color "red" or "green" according to the bit information) in order to recognize the location of the marker elements and the direction, then overlays an arrow symbol on the screen, indicating the direction of the destination. Meanwhile, because the resolution of the camera becomes higher in recent years, it causes higher image processing time that causes severe application delay. Moreover if there are non-marker objects of similar color to M-cubITS marker elements, the application misunderstands them as marker elements. As a result the correct location of marker elements cannot obtained and the device provide wrong navigation.

In this paper, section 2 explains conventional WYSIWYAS navigation system that offers uses intermittent WYASISYAS navigation. In section 3, the prototype of RS-WYSIWYAS navigation system is described. In section 4, usability improvements of the prototype application are conducted. The application's response is improved by resizing the captured movie (sequential pictures) and using native code programming. Moreover marker recognition performance is improved by applying pre-masking processing to a captured movie. This omits the information of unwanted areas where M-CubITS marker elements do not exist, and extracts marker sequence information. In section 5, this paper is concluded.

2 WYSIWYAS NAVIGATION SYSTEM

2.1 Concept of WYSIWYAS Navigation System

WYSIWYAS is a concept of Human Machine Interface (HMI) for virtual navigation systems [13]-[15]. When users employ smart devices with a WYSIWYAS navigation system, they capture images using the WYSIWYAS navigation application. Then the application displays an arrow symbol that points to the final destination on the captured image. Figure 1 shows a usage scene example of WYSIWYAS navigation system. If a user goes to the direction that is indicated by the arrow symbol, they will get to their destination. That is, a smart device does not need to know the accurate and precise own location information. To implement these features, the WYSIWYAS navigation system is required to recognize the relationship between the destination and location of the captured image, and direction of the smart device.

2.2 M-CubITS

M-CubITS is a positioning scheme that places multimodal marker elements (as shown in Fig. 2) according to M-sequences along a corridor, detects a row of M-CubITS elements with a camera, compares the row with a database, and determines the location and direction of the captured marker elements. M-CubITS marker elements are designated as either "0" or "1" and distinguished by the color of the figure. An example of a marker sequence arrangement is shown in Fig. 3. Marker elements are arranged to one line according to an M-sequence code. The top side of an isosceles triangle points to the forward direction. Because an M-sequence code is generated from the Linear Feedback Shift Register (LFSR) of m-stages with maximum code length 2^m-1, the device can recognize its unique position by observing m-chips (Fig. 3). For instance, if a passageway 1 km long is covered by markers that contain 1 bit of information placed 1 meter apart, the required number of shift register stages is 10 (maximum code length:1023 chips). Therefore, marker positioning is required to capture at least 10 markers in an image.

In the case of Fig. 3, when a smart device captures markers from the second to the fourth, it gets a bit-sequence "110". After that, the smart device finds the same bit-sequence from the row of an output bit of LFSR that is recorded to a map database. Then the smart device understands that location ID 2, 3 and 4 markers come out on the screen. Furthermore the smart device judges the direction where the user advances from the location ID of the present location and the location ID (it has been enrolled in map data) of the destination.

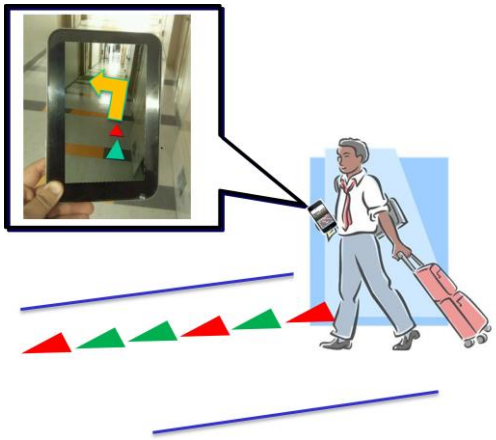
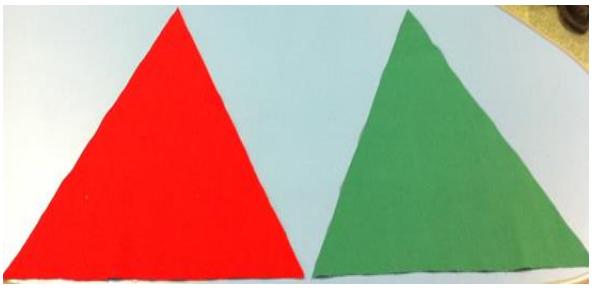


Figure 1: Usage scene example of WYSIWYAS navigation



Red: “1” Green: “0”

Figure 2: Marker elements for M-CubITS

After the marker elements installed into the facility, the location of each marker elements is related to the map data. When renewing guide information, only the map data is updated.

3 REAL-TIME AND SEAMLESS WYSIWYAS NAVIGATION SYSTEM [20]

3.1 Overview of Real Time and Seamless WYSIWYAS Navigation

To indicate the direction to the destination in real-time and seamlessly, the proposed system basically conducts conventional image processing to each frame of a captured movie. That is, multiple images captured successively should be processed in a short period. The device recognizes the sequence of the captured marker elements and its direction, and displays an arrow symbol on the original scenic image. The developing environment of this navigation application is the Eclipse [21] with Android SDK [22]. Besides, some image processing modules use the OpenCV library for the Android [23]. Although the process of marker recognition is based primarily on the scheme in [16], some of processes use techniques different from Ref. [16] in consideration of the processing time.

In this study, we adopted the M-CubITS marker elements shown in Fig. 2. These elements are either red or green; this binary-information is used as chip information for the M-sequences. The shape of the elements is triangular to facilitate recognizing the direction of the M-sequence.

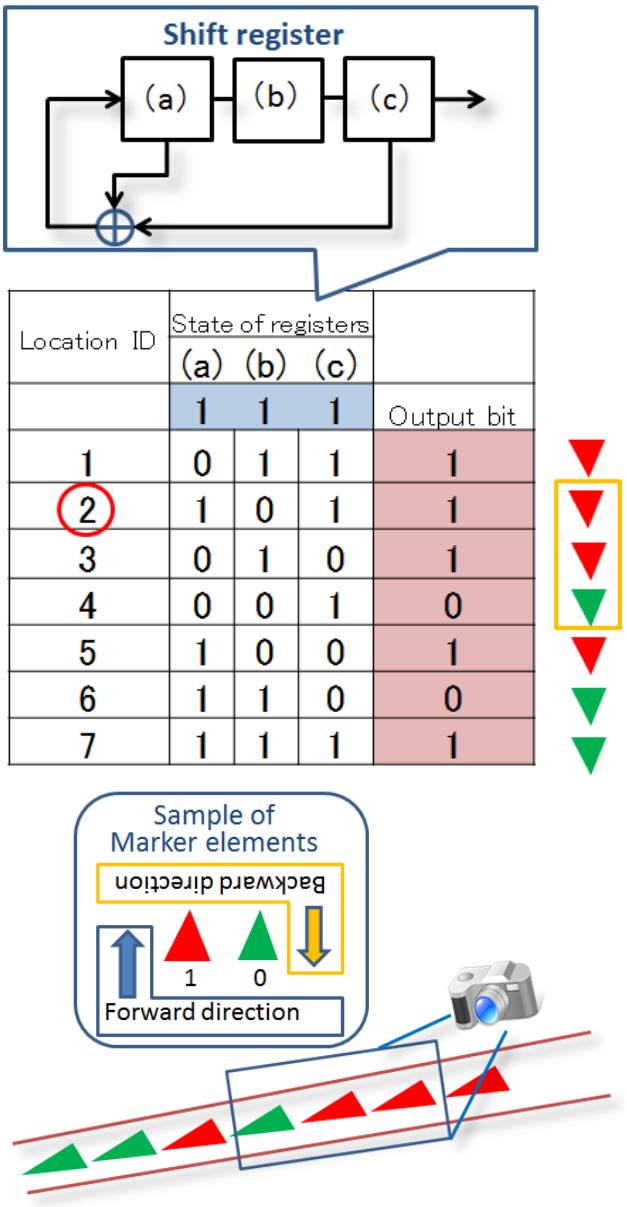


Figure 3: Example of marker sequence arrangement

3.2 Flow of Marker Recognition Process

The process for marker element recognition is as follows:
(1) Real-time image capturing by a rear camera
(2) Particular color extraction

The marker-colored areas that are colored green or red are extracted from the captured image and converted into an HSV (color model) image. In this process, the captured image becomes binary; red or green areas are designated as “1”; otherwise, they are designated as “0.” The color-extracted image is shown in Fig. 4 (This image is an 8bit gray scale image, and the brightness value of the marker-colored area is given 255 and the brightness value of other area is given 0). To reduce the noise of the binary image, a smoothing and morphology operation is conducted. An example of a noise-reduced image is shown in Fig. 5.

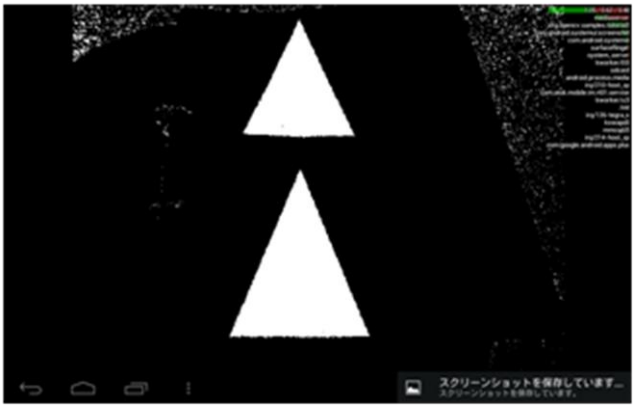


Figure 4: Particular color extracted image

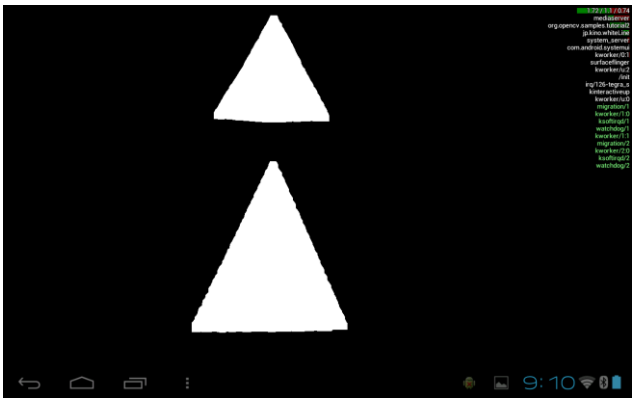


Figure 5: Noise reduction image

(3) Contour extraction

Contours of marker element are extracted and labeled with numbers. The label numbers are assigned in order from the bottom to the top of the image. In the marker element recognition process (described later), marker element information is acquired in a specific order, starting with the marker with the smallest label number. The contour image is shown in Fig. 6.

(4) Marker element recognition

To extract information from a marker element, the centrobaric coordinate of the contour is calculated. After that, a horizontal line that passes through the centrobaric coordinate is drawn, and two intersection coordinates with the contour line are calculated. A horizontal line with intersection coordinates (coordinates A and B) is drawn in Fig. 7. Then, the most common pixel color between coordinates A and B is judged as the marker color.

The direction of the extracted sequence of the marker elements is determined as follows:

- The application calculates the gradient of a line that passes through centrobaric coordinates on the first and second marker elements.
- It calculates two coordinates that intersect the horizontal line that passes through the centrobaric coordinate and contour line of the first marker element.
- By drawing two vertical lines from these intersection coordinates to the line that passes through the centrobaric coordinates, two line segments result.

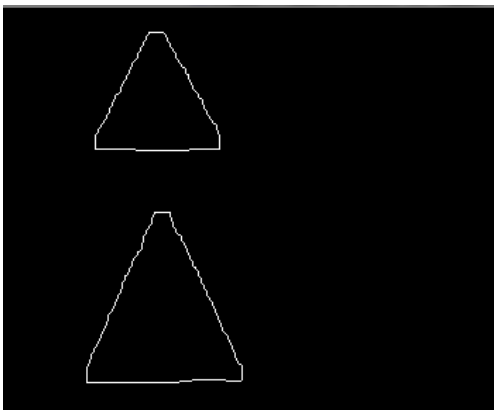


Figure 6: Contour extraction

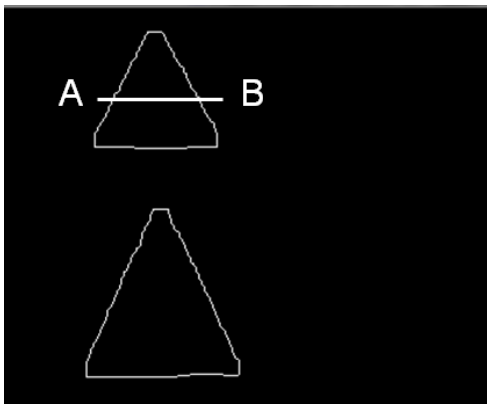


Figure 7: Recognition of marker element color

- These line segments are compared to determine the direction of marker elements. These are shown in Fig. 8. If the gradient of the line is a near right angle, the direction of the marker elements is determined by comparing the number of pixels within the contour line at y = Y_g and y = Y_g-5 (pixel).

(5) Navigation arrow symbol display

Using the marker color and the direction information, the device determines the sequence of marker elements. Then the system judges the direction to go, and overlay a guiding arrow symbol on the original scenic image. If the application cannot extract enough number of marker elements, the sequence of marker elements cannot be specified and arrow symbol is not updated in the processed frame.

3.3 M-CubITS Marker Database

Application of RS-WYSWYAS navigation needs the location of the marker elements that comes out captured movie. To obtain this information, the application references from a marker database. With regard to this system, it is assumed to be used in public or private areas. In public areas, because marker elements are used for public infrastructure, it is thought that a marker database can be used freely through the Internet via mobile networks. On the other hand, in private areas, a building manager might not want to disclose marker data about the building to unauthorized individuals. Moreover, mobile phone might be restricted within such private areas. In these cases, it is assumed that smart-device would access the M-CubITS database via the

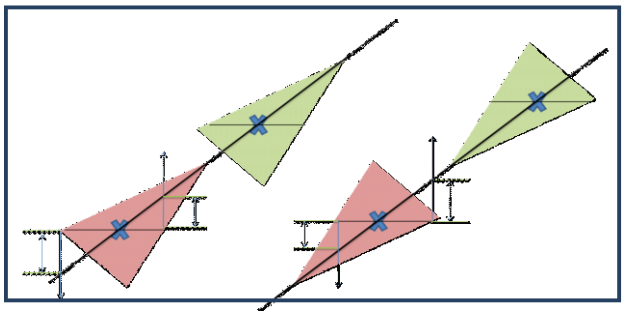


Figure 8: Determining marker direction

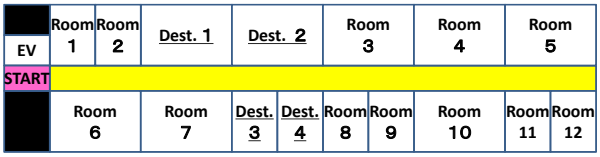


Figure 9: Diagram of experimental area

Table 1: Device specifications
(Toshiba Regza Tablet AT570, 2012)

CPU		NVIDIA® Tegra® 3 Mobile Processor
	Frequency	1.30GHz
	Number of cores	NVIDIA®4-PLUS-1™ Quad core
	Cache memory	1MB
Memory	Capacities	1GB（On board）
Camera		Front camera (1.2MP)
		Rear camera (8MP)

wireless local-private network prepared in the building. In this paper, we constructed an environment of WYSIWYAS navigation system for a corridor in a building. The prototype application performs navigation from the state where the M-CubITS data have already been downloaded in the device, assuming that use of the M-CubITS database has been allowed by the manager of the building.

3.4 Indoor Experiment of Prototype RS-WYSIWYAS Navigation

3.4.1 Overview of Experiment

To evaluate the usability of the proposed system, an M-CubITS WYSIWYAS navigation system was prepared for a building at Kanagawa Institute of Technology. In this experiment, an M-sequence generated from a 7-bit LFSR was adopted, and the marker elements were arranged 0.5 meter apart. A passageway of approximately 60 m was covered, and a smart device recognizes the captured marker positions on the passageway by capturing an image of at least seven marker elements. We made a marker arrangement where the triangle marker element (40 cm × 40 cm and made of cloth) lined up for a corridor (40m length), and have made a database which corresponds to this marker sequence. In this experiment, we used an Android tablet, the Toshiba REGZA Tablet AT570. The specifications of this device are listed in Table 1. The proposed application was installed on the tablet. A subject first selects a destination from among four predetermined locations (classrooms). Then, he/she faces the rear

Figure 10: Navigation application user interface

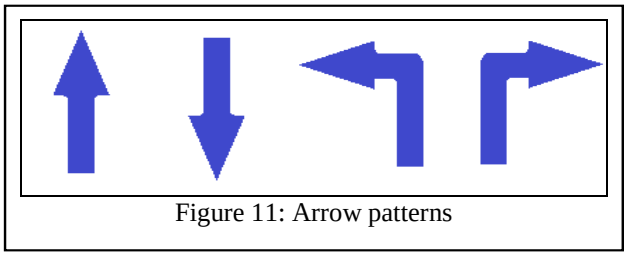


Figure 11: Arrow patterns

camera to the corridor and capture a movie with at least seven marker elements. The application then determines the location of the marker elements, searches for the destination, and displays a guidance arrow symbol on the captured movie to indicate which direction the user should go. In this experiment, because all subjects belongs to Kanagawa Institute of Technology, they might already know the location of each destination room. Therefore, we gave each destination a name that was different from any actual room name. Figure 9 shows a diagram of the experiment area. Figure 10 shows what the user will see on his/her device. There are four buttons on the left-hand side of the screen; touching one selects a destination. These destinations (Dest.1, 2, 3, 4 in Fig. 9) are at almost the same position from the starting point. During navigation, the device indicates the direction of the destination by a guidance arrow symbol; arrow patterns are shown in Fig. 11. For this experiment, a right turn arrow or a left turn arrow points entrance of rooms; it indicates that the subject has arrived at the room of destination. Subjects (12 peoples) conducted the following two experiments in order.

a) Intermittent Navigation (conventional system)

A subject stands at starting point and selects a random destination from the device. Then, the subject turns over the smart device in order to avoid looking at the screen. Next, a staff member shows the subject a diagram (Fig. 9) for a brief time period, permitting the subject to memorize the location of his or her selected destination. The subject then starts looking for the destination without watching the device. If the subject becomes lost, he/she stops, and faces the smart device to nearby marker elements and observes the guidance arrow. Then, the subject turns over the smart device again and continues looking for the destination.

b) RS Navigation (proposed)

The subject stands at starting point again and selects a different destination . He/she faces the camera of the smart device to marker elements and observes the guidance arrow. The subject continues looking for the destination while displaying guidance on the device screen.

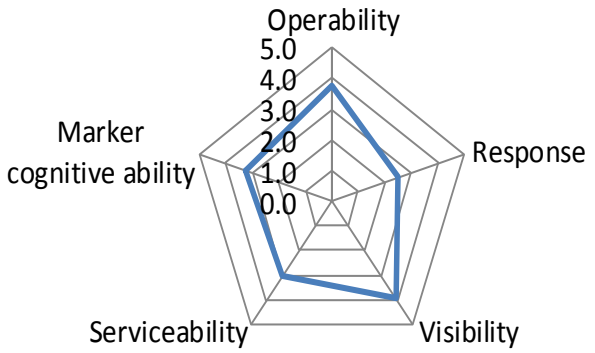


Figure 12: Result of Q1

Table 2: Result of Q2-Q4 and travel time

subjects	Q.2	Q.3	Q.4	Travel Time[s] (intermittent)	Travel Time[s] (RS)
A	Y	Y	RS	362	62
B	Y	N	RS	65	51
C	Y	N	RS	263	85
D	Y	N	intermittent	41	85
E	Y	N	RS	52	40
F	Y	N	RS	170	50
G	N	Y	RS	193	118
H	Y	Y	intermittent	89	115
I	Y	Y	RS	48	65
J	Y	Y	RS	52	39
K	Y	N	RS	278	61
L	Y	Y	intermittent	300	248

During the subject conducts an experiment, travel time from start to destination was measured. After each experiment, subjects answered questionnaires. The questions were as follows.
[Q.1] Please rate the following on a scale of one to five (Very Good: 5, Good: 4, Fair: 3, Poor: 2, Very Poor: 1). {Operability/ Response/ Visibility/ Serviceability/ Marker cognitive ability}
[Q.2] Is the navigation easy to understand? (yes/ no)
[Q.3] Is the load of the proposed system heavy? (yes/no)
[Q.4] Which application is easier to use? (intermittent / RS)
[Q.5] Did you have any trouble with the applications? (Free writing)

3.4.2 Results

The questionnaire results of Q1 is shown in Fig. 12 and answers of Q2, Q3, Q4 and travel times are listed in table 2. Subjects who answered the free writing questions of Q5 provided the following answers:

- "It is difficult to capture seven marker elements."
- "This application must improve its ability to recognize marker elements without error, the frame rate of the preview screen and the response speed of the arrow display."
- "Implementation of an audio based command feature is expected."

In these answers to the questionnaires, the application is given high marks in operability and visibility. On the other hand, response, marker cognitive ability are rated lower. It is assumed that lack of response and marker cognitive ability lead to low serviceability. Therefore, this application needs to improve the response and marker cognitive ability in order to increase the quality of the navigation system.

Moreover, since most of subjects answered that the navigation is easy to understand, this application achieves an intuitive navigation interface for users. The first experiment was performed using the conventional navigation with which the user is shown the direction to the destination when he takes a picture (intermittent navigation). Meanwhile, the second experiment was performed using the proposed application that offers real-time and seamless navigation. Our questionnaire indicated that most subjects navigated smoother in the RS-navigation experiment than that of intermittent navigation. Therefore, the proposed prototype application that captures a movie and processes them in every frame, providing the user with a real-time intuitive navigation system is more beneficial than the conventional scheme. In both experiments, all subjects arrived at their destination. However, there were some cases when the device could not detect marker elements. When the device cannot detect correct marker elements information, wrong directions are shown to the subject. Although most subjects arrived at the destination in about 40 seconds, some subjects took more than 5 minutes to arrive if they had trouble with marker recognition errors.

4 IMPROVEMENT OF USABILITY

4.1 Response Improvement

4.1.1 Resizing Captured Images

There is a concern that the high processing load would cause navigation because the resolution of current smart devices has become larger, delays. To reduce the influence of the image processing load that depends on image size, the size of the captured image is reduced before the main process.

4.1.2 Implementation of Native Methods

In order to reduce processing time, several methods have been replaced with native methods. In the prototype navigation application that works on the Android OS, most of methods are coded as non-native methods except for some image processing methods using the OpenCV library that provides the native method for image processing. Though non-native methods are used for maintenance and portability in the prototype application, they have to be replaced into native method to improve its response. The marker recognition process performed after each image capture is divided into eight parts, as shown in Fig. 13. Then, the response performance of the prototype RS-WYSIWYAS navigation system was analyzed. The prototype RS-WYSIWYAS navigation application was installed into a smart tablet (Nexus 7 2013 model, because the battery life of the terminal which we used ran out and it has stopped production, substituted smart devices are used), and the average processing time for each process was recorded. The top of the data shown in Fig. 14 is the result of the prototype application analysis. This result shows that a large screen size (1920*1200) costs

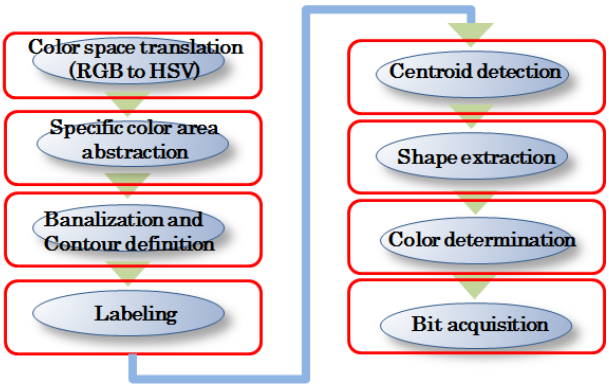


Figure 13: Flow of image processing

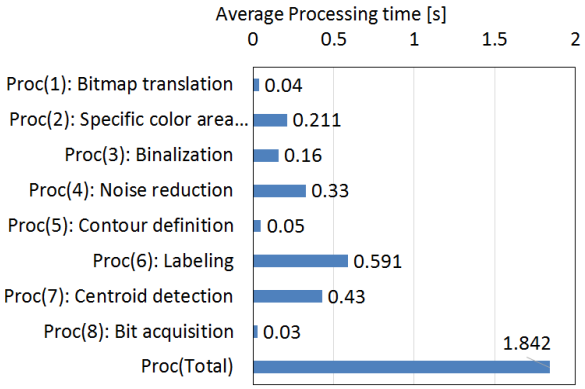


Figure 14: Analysis of processing time of prototype application

approximately 1.8 seconds of the entire processing time. This processing time could result unsatisfactory to users. Moreover, the processing time for labeling process occupies the largest percentage of the total processing time. This process also includes less native library code offered by OpenCV. Therefore, this process is the first candidate of the replacement to native code.

4.1.3 Evaluation Experiment

First, the influence of image resizing is evaluated. The second and third data in Fig. 15 are the results of the marker recognition process with resizing of the captured movie. In these results, the resizing rate of 0.35 shows approximately 0.31 seconds of processing time, which is approximately 1/6 of the prototype application, and this is the shortest processing time among the top three results. This processing time represents 3 Hz of the navigation updating cycle. Recently, most generic GPS receivers for navigation adopt 1 Hz of the positioning updating frequency. Compared with this, the improved application can navigate with higher updating frequency. Meanwhile, although we attempted to test a lower than 0.35 resizing rate, the image processing program cannot output a navigation arrow on the screen constantly because it cannot detect sufficient marker elements from the resized image. This result indicates the trade-off between the marker size and response. That is, though the prototype application has possibility to adopt 0.35 times of size (height: 0.15m, base length: 0.15m) of marker elements, improved application achieves more fast response in exchange for the down-sizing of the marker.

Table 3: Device specifications (Google Nexus 7 2013)

CPU		Snapdragon S4 Pro (APQ8064)
	Frequency	1.5GHz
	Number of cores	quad core
memory		2GB
Camera	Front	120MP
	Rear	500MP
Screen	Size	7inch
	Resolusion	1920 × 1200
OS		Android 4.3

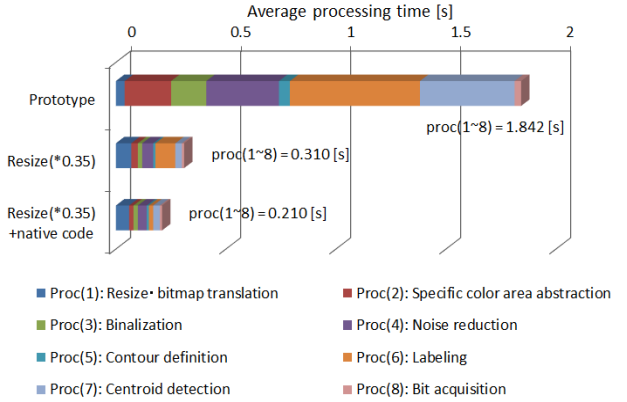


Figure 15: Result of performance evaluation

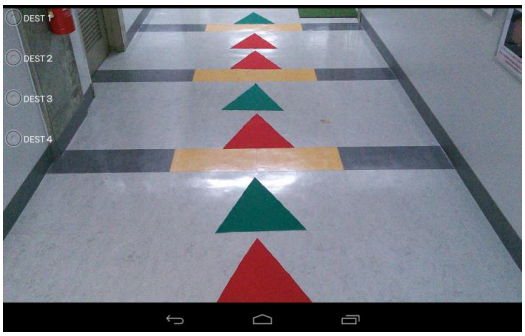
Furthermore, the influence of the replacement of byte codes with native code is evaluated. The results of the processing time of using native codes is shown at the bottom of Fig. 15. This indicates that the labeling processing time is reduced from 0.35s to 0.21s. It also indicates that the total processing time has been reduced to approximately 1/9 of the prototype application; that is, a navigation update of approximately 5 Hz frequency is achieved. Therefore, this prototype system has possibility to show the navigation information to the user by the update interval similar to the case to use GPS.

4.2 Improvement of Marker Element Recognition

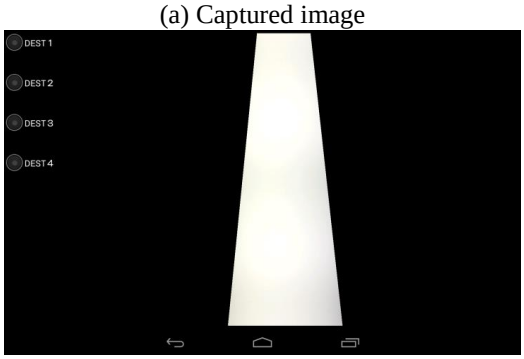
4.2.1 ROI Mask Generation Using Past-extracted Marker Information

Typically, some of objects on the corridor have similar colors of marker elements. For the prototype application, it is difficult to distinguish a non-marker object that has similar colors of marker elements from marker elements placed on the floor. If non-marker objects are extracted and treated as marker elements, the application shows the wrong arrow symbol to the user.

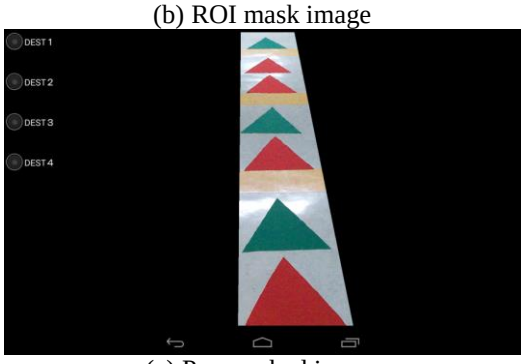
To reduce wrong guidance, pre-masking processing for the captured movie has been implemented. This process allows omitting the information of unwanted areas where M-CubITS marker elements are not placed, and extract only the marker sequence information. To exclude unwanted areas, the marker information extracted in the previous frame is used.



(a) Captured image



(b) ROI mask image



(c) Pre-masked image

Figure 16: Generating pre-masked image

First, we derive a regression line using the centrobaric coordinates of the marker captured previous frame. Next, the Region of Interest (ROI) mask image is generated with a trapezoidal or triangular region along the regression line. In this process, if the regression line crosses the top of the screen, a trapezoidal region is placed along the regression line; meanwhile, if the regression line crosses the side of the screen, a triangular region is allocated along the regression line. Subsequently, a pre-masked image that is used for marker detection is generated by the captured image filtered by the ROI image. Figure 16 shows a generated pre-masked image.

4.2.2 Evaluation Experiment

4.2.2.1 Evaluation Method

To evaluate the improved marker recognition, marker elements are placed on the corridor illustrated in Fig. 9. We used an Android tablet, the Google Nexus 7 (2013 model, in table3). In this corridor, there are some non-marker elements that has similar color of marker elements. Then four types of experiment is conducted.

i) Prototype app. / corridor without non-marker object
ii) Prototype app. / corridor with non-marker object
iii) Improved app. / corridor without non-marker object
iv) Improved app. / corridor with non-marker object

When experiment without non-marker object is conducted, non-marker object that has similar color of marker elements (green and red) are removed or covered with white cloth.

12 subjects conducted 4 types of experiment by following procedure. To take counter balance, half of the subjects (6 persons) conducted prototype application experiments (experiment (i) and (ii)) in the beginning, the rest of subjects (6 persons) conducted the improved application experiments (experiment (iii) and (iv)).

(1) The subject runs the navigation application (prototype or improved) on the smart device.
(2) The subject stands at the starting point.
(3) The subject is told the destination and sets the destination in the application.
(4) The subject searches for the destination according to the navigation instructions.

After subjects finish four experiments, they answer questionnaires. The questions are as follows:

Q.1 Wrong indication time ratio (WITR)
The wrong indication time ratio is defined as a time ratio that perceived as having been presented with the wrong indication. The subject estimates this value from 0% to 100% in step of 10%.

Q.2 Availability
A subject rates the usability of the application through navigation with three choices (fully available/partially available/not available).

Q.3 Acceptable WITR
This is defined as the value of WITR that the user may want to use the application. The subject answers this value from 0% to 100% in step of 10%.

4.2.2.2 Results

Figure 17 shows the WITR in the case where there are few non-marker objects around M-CubITS marker elements on the corridor. In this graph, the WITR using the prototype application distributes broadly. On the other hand, the WITR using the improved navigation application concentrates around 10%. This result indicates that most subjects feel that the improved application provides less wrong indications than the prototype application. Meanwhile, Fig. 18 shows the WITR in the case where there are some non-marker elements on the corridor. In this graph, the distribution of the WITR using the prototype application concentrates around 60% to 100%. It is considered that this degradation of the performance is caused by the marker sequence recognition error. On the other hand, the distribution of the WITR using the improved application concentrates around 0% to 30%. This result is approximately equal to the case where there are some non-marker elements on the corridor. It is indicated that improved application can suppress the influence of surrounding non-marker objects.

Next, the results from questionnaire Q.2 are shown in Table 4. In this question, subjects choose availability of this system. For the prototype system, there are few users judging it to be available enough. Especially, in the case where

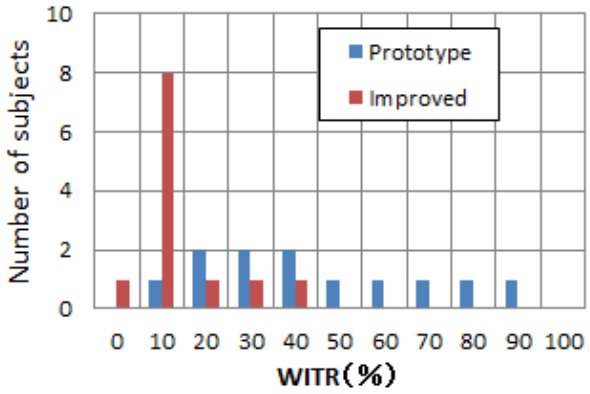


Figure 17: Wrong indication time ratio
(There are few non-marker objects)

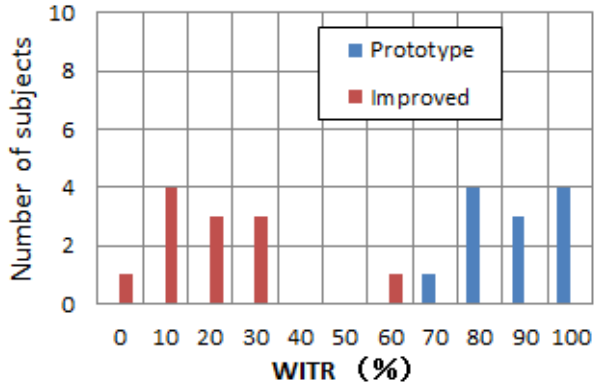


Figure 18: Wrong indication time ratio
(There are some non-marker objects)

Table 4: Availability

	Fully available	Partially available	Little available
Prototype	2	4	6
Prototype (some non-marker object)	0	2	10
Improved	5	4	3
Improved (some non-marker object)	2	7	3

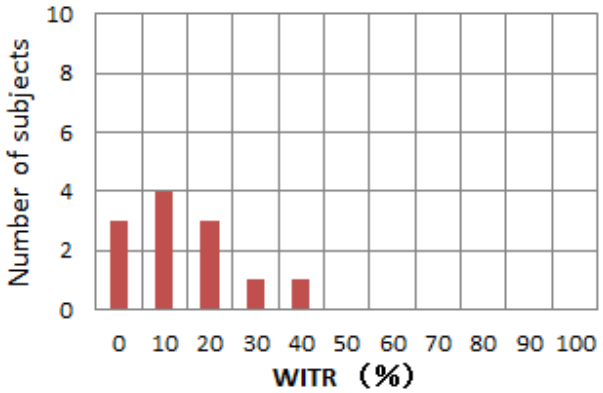


Figure 19: Acceptable WITR

there are some non-marker elements on the corridor, most users feel that the system is insufficient. On the other hand, for the improved system, the availability of subjects is increased. Whether there are non-marker objects or not, 75% of the subjects accept this improved application. However, three subjects feel that the improved application is not acceptable for navigation. This system is effective if it can extract correct marker information in previous frame. However if it cannot extract, it perform same processing as the prototype application. Because the user can recognize a marker on the screen, a function to input a centrobaric coordinates of extracted marker element into the application is expected.

Finally, the result from questionnaire Q.3 is shown in Fig. 19. In this graph, the acceptable WITR is distributed around 10%, and this result shows the same tendency as the time ratio perceived by subjects. On the other hand, three subjects answered that 0% of WITR is required, which corresponds to the results from questionnaire Q.2. Although the improved application reduces marker sequence recognition error, some subjects are not satisfied. This result will be an indicator for usability improvements.

5 CONCLUSION

In this paper, a real-time and seamless WYSIWYAS pedestrian navigation system for smart devices was proposed. In order to provide a user with real-time seamless navigation, the proposed navigation application recognizes consecutive some M-CubITS marker elements on every frame of captured movie, estimates its location and direction, and shows a user the direction that he wants to go. We compared the performance of a prototype of RS-WYSIWYAS navigation application with conventional WYSIWYAS navigation application through navigation experiments in an indoor corridor in which M-CubITS marker elements were arranged. The results of these experiments indicated that the proposed application was evaluated more intuitive and intelligible than the conventional intermittent navigation. Nevertheless, the recognition performance of marker elements and response time of the application were not highly evaluated.

Therefore, in order for the prototype application to become practical, the response and the marker elements recognition performance are improved. From the perspective of the response time, resizing captured images reduced the computation load. In addition, the implementation of native methods also reduced processing time. The results of processing time measurements indicated that a navigation updating cycle of approximately 5 Hz was achieved.

Meanwhile, from the perspective of marker elements recognition, pre-mask processing of captured images using the marker information obtained from a previous frame improved the marker elements recognition performance. The results of questionnaires for the navigation experiments indicated that the improved navigation application reduces the WITR compared with the prototype application. In addition, most subjects considered that the availability of the improved application is higher than the prototype application.

In this study we conducted experiments on a single floor indoor environment. However, generally a building with complicated structure needs the guidance of the user. Therefore, implementation methodology to complex buildings (multiple layer, crossing, etc.) is required. Moreover, to improve the marker recognition performance at the beginning of the navigation will be needed.

6 REFERENCES

[1] J. Soubielle, I. Fijalkow, P. Duvaut, and A. Bibaut, "GPS positioning in a multipath environment," IEEE Trans. Signal Process. vol. 50, no. 1, pp. 141-150 (2002).
[2] Y. Chen and H. Kobayashi, "Signal Strength Based Indoor Geolocation," in Proc. IEEE ICC, pp. 436-439 (2002).
[3] J. Moriya and T. Hasegawa "A study of Inverse GPS based positioning system," Proc. of IEEE ITSC 2004, pp.559-563 (2004).
[4] T. Hada, H. Sunaga, M. Akiyama, S. Ioroi, and H. Tanaka, "Investigation and Demonstration of Local Positioning System using Ultrasonic Sensors for Wide Indoor Areas," International Conference on Ambient Intelligence 2010, pp.280-284 (2010).
[5] A. Ward, A. Jones and A. Hopper, "A New Location Technique for the Active Office," IEEE Personal Communications, Vol. 4, No. 5, pp. 42-47 (1997).
[6] K. Kaemarungsi and P. Krishnamurthy, "Modeling of indoor positioning systems based on location fingerprinting," INFOCOM 2004. Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies, vol.2 Volume: 2, pp. 1012-1022 (2004).
[7] J. Rekimoto and K. Nagao, "The world through computer," Proceeding of the ACM Symposium on User Interface Software and Technology (UIST95), pp.29-36 (1995).
[8] I. Siio, T. Masui, and K. Fukuchi "Real-world Interaction using the FieldMouse," CHI Letters, Vol.1, Issue1, pp.113-119 (1999).
[9] I. Siio, "User Position Detection using RFID Tags," IPSJ-Technical Report 88, pp.45-50 (2000).
[10] H. Matsubara, N. Fukasawa, K. Goto, K. Kawai, T. Sato, T. Aoki, N. Mizukami, K. Fujinami, and A. Shinomiya, "Interactive Guidance System for Passengers," QR of RTRI 42(4), pp.201-206 (2001).
[11] T. Hasegawa, "A study on ITS and systems innovation," Technical Report of IEICE, no. ITS 2002-120, pp. 13-17 (2003) (in Japanese).
[12] T. Hasegawa, "ITS platform EUPITS–approach to realization," Technical Report of IEICE, no. ITS 2003-8, pp. 41-47 (2003) (in Japanese).
[13] K. Kurihara and F. Sato, "Marker Based Indoor Navigation System Sharing Location Information," DICOMO2014 Symposium, pp.1099-1103 (2014) (in Japanese).
[14] H. Kato and M. Billinghurst, "Marker Tracking and HMD Calibration for a video-based Augmented Reality Conferencing System," Proc. of the 2nd International Workshop on Augmented Reality, pp.85-94 (1999).
[15] T. Hasegawa, "ITS platform EUPITS–toward realization," Technical Report of IEICE, no. ITS 2003-26, pp. 29-35 (2003) (in Japanese).
[16] S. Yamashita, and T. Hasegawa, "On the M-CubITS pedestrian navigation system by a camera-equipped mobile phone," Proc. of ITSC2004, pp.714-717 (2004).
[17] T. Manabe, S. Yamashita, and T. Hasegawa, "On the M-CubITS pedestrian navigation system," Proc. 9th IEEE Int. Conf. on ITS, pp. 793-798 (2006).
[18] T. Manabe, T. Hasegawa, Y. Matsuoka, S. Furukawa, and A. Fukuda, "On the M-CubITS pedestrian WYSIWYAS navigation using tile carpets," Proc. 10th IEEE Int. Conf. on ITS, pp. 879-884 (2007).
[19] R Masuda, J Kim, T Hasegawa, "On Direction Position Oriented M-CubITS Vehicle WYSIWYAS Navigation," IEICE Trans. Vol. J91-A,No.1,pp.11-20 (2008) (in Japanese).
[20] Y. Takatori and H. Takeo, "A real-time seamless WYSIWYAS navigation system for smart devices," Proc. of ITST2013, pp. 163-168 (2013).
[21] Eclipse, http://www.eclipse.org/
[22] Android SDK, http://developer.android.com/index.html
[23] OpenCV library for Android, http://opencv.org/platforms/android.html

(Received September 30, 2015)
(Revised December 26, 2015)

Yusuke Takatori received his B.E. and M.E. degrees in Electrical and Electronic Systems Engineering from Saitama University, Japan in 2002 and 2004 respectively. He also received a Ph.D. degree in Information and Mathematical Science from Saitama University, Japan in 2007. From 2007 to 2012, he was an Assistant Professor in the Department of Management Science, Faculty of Engineering, and Tokyo University of Science. From 2012 to 2015, he was an Assistant Professor in the Department of Electrical and Electronic Engineering, Kanagawa Institute of Technology. Since 2015, he has been an Associate Professor. His research interests are in the area of Intelligent Transport Systems (ITS). Dr. Takatori is a member of IEEE and IEICE.

Shohei Iwasaki received his B.E. degrees in Electrical and Electronic Systems Engineering from Kanagawa Institute of Technology, Japan in 2015. His research interests include image processing for Intelligent Transport Systems. He is currently working at Daito electron co., ltd.

Tatsuya Henmi received his B.E. degrees in Electrical and Electronic Systems Engineering from Kanagawa Institute of Technology, Japan in 2015. His research interests include image processing for Intelligent Transport Systems. He is currently working at Rokko & Associates, Inc.

Hideya Takeo received B.S. and M.S. degrees from Kanagawa University, Japan in 1984 and 1986 respectively. He also received Ph.D. degree on bio-applications and systems engineering from Tokyo University of Agriculture and Technology, Japan in 2005. From 1986, He worked Fuji Photo Film Co., Ltd. and engaged in research on Medical Imaging Technology. From 2006 to 2009, he was an Associate Professor, and From 2009, he has been a Professor in the Department of Electrical and Electronic Engineering, Kanagawa Institute of Technology. His research interests are in the area of Image processing. He is a member of ITE (The Institute of Image Information and Television Engineers), Jamit(The Japanese society of Medical. Imaging Technology), IIEEJ (The Institute of Image Electronics Engineers of Japan), and MII (Medical Imaging and Information Sciences).

Evaluation About the Feasibility of an Unconscious Participatory Sensing System with iOS Devices

Takamasa Mizukami†, Katsuhiro Naito‡, Chiaki Doi*, Ken Ohta*,
Hiroshi Inamura*, Takaaki Hishida‡, and Tadanori Mizuno‡

†Graduate School of Business Administration and Computer Science, Aichi Institute of Technology, Japan
‡Faculty of Information Science, Aichi Institute of Technology, Japan
* Research Laboratories, NTT DOCOMO, Inc., Japan
naito@pluslab.org hishida@aitech.ac.jp mizuno@mizulab.net

Abstract - This paper develops a prototype implementation of our proposed unconscious participatory sensing system and evaluates its performance about a measurement process on a real smartphone device. The proposed system consists of various beacon devices and smartphone devices. It also requests smartphone owners to install a special application on their smartphones to collect measurement information from beacon devices. Each beacon device has Bluetooth low energy (BLE) module to communicate with a smartphone device. Hence, the proposed system can collect measurement information from beacon devices through smartphones. Additionally, beacon devices can work by a battery for a long time because BLE is appropriate for a low power operation. The feature of the proposed scheme is to activate a special smartphone application in a sleep state to upload measurement information. Therefore, it does not require smartphone owners to launch the smartphone application to collect measurement information. We employ the iBeacon function which is the special beacon mechanism for iOS to active the application. In the experimental evaluation, we have developed a beacon device with a Raspberry Pi and a special application for iOS. Since a background operation period of iOS applications is limited due to the power saving mechanism on iOS, we have evaluated the measurement process of the proposed system. From the results, we can find that the developed application for iOS can collect information from beacon devices in a background operation even if the background operation period is limited.

Keywords: Unconscious participatory sensing, iBeacon, BLE, Smartphone device, Beacon device.

1 INTRODUCTION

Sensor networks have received considerable attention to collect various information [1]. Participation sensing systems, where general participants collect information, have been proposed to realize more flexible sensing systems [2]–[4]. Conventional participatory sensing systems usually request general participants to join their sensing network system as a volunteer to collect information flexibly. In sensing process, the system requests some participants to measure information and to report the measured information to the system [5]. Therefore, participating in the form of volunteering is a key point to realize a practical sensing system. The types of measurement information collected in participatory sensing systems are classified as abstract information, which is evaluated by the participants [6]–[8], and quantitative information, which is measured by sensors [9]–[11]. Since measurements of abstract information are difficult to acquire by sensors, participants should join the sensing process voluntarily to realize effective participatory sensing systems. Hence, various studies have investigated techniques that incentivizes participants to join the sensing process [12], [13]. Additionally, some researchers have attempted to realize a hybrid sensing of participatory sensing and social media [14].

Participatory sensing systems for quantitative information also require participants' interaction behavior, such as checking measurement requests, moving to a measurement location, launching the application, and reporting measurement information. Therefore, participants in conventional systems should operate a measurement application to obtain target information even if the target information may be automatically measured by sensors. For the above reasons, conventional participatory sensing systems currently attract early adopters who are interested in the new service.

Some studies employ smartphones as a communication device and special measurement devices to achieve accurate measurement by the same type of sensors and under the same implementation conditions [15]. The accuracy of measurement information given by acceleration sensors, magnetism sensors, etc. is generally stable, even if the actual specification of sensors is different, because the accuracy depends on a specification of sensors and does not depend on the implementation conditions. Hence, acceleration sensors and magnetism sensors are usually used to realize an indoor pedestrian navigation [16]. However, the accuracy of measurement information given by temperature sensors, illumination sensors, etc. is expected to vary according to the precision of the sensor and implementation environment because different smartphones have different types of sensors, and the implementation condition of the sensors is also different. Therefore, different smartphones obtain a different measurement value even when they try to measure a same environment. As a result, the built-in sensors in smartphones are not sufficient to acquire accurate measurement information in real situations.

We have considered how to carry out a sensing process without participants' interaction [17]. The proposed system consists of beacon devices and a special smartphone application. We assume that beacon devices are installed at a specific

place because our target service is an environmental measurement at a fixed place. Since a background processing period of an application on iOS is limited within 10 seconds, the special smartphone application for the measurement is usually suspended by iOS. Therefore, we employ iBeacon [18] function to trigger the special smartphone application in the suspended status to activate a background processing. As a result, the proposed system can realize that the special smartphone application can collect measurement information from beacon devices automatically.

In the experimental evaluation, we have developed a prototype beacon device with a Raspberry Pi which is an ARM-based microcomputer board and a special application for iOS. Since the developed application is usually suspended by iOS, the power consumption of the application is limited. Additionally, iOS permits a background processing of applications within 10 seconds. Therefore, we have evaluated that the proposed information collection process can complete within 10 seconds. The evaluation results show that the information collection process can complete within 5 seconds with a real iOS-based hardware.

2 THE PROPOSED UNCONSCIOUS PARTICIPATORY SENSING SYSTEM

2.1 Overview of Unconscious Participatory Sensing System

Figure 1 shows an overview of the unconscious participatory sensing system. The system consists of the beacon device with sensors, the scanning application on a smartphone, and management servers. We employ Bluetooth Low Energy (BLE) as a communication device because power consumption of BLE is quite low and almost all smartphones implement a BLE module[19]. Since the proposed unconscious participatory sensing system requires an automatic operation mechanism of the scanning application, we use iBeacon on BLE communication to activate the scanning application by the beacon device. The benefits of the proposed system are followings.

- Accurate measurement
  The proposed system uses sensors on a beacon device. Therefore, the measurement is performed by the same sensors even if different smartphones collect sensed data. As a result, the system can collect accurate information that does not depend on specifications of smartphones.

- Continuous measurement
  The proposed system can install a beacon device at a specific location. Therefore, the system can easily measure an environment at the specific location continuously.

- Unconscious participants' interaction
  Beacon devices in the proposed system can activate the scanning application through the iBeacon function. Therefore, the system does not require participants' interaction to collect information.

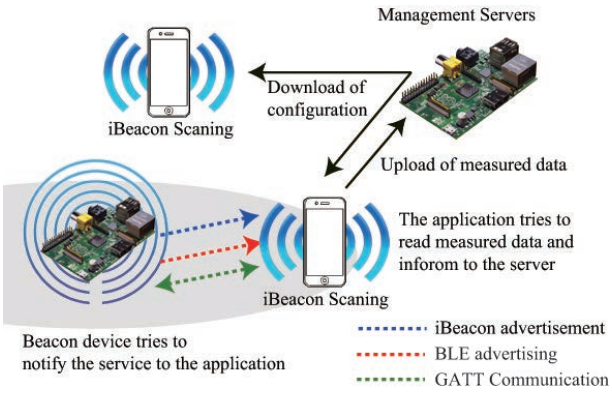


Figure 1: Unconsciousness participation sensing system.

- Usage of participants' cellular network
  Each beacon device uses participants' cellular network to upload sensed information to management servers. Therefore, beacon devices implement only a short-range communication module (BLE).

- Privacy oriented
  The proposed system does not have any privacy information such as participants' positions, status, etc. because the system uses a beacon device to activate the scanning application.

2.2 Structure of Unconscious Participatory Sensing System

The proposed system consists of the management server for the management of beacon devices and measurement information, the scanning application in the smartphone OS to search for beacon devices and to acquire measurement information with built-in sensors, and detectable beacon devices for beacon announcements and measurements with sensors. These components have the following functions.

- Beacon device
  A beacon device has a sensor and a BLE module. The main functions of the beacon devices are to trigger a scanning application installed on neighboring smartphones using the iBeacon function, and to allow the application to detect the beacon devices themselves. The application assesses the legitimacy of the beacon device by evaluating the hash value after it has detected the beacon device, and starts dedicated operations according to the beacon device's configuration. The proposed system uses built-in sensors in the smartphone or sensors in the beacon devices because measurement information may be affected by differences in implementation conditions or sensor specifications. Therefore, the application transfers measurement information from a beacon device using BLE when the system uses sensors on the beacon device. The beacon device performs measurements depending on predefined rules. For example, it starts the iBeacon function to trigger a neigh-

borhood smartphone device after performing continuous measurement operations.

- Scanning application
  The functions of the scanning application can be roughly classified into searching for a beacon device, acquiring measurement information from the beacon device, taking measurements, and sending measurement information to the management server. The scanning application generally should not search a beacon device not to consume a battery energy. Additionally, iOS does not permit the background processing of applications more than 10 seconds. Therefore, the proposed system employs the iBeacon function to trigger the background processing of the suspend scanning application.

- Management server
  The functions of the management server are information management for each beacon device and storing of measurement information. Since iBeacon uses a UUID, major value, and minor value to identify each beacon device, the management server should handle these parameters for beacon devices and should determine suitable measurement rules of each beacon device. Additionally, the server should store measurement information from the scanning application.

2.3 Signaling Process of Unconscious Participatory Sensing System

Figure 2 shows the detail signaling process of the proposed unconscious participatory sensing system. We have classified into each subprocess: the beacon detection period, the scanning period, the recognition period, the initialization period, the data obtaining period, and the acknowledgement period.

A Beacon detection period: iBeacon packets are periodically transmitted from the beacon device. The period starts when the smartphone enters in the detectable area of iBeacon packets and ends when iOS detects an iBeacon packet.

B Scanning period: the scanning application should scan a BLE advertisement packet. The period stats when the scanning application is activated by iOS and ends when the scanning process of BLE services is completed.

C Recognition period: the scanning application should recognize a proposed service by checking parameters. The period starts when the application received the SCA_RES packet and ends when it transmits CONNECT_REQ packet.

D Initialization period: the scanning application should initialize data communication service of BLE before obtaining information. The period starts when the scanning application starts an initialization process of BLE and ends when an authentification is completed.

E Data obtaining period: the scanning application should obtain information by BLE communication. The period

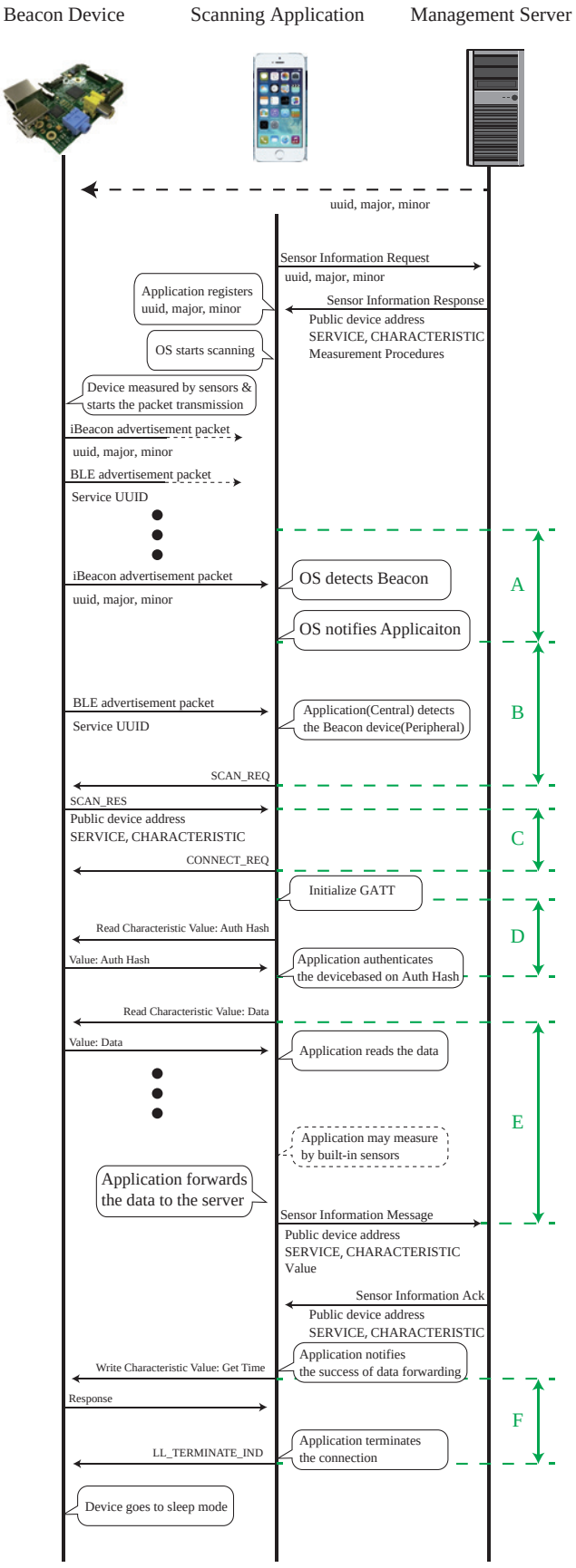


Figure 2: Proposed signaling process.

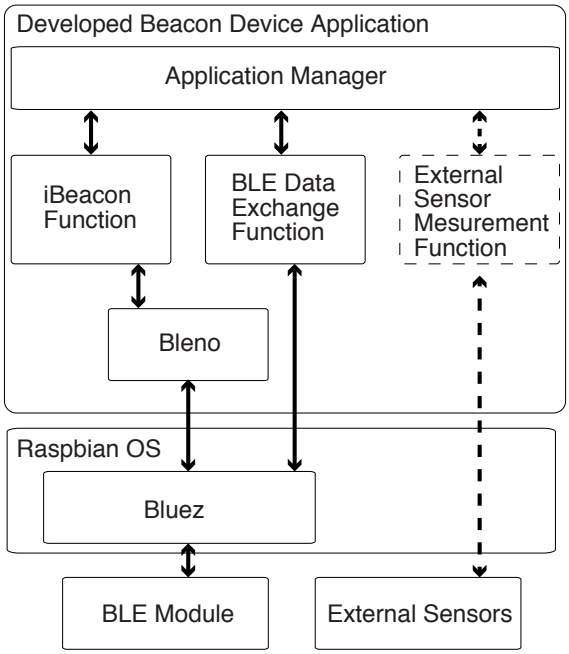


Figure 3: Implementation model of a beacon device.

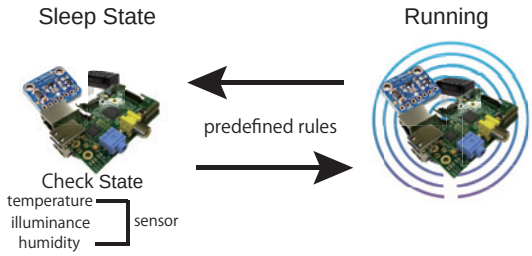


Figure 4: Operation example.

starts when the scanning application starts data communication and ends when the whole information is transferred.

F Acknowledgement period: The beacon device requires an acknowledgement for management of local information. The period starts when the scanning application starts to write an acknowledgement information and ends when it disconnects data communication of BLE.

3 IMPLEMENTATION OF UNCONSCIOUS PARTICIPATORY SENSING SYSTEM

3.1 Beacon Device

We have implemented a beacon device for the proposed method using Raspberry Pi that is an ARM-based microcomputer board. Figure 3 shows the implementation model of the beacon device. The developed application is classified into the application manager, the BLE function and the measurement function with external sensors. The BLE function provides the iBeacon function and the BLE communication function. We have employed Raspbian OS as an operation system, Bluez [20] library for the BLE function, and Bleno [21] library for iBeacon function. As an example sensor, we use a temperature sensor (ADT7410). The prototype beacon device has two operational states: sleep state and running state in Fig. 4 even if Raspberry Pi does not have a power saving mechanism because practical beacon devices should operate by a battery In our future development, we have a schedule to employ special System-on-a-Chip (SoC) modules for BLE to implement a beacon device operating by a battery.

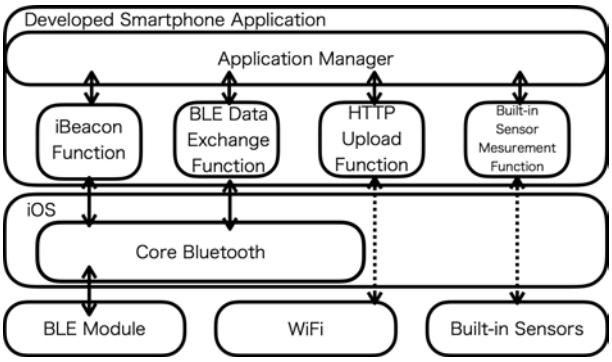


Figure 5: Implementation model of scanning application.

3.2 Scanning Application

Devices for the scanning application should implement BLE module because the proposed system utilizes BLE communication mechanisms. Therefore, we employ iPod touch (fifth generation, version 8.4.1) as a smartphone device. Figure 5 shows the implementation model of the scanning application. The developed application consists of the application manager, the iBeacon function, the BLE data exchange function, the uploading function by hypertext transfer protocol (HTTP) and the measurement function by built-in sensors. Since iOS provides core Bluetooth library, we use it to develop the iBeacon and the BLE data exchange functions. iOS usually suspend applications in background processing. Therefore, the developed application is also suspended by iOS to reduce power consumption. On the contrary, the iBeacon function can activate applications to realize background processing for a limited period around 10 seconds. The developed application also employs the iBeacon function to realize background processing. As a result, we have confirmed that the developed application can work in background processing when iOS detects a designated iBeacon message.

We can develop a similar scanning application for Android OS because Android OS can also receive iBeacon message and permits applications to work in background processing. Additionally, the proposed mechanism can port to Eddystone mechanisms that is the special beacon function proposed by Google [22].

3.3 Management Server

The management server manages identifier information and operational settings for each beacon device. Additionally,

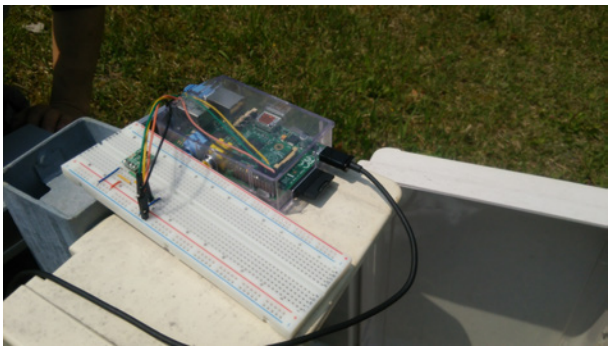


Figure 6: Overview of beacon device.

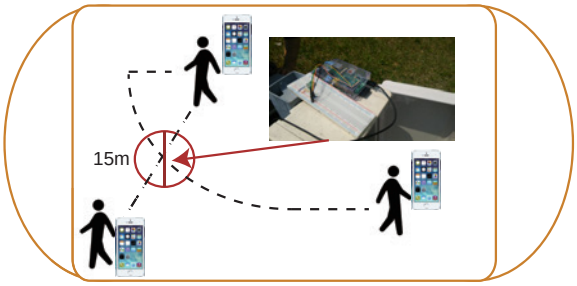


Figure 7: Experiment environment.

it also stores upload measurement information from beacon devices. Since recent smartphone OSs prepare useful APIs for HTTP communication, we employ HTTP to communicate between the management server and the scanning application. As a data format, we use JavaScript Object Notation (JSON). We employ Apache [23] HTTP server and MySQL [24] database server to develop the management server function. We use Raspberry Pi as a prototype hardware due to a portability issue for experimentation.

4 EXPERIMENTS AND EVALUATION

4.1 Objective and Evaluation Points

We have measured the processing period of each task in the proposed unconscious participatory sensing. Due to the limitation of iOS platform, the proposed process should be completed within about 10 seconds. Therefore, we should evaluate that the total process can be completed within the limitation period in iOS platform. We have conducted the experiment in an athletic ground of our university where we can guarantee a Line-of-Sight communication between a beacon device and a smartphone. We have developed a beacon device with a Raspberry Pi and a BLE dongle and a scanning application for an iPod touch device. The scanning information is stored in a database on a laptop PC. Figure 6 shows the overview of the beacon device that was set on the ground. The beacon device implements a temperature sensor and measures an environment at a fixed point. The identification area of iBeacon is about 15 meter according to the advance verification. Therefore, the smartphone moves from 20 meter away

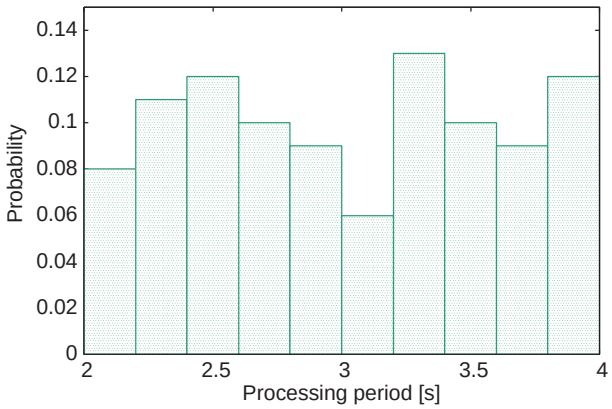


Figure 8: A: Beacon detection period.

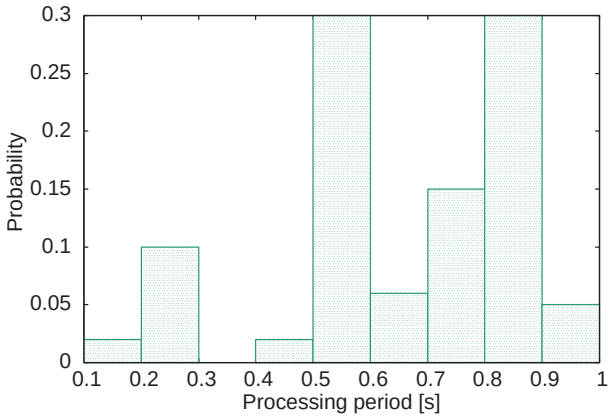


Figure 9: B: Scanning period.

from the beacon device to 15 meter that is the detectable distance of iBeacon. The number of experimental trials is 100 times. Figure 7 shows that experimental environment.

4.2 Experimental Results

A Beacon detection period

Figure 8 shows the probability density of the beacon detection period. The X-axis is the processing period in the second and the Y-axis is the occurence probability of each period. The numerical result shows that iOS requires a few seconds to detect an iBeacon advertisement packet from the device. The main reason of the period is an interval period of an advertisement packet and uncertainness of a transmission range of the packet. Maximum transmission range of typical hardware for BLE is about 100 meter. Therefore, the scanning application has enough period to communicate with the beacon device even if the maximum period is required to detect a packet. Additionally, iOS application can process in a background for around 10 seconds since it detects a packet. Therefore, the beacon detection period is not a critical issue when we assume sufficient transmission ranges.

B Scanning period

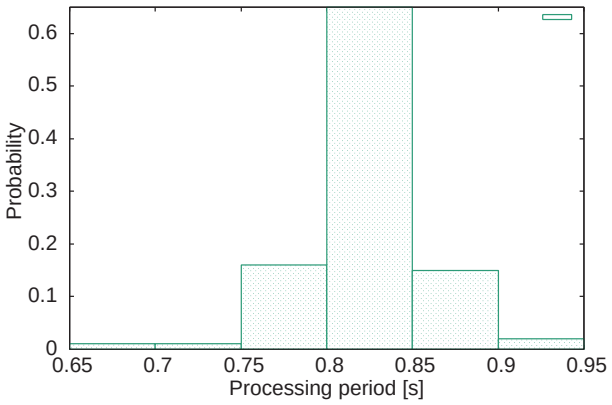


Figure 10: C: Recognition period.

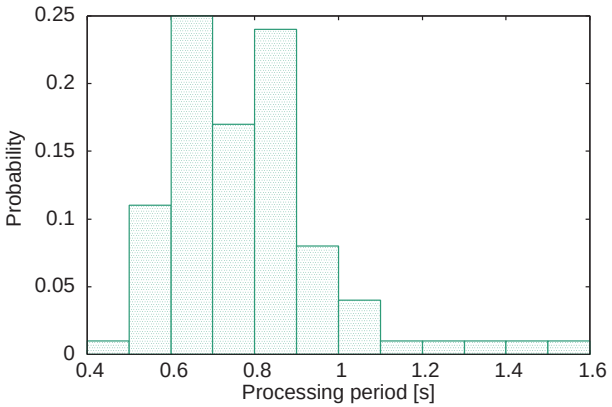


Figure 11: D: Initialization period.

Figure 9 shows the probability density of the scanning period of a BLE advertisement packet. The result shows that the scanning process is completed within 1 second. The scanning period depends on a processing time of an advertisement packet and a transmission timing of a BLE advertisement packet. In the experimental trial, we set 100 milliseconds intervals between iBeacon packet and a BLE advertisement packet.

C Recognition period
Figure 10 shows the probability density of the recognition period of the beacon device. The result shows that the recognition period is completed within 1 second. The recognition period depends on a processing time of a beacon device for a connection setup. From the result, we can find that the developed application can start BLE communication with an adequate background processing period.

D Initialization period
Figure 11 shows the probability density of the initialization period of GATT communication. The result shows that the initialization period is completed within 1.6 seconds. The initialization period depends on a processing time of a beacon device for GATT communication and BLE communication for an authentification.

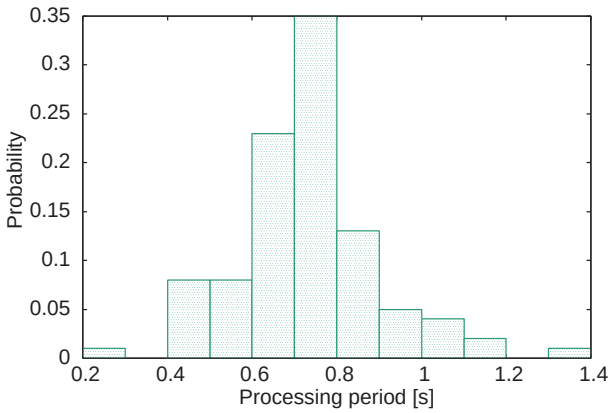


Figure 12: E: Data obtaining period.

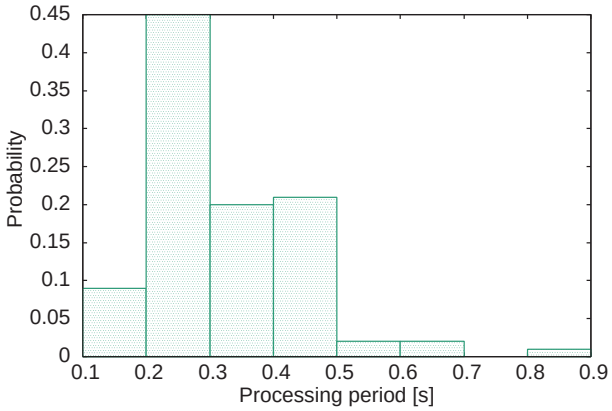


Figure 13: F: Acknowledgement period.

E Data obtaining period
Figure 12 shows the probability density of the data obtaining period. The period includes the data obtaining period from the beacon device and the uploading period to the management server. Therefore, it depends on a processing time of a beacon device for BLE communication and a network delay between a management server and a smartphone. The result shows that the data obtaining period is completed within 1.3 seconds. In practical cases, we should consider longer network delay to the management server. However, we have sufficient period to complete the whole process within 10 seconds. As a consequence, the proposed signaling can work when we consider the number of obtained data carefully.

F Acknowledgement period
Figure 13 shows the probability density of the acknowledgement period. The period depends on a processing time of a beacon device for BLE communication. The result shows that the acknowledgement period is completed within 1 second.

Figure 14 shows the total period of the proposed processing. The result shows that the proposed processing can be completed within about 5 seconds. The maximum processing

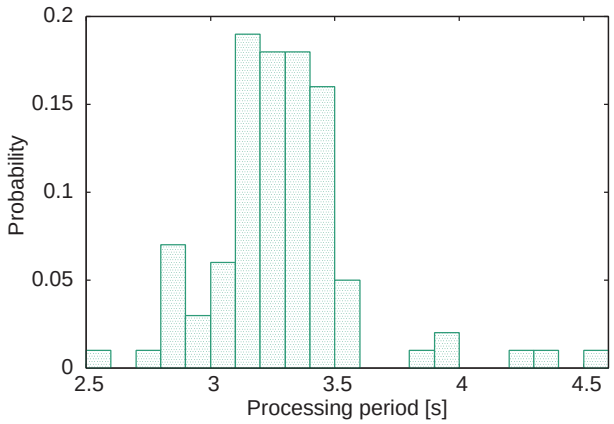


Figure 14: Total processing time.

period of iOS is about 10 seconds. Therefore, the proposed processing can be performed on practical smartphone devices since typical network delay is less than 1 second.

5 ELECTRICITY CONSUMPTION EXPERIMENT IN THE SMARTPHONE APPLICATION

5.1 Objective and Evaluation Points

The benefit of the proposed participatory sensing is that the scanning application can sense an environment and upload sensing data in a background processing. A main reson of power consumption in typical smartphones is a communication module and display. Therefore, we have measured the operation period of a smartphone in both a background processing and a foreground processing. We have developed a special application to perform periodical uploading to a management server. The interval of the uploading is once a second. We have measured the processing period of each task in the proposed unconscious participatory sensing.

5.2 Experimental Results

Figure 15 shows the power consumption of the smartphone device with the foreground processing. The X-axis is the operation period and the Y-axis is the residual battery percentage. The result shows that the smartphone device can work for three hours and 47 minutes.

Figure 16 shows the power consumption of the smartphone device with the background processing. Both X-axis and Y-axis are the same in Fig. 15. The result shows that the smartphone device can work for 15 hours and 27 minutes. The difference between the background processing and the foreground processing is the display use. Therefore, the proposed participatory sensing has a benefit in power consumption comparing to conventional participatory sensing systems.

6 APPLICATION EXAMPLES

The proposed system can apply for various kinds of sensing applications. The following is the example use cases.

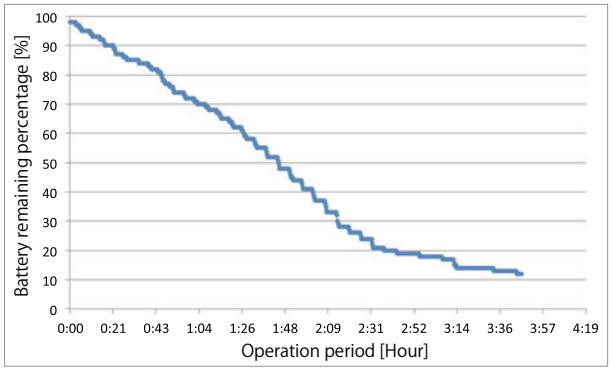


Figure 15: Residual battery (Foreground).

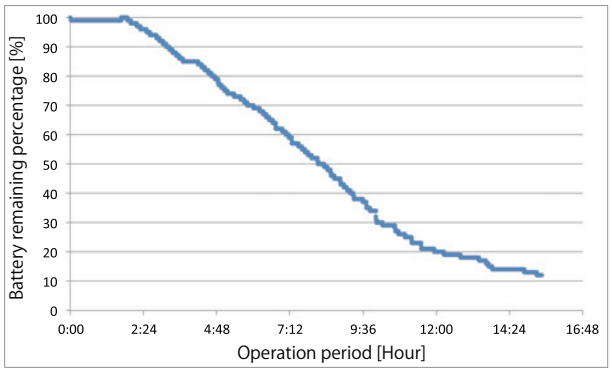


Figure 16: Residual battery (Background).

- Warning system for heatstroke
  The number of people hospitalized suffering heat-related ailments, including heatstroke, is increasing according to global warming. Temperature and humidity are useful information to warn a possibility of heatstroke. The proposed system can realize the warning system for heatstroke by preparing a beacon device with temperature and humidity sensors and an application for smartphones. The beacon device can calculate the waning index by measuring environments periodically, and activate the application on smartphones around the beacon device to inform the risk of heatstroke at the location.

- Personalized environmental control
  Personalized environmental control in facilities is useful technology to improve human productivity. The proposed system can prepare a lot of beacon devices with sensors. Each beacon device can measure accurate environment condition by sensors and inform a facility control application on user's smartphone. Therefore, the application on user's smartphone can recognize the environment condition and control the facility around the users according to predefined rules.

- Crowd sensing
  Human movements in public facilities are useful information to estimate the environmental condition. On the contrary, capturing each human movement is difficult for general sensors. The proposed system can acti-

vate an application on a smartphone to measure human movement by built-in sensors. According to the collected human movement information, the system can estimate crowd condition around a beacon device.

7 CONCLUSION

This paper has developed a prototype implementation of the proposed unconscious participatory sensing system, and has assessed the practicality of the proposed system with the prototype implementation. The prototype implementation uses a Raspberry Pi board and iOS-based hardware. Therefore, we have developed the special beacon application on the Linux OS and the special scanning application on iOS. Due to the limitation of iOS, the scanning application can work in background within 10 seconds. The evaluation results show that the maximum processing period of the total procedures is about 5 seconds. As a result, we have found that the proposed procedures can work well in practical hardwares. Additionally, the proposed mechanism can be port to any beacon mechanism on different OS such as Android OS. Therefore, we believe that it can be a fundamental mechanism to realize unconscious participatory sensing system with the BLE communication.

REFERENCES

[1] F. Viani, P. Rocca, G. Oliveri, and A. Massa, "Pervasive remote sensing through WSNs," 2012 6th European Conference on Antennas and Propagation (EUCAP), pp. 49-50 (2012).

[2] N. D. Lane, S. B. Eisenman, M. Musolesi, E. Miluzzo, and A. T. Campbell, "Urban sensing systems: opportunistic or participatory?," In Proceedings of the 9th Workshop on Mobile Computing Systems and Applications, HotMobile'08, pp. 11-16 (2008).

[3] N. D. Lane, E. Miluzzo, H. Lu, D. Peebles, T. Choudhury, and A. T. Campbell, "A survey of mobile phone sensing," IEEE Communications Magazine, Vol. 48, No. 9 (2010).

[4] P. Mohan, V. N. Padmanabhan, and R. Ramjee, "Nericell: rich monitoring of road and traffic conditions using mobile smartphones," SenSys'08: Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems (2008).

[5] J. Burke, D. Estrin, M. Hansen, A. Parker, N. Ramanathan, S. Reddy, and M. B. Srivastava, "Participatory sensing, Mobile Device Centric Sensor Networks and Applications," In Workshop on World-Sensor-Web (WSW), pp. 117-134 (2006).

[6] D. Wang, M. T. Amin, S. Li, T. Abdelzaher, L. Kaplan, S. Gu, C. Pan, H. Liu, C. C. Aggarwal, R. Ganti, X. Wang, P. Mohapatra, B. Szymanski, and H. Le, "Using Humans as Sensors: An Estimation-theoretic Perspective," In IPSN'14 Proceedings of the 13th International Symposium on Information Processing in Sensor Networks, pp. 35-46 (2014).

[7] E. Niforatos, A. Vourvopoulos, M. Langheinrich, P. Campos, and A. Doria, "Atmos: a hybrid crowdsourcing approach to weather estimation," UbiComp'14: Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing (2014).

[8] A. H. Lam, Y. Yuan, and D. Wang, "An occupant-participatory approach for thermal comfort enhancement and energy conservation in buildings," The 5th International Conference on Future Energy Systems (e-Energy'14) (2014).

[9] M. Budde, R. E. Masri, T. Riedel, and M. Beigl, "Enabling Low-Cost Particulate Matter Measurement for Participatory Sensing Scenarios," In Proceedings of the 12th International Conference on Mobile and Ubiquitous Multimedia (MUM'13) (2013).

[10] B. Hull, V. Bychkovsky, Y. Zhang, K. Chen, M. Goraczko, A. Miu, E. Shih, H. Balakrishnan, and S. Madden, "CarTel: a distributed mobile sensor computing system," In SenSys'06, pp. 125-138 (2006).

[11] F. Zeiger and M. Huber, "Demonstration abstract: participatory sensing enabled environmental monitoring in smart cities," The 13th International Symposium on Information Processing in Sensor Networks (IPSN'14) (2014).

[12] A. Tomasic, J. Zimmerman, A. Steinfeld, and Y. Huang, "Motivating Contribution in a Participatory Sensing System via Quid-Pro-Quo," In CSCW'14 Proceedings of the 17th ACM Conference on Computer Supported Cooperative Work & Social Computing, pp. 979-988 (2014).

[13] D. Zhang, H. Xiong, L. Wang, and G. Chen, "CrowdRecruiter: selecting participants for piggyback crowdsensing under probabilistic coverage constraint," UbiComp'14: Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing (2014).

[14] M. Demirbas, M. A. Bayir, C. G. Akcora, Y. S. Yilmaz, and H. Ferhatosmanoglu, "Crowd-sourced sensing and collaboration using twitter," WOWMOM'10 Proceedings of the 2010 IEEE International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM), pp. 1-9 (2010).

[15] L. Li, Y. Zheng, and L. Zhang, "Demonstration abstract: PiMi air box: a cost-effective sensor for participatory indoor quality monitoring," IPSN'14 Proceedings of the 13th International Symposium on Information Processing in Sensor Networks, pp. 327-328 (2014).

[16] Y. Li, Y. Zhuang, H. Lan, P. Zhang, X. Niu and N. El-Sheimy, "Self-Contained Indoor Pedestrian Navigation Using Smartphone Sensors and Magnetic Features," IEEE Sensors Journal, Vol. 16, No. 19, pp. 7173-7182 (2016).

[17] T. Mizukami, K. Naito, C. Doi, T. Nakagawa, K. Ohta, H. Inamura, T. Hishida, and T. Mizuno, "Fundamental Design for a Beacon Device Based Unconscious Participatory Sensing System," International MultiConference of Engineers and Computer Scientists 2015, Vol. 2 (2015).

[18] iBeacon for Developers, `https://developer.apple.com/ibeacon/`, Retrieved (2014).

[19] BLUETOOTH SPECIFICATION Version 4.0, https://www.bluetooth.org/ja-jp/specification/adopted-specifications, Retrieved (2014).
[20] Official Linux Bluetooth protocol stack, http://www.bluez.org, Retrieved (2014).
[21] A node.js module for implementing BLE (Bluetooth low energy) peripherals, https://github.com/sandeepmistry/bleno, Retrieved (2014).
[22] Eddystone, https://github.com/google/eddystone/tree/master/eddystone-tlm, Retrieved (2016).
[23] Apache, http://www.apache.org, Retrieved (2014).
[24] MySQL, http://www.mysql.com, Retrieved (2014).

(Received September 13, 2015)
(Revised October 30, 2015)

Takamasa Mizukami received the B.I. degree in Information Science from AICHI INSTITUTE OF TECHNOLOGY, Japan in 2014. He is currently working towards the M.B. degree at AICHI INSTITUTE OF TECHNOLOGY. In 2015, he received Excellent paper award at the International Workshop on Informatics. His current research interests include mobile computing.

Katsuhiro Naito received the B.S. degree in Electronics Engineering from Keio University, Japan in 1999, and received the M.S. and Ph.D. degrees in Information Engineering from Nagoya University, Japan in 2001 and 2004, respectively. From 2004 to 2014, Dr. Naito was an assistant professor in the electrical and electronic engineering department of Mie university. He was a visiting scholar in the computer science department of University of California, Los Angeles (UCLA) in 2011. Since 2014, he has been an associate professor in the information science department of Aichi Institute of Technology. His research interests include 5G technologies, vehicular communication systems, Internet of Things (IoT) and Machine to Machine (M2M) systems, overlay networks, and network protocols and architectures.

Chiaki Doi joined NTT DOCOMO, Inc. in 2009. Her research interests include data mining and security for Android application. She received M.E. degree from Keio University, Japan in 2009. She is currently a researcher at Research Laboratories, NTT DOCOMO, Inc. She is a member of Information Processing Society of Japan.

Ken Ohta received the BE, ME, and DE degrees from Shizuoka University, Japan in 1994, 1996, and 1998, respectively. In 1999, he joined NTT Mobile Communications Network Inc.(NTT DOCOMO). His research interests include mobile computing, distributed systems, and system security. He is a member of the Information Processing Society of Japan and of the Institute of Electronics, Information and Communication Engineers.

Hiroshi Inamura joined NTT DOCOMO, Inc. in 1998. His research interests include system research on mobile device and distributed system. Before DOCOMO, he was a research engineer in NTT labs since 1990. From 1994 to 1995, he was a visited researcher in the Department of Computer Science, Carnegie Mellon University. He received B.E., M.E. and D.E. degree in Keio University, Japan. He is a member of IPSJ, IEICE, ACM and IEEE.

Takaaki Hishida received the Ph.D. degree in Enginnering from Gifu University, Japan, in 2000. He is an Associate Professor at the Aichi Institute of Technology, Japan. His research interests include mobile computing, computer networks, educational technologies, broadcast communication and information processing. He is a member of the Information Processing Society of Japan, the Institute of Electronics, Information and Communication Engineers, the Japan Society for Science Education.

Tadanori Mizuno received the B.E. degree in Industrial Engineering from the Nagoya Institute of Technology in 1968 and received the Ph.D. degree in Computer Science from Kyushu University, Japan, in 1987. In 1968, he joined Mitsubishi Electric Corp. From 1993 to 2011, he had been a Professor at Shizuoka University, Japan. Since 2011, he is a Professor at the Aichi Institute of Technology, Japan. His research interests include mobile computing, distributed computing, computer networks, broadcast communication and computing, and protocol engineering. He is a member of Information Processing Society of Japan, the Institute of Electronics, Information and Communication Engineers, the IEEE Computer Society and Informatics Society.

Reducing Interruption Time by Segmented Streaming Data-Scheduling in Hybrid Broadcasting Environments

Tomoki Yoshihisa

Cybermedia Center, Osaka University, Japan
yoshihisa@cmc.osaka-u.ac.jp

Abstract - In hybrid broadcasting environments, clients play streaming data such as video or audio while receiving them. Playback interruptions occur when data reception is later than the start time of data play. Although methods of reducing interruption time have been proposed, these methods have large drawbacks in that the server often broadcasts data that most clients have already received. Hence, I propose an interruption-time reduction method in which the server schedules some segmented streaming data. By broadcasting segments according to a schedule, the server can deliver the segments that many clients have not yet received, thus effectively reducing the interruption time.

Keywords: Broadcast Schedule, Continuous Media, Interruption Time, Video-on-Demand

1 INTRODUCTION

There has recently been a great deal of interest in hybrid broadcasting environments and in streaming delivery using these environments. In this type of delivery, servers deliver streaming data such as video or audio from both broadcasting systems and communication systems. In broadcasting systems (e.g., TV and radio), servers broadcast data according to predetermined broadcast schedules and can deliver data to many clients concurrently. In communication systems (e.g., the Internet), clients can receive the data they want by requesting them directly from servers at arbitrary timings. The hybrid broadcasting environments of these systems are effective for streaming delivery, because clients can receive data from both broadcasting systems and communication systems.

In streaming delivery in hybrid broadcasting environments, clients play streaming data while receiving them from both systems. When data reception is later than the start time of data play, playback interruptions occur. A short interruption time is preferable for viewers to enjoy video or audio. Methods of reducing the interruption time for streaming delivery in hybrid broadcasting environments have been proposed [1]–[6]. Here, *interruption time* means the elapsed time during which playback interruptions occur. This includes the time elapsed from when a request is made to play the data to when the data start to play.

In previous methods, the data are divided into segments of fixed sizes and the server broadcasts the data segment that is

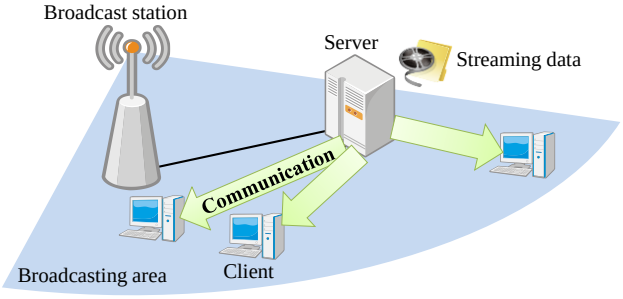


Figure 1: A hybrid broadcasting environment

requested by the client that has the shortest *margin time* (i.e., the margin between the current time and the time when the next interruption occurs). However, this approach has the following drawback. The newest client is the one with the shortest margin time, and this client requests the first segment. The server therefore often broadcasts this segment but many clients have already received it. (For details see Section 3.2.) The server can therefore reduce the interruption time effectively by broadcasting segments that the majority of clients have not yet received.

Here, I propose an interruption time reduction method for hybrid broadcasting environments[1]. In the proposed method, the server can broadcast segments that many clients have not yet received by scheduling some segments dynamically. This is the key point of the proposed method and is different from conventional methods used in hybrid broadcasting environments. Appropriate number of segments scheduled can be estimated by computer simulations. The system can find this by simulating interruption times changing the parameters. By broadcasting segments according to the created schedule, the server does not always broadcast only the first segment, even when new clients request data play. By broadcasting segments that many clients have not yet received, the server can reduce the average interruption times when there are large numbers of clients.

I also describe how to determine the number of scheduled segments in this paper. By adjusting the estimated interruption time so that it is close to the appropriate value, the proposed method can effectively reduce the average interruption time. I evaluate the proposed method under some average request arrival intervals and reveal that the method can reduce average interruption times further than can conventional methods.

[1] This research was supported in part by the Strategic Information and Communications R&D Promotion Programme (SCOPE) of the Ministry of Internal Affairs and Communications, Grant-in-Aid for Scientific Research (B) number 15H02702, and Grant-in-Aid for Challenging Exploratory Research number 26540045.

The rest of the paper is organized as follows. Section 2 explains related work. The proposed methods are presented in Section 3 and evaluated in Section 4. Finally, I present my conclusions in Section 5.

2 RELATED WORK

Methods of reducing interruption times have been proposed before [7]–[12]. After explaining hybrid broadcasting environments, I will introduce some of the methods used for streaming delivery with reduced interruption times in hybrid broadcasting environments.

2.1 Hybrid Broadcasting Environments

Figure 1 shows the hybrid broadcasting environment assumed in this paper. The clients in the broadcasting area can receive data from the broadcasting system. Also, they can ask the server for the data they want and receive them from the communication system. The broadcast station delivers the data via broadcast channels and is managed by the server. The server has streaming data and can broadcast the data to the clients by using the broadcast station. It can also send the data to the clients by unicasting using the communication system. The streaming data consist of segments. The segments are units for playing the streaming data, such as GOPs (Groups of Pictures) for MPEG-encoded streaming data. Examples of streaming delivery in hybrid broadcasting environments are the delivery of video data to TVs or delivery to smart phones connected to the Internet. In this case, the broadcasting system is a terrestrial broadcasting system or a satellite broadcasting system and the communication system is the Internet.

2.2 Methods for Interruption Time Reduction

In the UVoD (Unified Video-on-Demand) method [2], the server broadcasts the streaming data cyclically via each broadcast channel. Because the time to the start of each broadcast cycle is delayed for all broadcast channels, clients get more opportunities to receive the data. When an interruption is about to occur, the client tries to receive the data that causes the interruption directly from the server via the communication system.

In the SSVoD (Super-Scalar Video-on-Demand) method [3], the server broadcasts the data in the same way as with UVoD. However, unlike with UVoD, the server does not send the requested data to the clients until other clients request the same data. After the server has received a number of requests, the server multicasts the requested data to the clients.

In the NBB VoD (Neighbors-Buffering Based Video-on-Demand) method [4], the server broadcasts the data in the same way as with UVoD, but it then uses a peer-to-peer (P2P) approach. Clients receive the desired data from other clients that have already received them. If the other clients do not have the data, the server sends them directly to the client requesting them.

In the above methods, the server broadcasts all of the data repeatedly, although clients can receive some parts of the data from the communication system. In contrast, FC (First segment from Communication), MC-LB (Middle segment from Communication and Last segment from Broadcast), and MC-LC (Middle segment from Communication and Last segment from Communication) methods have been proposed [5]. In these methods, the server predicts the data that the clients will receive from the communication system and eliminates the predicted data from the broadcast schedule. These three methods differ in terms of which data are eliminated. However, the broadcast schedule is static and the methods do not consider the data that the clients have already received.

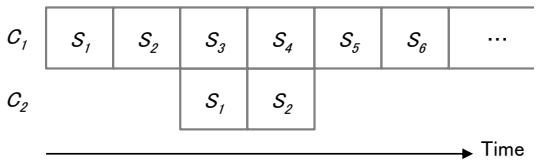


Figure 2: Broadcast schedule using the DHB method

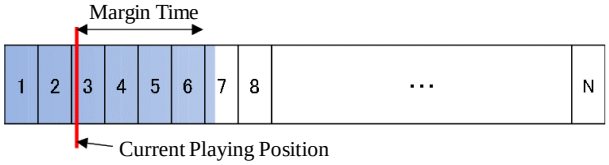


Figure 3: Example of margin time

In the DHB (Dynamic Heuristic Broadcasting [7]) method, the server dynamically broadcasts the requested data by using other broadcast channels. Figure 2 shows a broadcast schedule under the DHB method. C_i (i = 1, 2, ...) denotes the broadcast channels. The data are divided into six segments, S_1, ... , S_6. The server broadcasts S_1, ... , S_6 sequentially via C_1. As an example, suppose that a new client requests data play during broadcasting S_2. The client can receive S_3, ... , S_6 from C_1. To make the interruption time for this client short, the server broadcasts S_1 and S_2 via C_2. However, this method considers data reception only from the broadcasting system, not from the communication system.

In the SET-C (Shortest Extra Time per Client) method [6], the server determines the data to broadcast dynamically, taking into account margin time. As explained in Section 1, the margin time is the time between the current time and the time when the next interruption will occur, and it is calculated from the data that the clients have already received. Figure 3 gives an example of margin time. The vertical red line indicates the current time and the colored area indicates the data that the client has already received. At the time shown in the figure, the client is receiving the 7th segment and the margin time is the time until that segment will start to play. The SET-C method can reduce interruption time, because the probability that the client will avoid interruptions is increased by broadcasting the segment that the client with the shortest margin time requests. In the SET-C method, however, the server broadcasts the first segment when a new client requests data play, because the server determines the next segment to broadcast every time a broadcasting block finishes. Accordingly, the server often broadcasts segments that many other clients have already received. This is the main problem of the SET-C method.

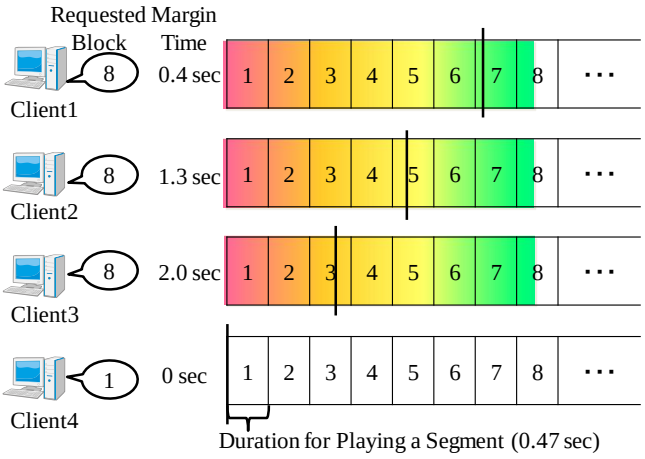


Figure 4: A figure to explain the problem

3 PROPOSED METHOD

This section explains the interruption-time reduction method that I propose. After explaining the assumed system environments, I explain the data delivery for broadcasting systems and for communication systems.

3.1 Assumed System Environments

This research assumes streaming delivery on hybrid broadcasting environments (explained in Section 2.1). Because the server has many streaming data and it is difficult to predict which data the clients will play, the clients do not receive data before they request data play. Clients play streaming data from the beginning to the end continuously, without fast-forwarding or rewinding. Their storage capacity is larger than the size of the requested streaming data. The broadcast station uses one broadcast channel to broadcast one data stream, as occurs in actual systems.

3.2 Main Problem of the Existing (SET-C) Method

In previously proposed methods, the advantage of the broadcasting system (i.e., concurrent delivery of the same data to multiple clients) is the most apparent when the number of clients receiving the same data increases. However, as explained in Sections 1 and 2, the server broadcasts the first segment when new clients request data play, because it is the new clients that have the shortest margin time. Here, the term *new client* means a client that has just started data play from the beginning of the streaming data. Accordingly, the server often broadcasts segments (including the first segment) that many clients have already received. Figure 4 gives an example. Clients 1 to 3 have received the preceding seven segments. The black vertical lines indicate the playing positions of each client. In this situation, the clients are requesting data play at different times and the playing positions differ among the clients, although the data received are the same. Here, suppose the case in which the new client, Client 4, requests data play. Clients 1 to 3 are receiving segment 8 and their margin times are respectively 1.5, 4.8, and 9.2 s. The margin time of Client 4 is 0 s, because this client has not received any segments. In this case, under the previously proposed (SET-C) method, the server broadcasts segment 1, because the margin time of Client 4 is the shortest and Client 4 has requested segment 1. However, the other clients have already received segment 1 and the server cannot exploit the advantage of the broadcasting system. One of the solutions is for the server not to broadcast the segment that the new client requests, but instead to broadcast the segment that many clients have not yet received.

3.3 G-SET-C (Grouped SET-C) Method

The proposed G-SET-C method increases the probability that the server will broadcast the segment that many clients have not yet received, thus reducing interruption time. The server broadcasts the segment that the new client requests after it has broadcast some of the other segments.

The G-SET-C method does not directly consider the number of clients requesting the same segments. The reason is that direct consideration by the server of the number of clients requesting the same segments gives a longer interruption time than the G-SET-C method, as shown in the evaluation results (see the G-MRB method in Section 4.2). This is because clients receive their requested segments from the communication system while the server broadcasts the scheduled segments. Thus, the advantage of the broadcasting system (delivery of data to all clients concurrently) does not operate. However, with the G-SET-C method, by scheduling some of the segments, the server can broadcast segments that many clients have not yet received. This is the key point of the G-SET-C method. The details are given in the next section.

3.3.1 Effectiveness of Grouped Scheduling

The G-SET-C method uses grouped scheduling. That is, the server schedules some segments every time broadcasting of scheduled segments finishes. This avoids the problem described in Section 3.2.

There are two reasons why grouped scheduling achieves the broadcasting of segments that many clients have not yet received. The first is that the server does not consider the margin times of new clients until the next scheduling time. The server can therefore broadcast segments other than the first one, even when a new client comes along. Because many clients have already received the first segment, the server can avoid broadcasting that segment and can instead broadcast segments that most clients have not yet received. In the original SET-C method, the server soon broadcasts the first segment when a new client comes along, because the server schedules only one segment at the finish of every broadcasting segment.

The second reason is that the segment that each client does not have and that is the closest to the current playing position for each client gradually becomes the same as time proceeds. This is because the segment requested by the client with the shortest margin time is not broadcast for a long time, and the broadcast segments are eventually received by all clients. This phenomenon (i.e., in which clients *catch up* with other clients that have started playing the data earlier) is also

observed with the original SET-C method. However, for the first reason given above, clients easily catch up with other clients with the G-SET-C method. For these reasons, with the G-SET-C method, the server can broadcast segments that many clients have not yet received without considering the number of clients requesting the same segments. The main difference between the G-SET-C method and the SET-C method is the number of scheduled segments.

3.3.2 Delivery on Broadcasting Systems

With the G-SET-C method, the server schedules G segments when creating the next broadcast schedule. The server creates broadcast schedules every time the broadcasting of all scheduled segments finishes. The server creates the schedules by considering the segments requested by clients that will have shorter margin times. For this, the server predicts the margin times of all clients.

Let $S(g)$ denote the time to start broadcasting the gth segment (g = 1, ... , G) included in the broadcast schedule, and let $E_i(g)$ denote the predicted margin time of client i. The server can get the actual margin time (=E_i (1)) because the server creates the broadcast schedule at $S(1)$ and this is the current time. Let $R_i(g)$ denote the time for client i to finish receiving the segment requested at $S(g)$. The time needed to broadcast one segment is B_b, and the duration of play of one segment is P_b.

First, when $S(f+1) < R_i(f)$ (f = 1, ... , G-1)—that is, client i cannot finish receiving the requested segment before the broadcasting start time of the f + 1th scheduled segment—the margin time at $S(f+1)$ decreases by the amount of time taken to play one segment. So, $E_i(f+1) = E_i(f) - B_b$. If this is a negative value, $E_i(f+1) = 0$. Next, suppose the case when $R_i(f) < S(f+1)$, that is, client i can finish receiving the requested segment before the broadcasting start time of the f + 1th scheduled segment. When an interruption occurs before the finish time of reception ($E_i(f) < R_i(f) - S(f)$), the client restarts playing the data after its reception. So, $E_i(f+1) = R_i(f) + P_b - S(f+1)$. This always takes a positive value, because $B_b < P_b$ in this research. Otherwise, if ($R_i(f) - S(f) < E_i(f)$), the margin time increases by the amount of time it takes to play one segment. So, $E_i(f+1) = E_i(f) - B_b + P_b$. Hence,

$$E_i(f+1) = \begin{cases} 0 & (S(f+1) < R_i(f), E_i(f) < B_b) \\ E_i(f) - B_b & (S(f+1) < R_i(f), E_i(f) > B_b) \\ R_i(f) + P_b - S(f+1) & (S(f+1) > R_i(f), E_i(f) < R_i(f) - S(f)) \\ E_i(f) - B_b + P_b & (S(f+1) > R_i(f), E_i(f) > R_i(f) - S(f)) \end{cases}$$

In the proposed G-SET-C method, the server schedules the segment that is requested by the client j that satisfies the following equation for each g. N is the set of clients. When some clients have equivalent margin times, the server schedules the segment that was requested by the client that requested the initial data play earliest.

$$E_j(g) = \min_{i \in N} E_i(g)$$

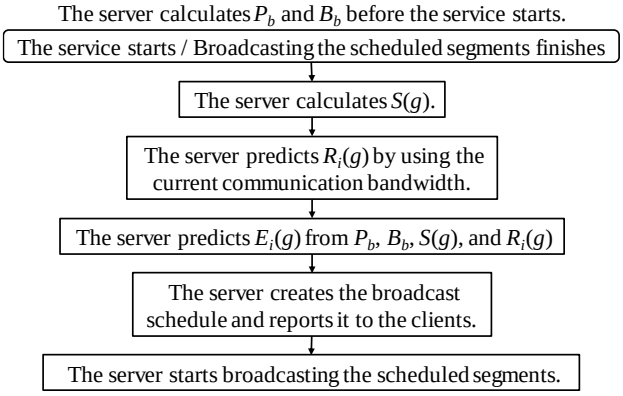


Figure 5: Flow of the G-SET-C method

Figure 5 shows the flow of the G-SET-C method. The server can calculate P_b and B_b before the streaming data delivery service starts, because these values are constant during the service. When the service starts, or when the server finishes broadcasting the scheduled segments, the server starts the process of broadcast schedule creation. First, the server calculates $S(g)$. The value can be calculated from the current time and B_b. After that, the server predicts $R_i(g)$ by using the current communication bandwidth. The server also predicts $E_i(g)$. From the predicted values of $E_i(g)$, the server creates broadcast schedules. The server reports the broadcast schedule to the clients for their determination of the segment to receive from the communication system. The server then starts broadcasting the scheduled segments.

3.3.3 Margin Time Prediction

With the G-SET-C method, to calculate the predicted margin time $E_i(g)$ (g = 1, ... , G), the server needs to predict the time taken to finish receiving segment $D_i(g)$ that client i requests at $S(g)$.

When the client receives $D_i(g)$ from the broadcasting system, the server can calculate $R_i(g)$ by using $S(g)$, because the server can calculate the time to start broadcasting each scheduled segment. For example, if $D_i(g)$ is scheduled to the eth segment in the broadcast schedule, $R_i(g) = S(e) + B_b$.

When the client receives $D_i(g)$ from the communication system, the server predicts $R_i(g)$ by using the communication bandwidth for client i at the time of creation of the broadcast schedule, C_i. C_i is the bandwidth between the server and client i. First, client i may have received part of $D_i(1)$ at that time. So, $R_i(1)$ is given by the remaining data size divided by C_i. Next, for $D_i(f+1)$ (f = 1, ... , G–1), $R_i(f+1) = R_i(f)$ if $D_i(f+1)$ is the same segment as $D_i(f)$. Otherwise, $R_i(f+1)$ is the value obtained by adding the segment size divided by C_i to $R_i(f)$.

The following time sequence shows how the server decides on broadcast segments in the G-SET-C method.

1. The server predicts the margin times. For prediction of the time taken for each client to finish receiving segments from the communication system, the server uses the current bandwidths.
2. The server creates the broadcast schedule on the basis of the predicted margin times.
3. The server notifies all clients of the broadcast schedule.

4. When a client finishes receiving segments from the communication system, the client decides on the next segment to receive from the communication system. For this decision, clients use the notified broadcast schedule.
5. The client starts receiving the decided segment from the communication system. If the client's bandwidth changes greatly, the chosen segment will differ from the server's prediction. Otherwise, it will be the same.

The predicted margin times, including R_i (g), are based on the bandwidths for communication with the client at the time of creation of the broadcast schedules. This is merely a prediction. So, margin times can differ greatly from actual times if the bandwidths change greatly. Otherwise, they are close to the actual times. The sequence therefore does not include cyclic dependency.

3.3.4 Determination of G

The interruption time depends on the number of scheduled segments, G. As confirmed in the evaluation section, the interruption time is reduced further by giving an appropriate value for G. The appropriate value depends on the average arrival interval, the bit rate, etc. and is difficult to estimate. However, the system can give a value close to the appropriate one by performing a computer simulation of the average interruption time. Some network simulators have been developed and we can use these simulators. The system can adjust the value to make it close to the appropriate value while delivering the data.

For example, service providing systems can find the most appropriate value of G by modifying the parameters for simulations based on actual values. They can get actual values by measuring them for a period, e.g. day, week, month, after stopping streaming data delivery services. Simulation results in Section 4 are measured using the author developed simulator. By using such a simulator, the systems can find the value of G that is close to the appropriate value.

3.3.5 Example of Broadcast Schedule Creation

Here, I give an example of the creation of a broadcast schedule by using a simulation result. The simulated situation is shown in Table 1. In this situation, the time to play a segment P_b = 0.469 s and the time to broadcast a segment B_b = 0.125 s. At 1982.899 s after the beginning of the simulation, client 395 is playing segment 22 and the segment that the client does not have and that is the closest to the current playing position is segment 23. The actual margin time is the time between the current time and the time to start playing segment 23 and is 0.445 s.

In this example, G = 2. The server schedules segment 23 as the first scheduled segment since E_i (1) = 0 (i = 394, 396, 397, 398, 399). When the predicted margin times are equivalent, the server schedules the segment requested by the client that requested data play earliest. Here, again, E_i (g) (g = 1, ... , G) is the predicted margin time at the time to start broadcasting the gth scheduled segment. Next, the server predicts the margin times of all clients at the time of the start of broadcasting of the second scheduled segment so as to determine the second segment to be included in the broadcast schedule. The margin times of clients 394 and 395 are equal

Table 1: Example of segment scheduling

Time [s]	Client ID	Playing segment	Closest Non-received segment	Margin time [s] (=E_i (1))
1982.899	394	22	23	0
	395	22	23	0.445
	396	2	3	0
	397	1	2	0
	398	1	2	0
	399	0	1	0
1984.149	394	24	25	0.250
	395	24	25	0.250
	396	5	7	1.000
	397	3	4	0
	398	3	4	0.125
	399	2	4	0.750
1986.649	394	29	40	5.375
	395	29	30	0.500
	396	8	10	0.750
	397	6	10	1.875
	398	6	10	1.875
	399	6	10	1.875

to P_b minus B_b, because they request segment 23. So, at 1982.899 s, E_{394} (2) = E_{394} (1) – B_b + P_b = 0 – 0.125 + 0.469 = 0.344 s and E_{395} (2) = E_{395} (1) – B_b + P_b = 0.445 – 0.125 + 0.469 = 0.789 s. In this simulation, the time for client 396 to finish receiving segment 3 from the communication system R_{396} (1) = 1982.993 s. So, the margin time E_{396} (2) = R_{396} (1) + P_b – S(2) = 1982.993 + 0.469 – (1982.899 + 0.125) = 0.438 s. The times for other clients to finish receiving their requested segments are later than S(2) and E_j (2) = 0 (j = 397, 398, 399). Therefore, the server schedules segment 2, which is requested by client 397, as the second scheduled segment.

Just at 1986.649 s, the server again schedules two segments and first schedules segment 30 as the first scheduled segment, since E_{395} (1) is the shortest and client 395 requests the segment. Next, the server predicts the margin times for each client and decides on the second scheduled segment. At 1986.649 s, the time to start broadcasting the second segment is 1986.649 + 0.125 = 1986.774 s. The predicted margin times for this time are; E_{394} (2) = E_{394} (1) – B_b = 0.5375 – 0.125 = 5.25 s, E_{395} (2) = E_{395} (1) – B_b + P_b = 0.500 – 0.125 + 0.469 = 0.844 s, E_{396} (2) = E_{396} (1) – B_b = 0.750 – 0.125 = 0.625 s, and E_{397} (2) = E_{398} (2) = E_{399} (2) = 1.875 – 0.125 = 1.75 s. These values are the predicted margin times at 1986.649 s and are different from those at 1982.899 s through the symbols are the same. The client that has the shortest predicted margin time is client 396 and the client requests segment 10. So the server schedules segments 30 and 10 at 1986.649 s.

The catch-up phenomenon explained in Section 3.3.1 is observed in the following way. The time of 1982.899 s is immediately after client 399 requests data play, at which time the client requests segment 1. At 1984.149 s, the segments that client 399 has received are the same as those received by clients 397 and 398, and they are requesting segment 4. That is, client 399 catches up with clients 397 and 398. Also, at 1986.649 s, the segments that have been received by clients 396 to 399 are the same, and they are requesting segment 10.

Table 2: Simulation parameter values

Item	Value
Duration of data streaming	25 min
Bit rate	2 Mbps
Broadcast bandwidth	8 Mbps
Clients' communication bandwidth	1 Mbps
Server's communication bandwidth	30 Mbps
Segment size	125.012 Kbytes
Header size	12 bytes

Clients 397 to 399 catch up to client 396. The advantage of the broadcasting systems (i.e., that the server can deliver the same data to all clients) operates well here, because the segment that clients 396 to 399 do not have, and that is the closest to the current playing position, is the same at 1986.649 s.

3.3.6 Delivery via Communication Systems

As in most of the previously proposed methods, clients start receiving blocks from the communication system when they request data play. Clients receive the block that satisfies the following conditions:

- The block will cause interruptions if the client waits for its broadcasting.
- The block can be received faster than reception from the broadcasting system.
- The block is closest to the current playing position.

If there are no blocks that satisfy these conditions, to avoid redundant communication the clients do not receive the blocks from the communication system. They request the next block when they finish receiving each block.

4 EVALUATION

In this section I present the results of simulations used to evaluate the proposed G-SET-C method.

4.1 Evaluation Environments

Table 2 shows the evaluation parameter values. The assumed streaming data are MPEG2-encoded (2 Mbps) movie data with a duration of 25 min. The segments consist of GOPs and the data size is the same as the general GOP data size (0.5 s). The broadcast bandwidth is 8 Mbps, assuming that the broadcasting system is a terrestrial one. The assumed communication system is the Internet. The communication bandwidths for all clients are equivalent. If the total communication bandwidth for the clients exceeds the server's communication bandwidth, the server's communication bandwidth is apportioned equally to each client. The header includes information on the identifiers for the streaming data and segments and the number of segments. The data size for each item of information is 4 bytes and the header size becomes 4 × 3 = 12 bytes. G indicates the number of scheduled segments.

In the simulation, clients request data play when they arrive in the system and leave the system when they finish playing the data. I measured interruption times until the number of clients arriving reached 4000. Interruption times saturate when the number of clients arriving is 4000 and this was a sufficient number of clients to calculate average interruption times.

4.2 Comparison Methods

The performance of the original SET-C method is equivalent to that of the G-SET-C method when $G = 1$. Other comparison methods are explained below.

- BCD-BE-AHB (Broadcast- and Communication-based Delivery-BE-AHB) Method

  This method applies the BE-AHB method proposed in [12] to hybrid broadcast environments. Data delivery on the broadcasting system is similar to the original. The data is divided into some segments. The data sizes for segments are calculated from broadcasting bandwidths. The segments are repeatedly broadcast via each channel. Data delivery on the communication system under this method uses the same algorithm as that under the proposed method. The broadcast schedule is static with this method, whereas that under the proposed method is dynamic.
- G-MRB (Grouped-Most Requested Block) Method

  With this method, the server schedules the top G segments requested by the majority of clients. When there are some segments for which the number of clients requesting the segment is the same, the server schedules the segment that is requested by the client that started data play earliest. When the number of segments requested is less than G, the server broadcasts all requested segments.
- G-LTIT-C (Grouped-Longest Total Interruption Time per Client) Method

  With this method, the server schedules the segments that are requested by the client with the longest interruption time at the time of broadcast of each scheduled segment. Let $I_i(g)$ denote the predicted interruption time for client i at the time of broadcast of the gth scheduled segment (g = 1, ... , G), i.e., $S(g)$. The server can get $I_i(g)$ by asking each client their interruption times. $I_i(f+1)$ (f = 1, ... , G–1) is given by the following equation.

$$I_i(f+1) = \begin{cases} I_i(f) + B_b - E_i(f) & (S(f+1) < R_i(f), E_i(f) < B_b) \\ E_i(f) & (S(f+1) < R_i(f), E_i(f) > B_b) \\ I_i(f) + R_i(f) - S(f) - E_i(f) & (S(f+1) > R_i(f), E_i(f) < R_i(f) - S(f)) \\ I_i(f) & (S(f+1) > R_i(f), E_i(f) > R_i(f) - S(f)) \end{cases}$$

  In the G-LTIT-C method, the server schedules the segment that is requested by the client j that satisfies the following equation for each g. When some clients have the same interruption time, the server schedules the segment that was requested by the client that started data play earliest.

$$I_j(g) = \max_{i \in N} I_i(g)$$

  The G-LITT-C method differs from the SET-C method and the G-SET-C method, even when $G = 1$, because the G-LITT-C method considers interruption time. The SET-C

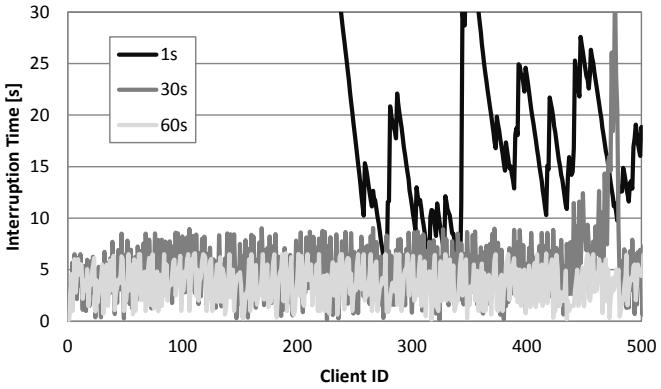


Figure 6: Clients' interruption times

method and the G-SET-C method consider margin time. Moreover, in the G-LITTC method the server schedules several segments, whereas in the SET-C method it schedules only one segment.

4.3 Interruption Time

The simulated interruption times for each client under the proposed G-SET-C method are shown in Fig. 6. The figure shows only the interruption times for the preceding 500 clients. The simulated average arrival intervals are 1, 30, or 60 s. The horizontal axis is the client ID, which is given along with the request time for data play, and the vertical axis is the interruption time. We can see that the interruption time has an upper limit, though it has some dispersion. Hence, the average interruption time is used as an evaluation criterion.

4.4 Influence of the Number of Scheduled Segments

The time taken to create the broadcast schedule depends on the number of scheduled segments, *G*. The probability that segment 1, which is requested by new clients, will be broadcast increases as the interval taken to create the broadcast schedule shortens, and the server often broadcasts segment 1. Hence, I measure the average interruption times under different *G* values. Because the characteristics of the results change with the average arrival interval, the following sections discuss each case individually.

4.4.1 Cases When the Average Arrival Interval is 1 s

Figure 7 shows the average interruption times when the average request arrival interval is 1 s. Figure 8 is an enlargement of Fig. 7.

From these figures, we can see that in most cases the proposed G-SET-C method gives the shortest average interruption time. This is because the server does not always broadcast the segment requested by new clients but instead schedules the segments requested by other clients. Thus, the probability of broadcasting a segment that many clients have not yet received increases. Discussions of each method follow.

With the G-SET-C method, the average interruption time decreases as *G* increases when *G* is less than 90. In such cases, as *G* increases, the server broadcasts more segments that are

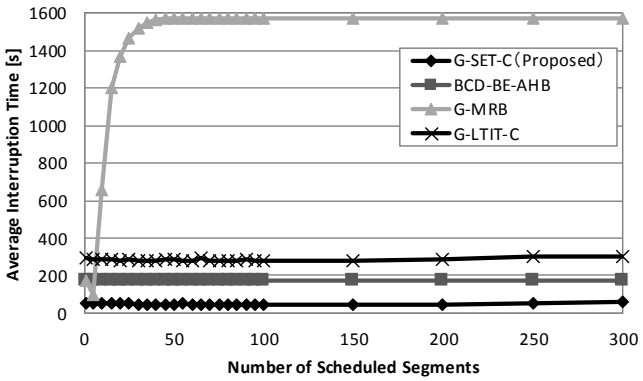


Figure 7: Average interruption time (average request arrival interval 1 s)

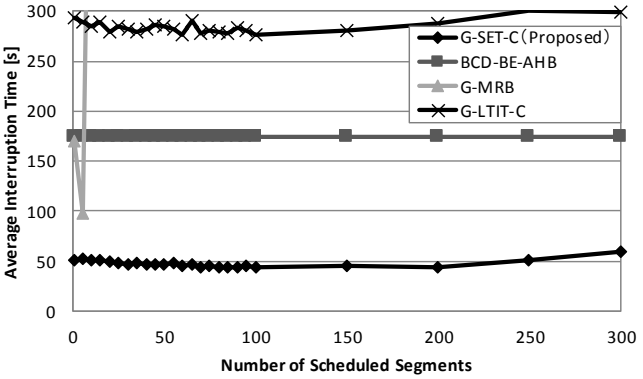


Figure 8: Details of average interruption time (average request arrival interval 1 s)

requested by clients other than new clients, that have just started data play. Therefore, the chances that the server will broadcast segments that many clients have not yet received increases and the average interruption time is reduced effectively. When *G* is greater than 90, the average interruption time increases as *G* increases. In such cases, the time elapsed until the server broadcasts segment 1, which is requested by new clients, increases too much and the average interruption time lengthens. When the value of *G* is much larger, the average interruption time does not change greatly, because new clients receive segment 1 from the communication system while the server broadcasts other segments.

The average interruption time under the G-SET-C method approaches that under the BCD-BE-AHB method when *G* is too large. The reason is that the server schedules the segment that was requested by the client that started playing the data earliest. The broadcast schedules under the proposed method therefore become similar to those under the BCD-BE-AHB method and the average interruption time becomes equivalent when *G* is too large.

The BCD-BE-AHB method does not create the broadcast schedule dynamically, and the broadcast schedule does not depend on *G*. So, the method is not effective and the average interruption time is longer than that under the proposed G-SET-C method.

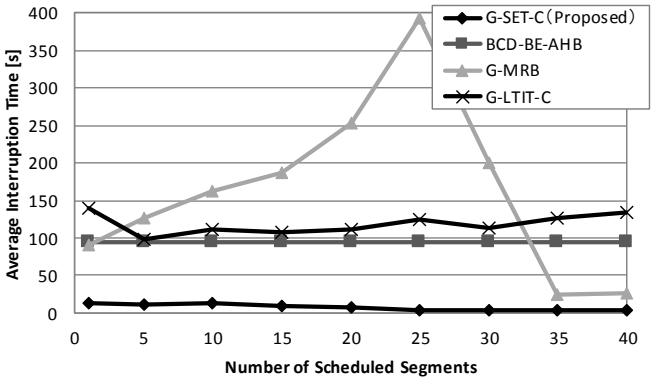


Figure 9: Average interruption time (average request arrival interval 30 s)

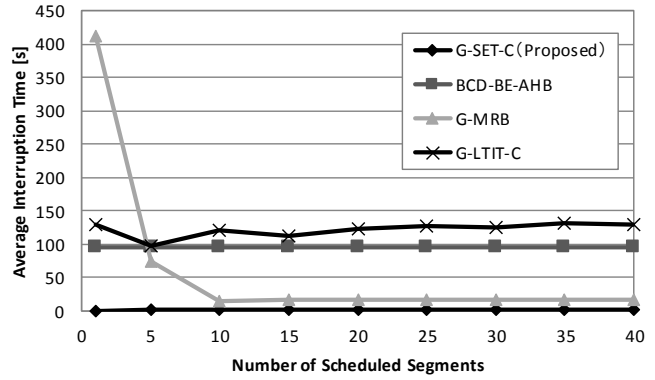


Figure 10: Average interruption time (average request arrival interval 60 s)

With the G-MRB method, the average interruption time is shortest when G is 2. When G is 1 or 2, segments that are requested by clients with long interruption times are included in the broadcast schedule. However, when G is larger than 2, the interval for creating the broadcast schedule lengthens. Therefore, the probability that such segments are included in the broadcast schedule decreases and the interruption time increases. Also, when G is larger than 50, the average interruption time is constant. This is because the number of clients that request segments is less than G and the broadcast schedule does not change even if G increases.

With the G-LTIT-C method, the average interruption time is shortest when G is 85. This is because the server broadcasts more segments requested by clients other than new clients. However, the time elapsed until the server broadcasts the first segment increases as G increases. For the same reason, the value of G that gives the shortest average interruption time is the same as that with the G-SET-C method.

4.4.2 Cases When the Average Arrival Interval is 30 s

Figure 9 shows the average interruption times when the average request arrival interval is 30 s. The proposed G-SET-C method gives the shortest average interruption time in all cases. This is because, for the same reason as when the average arrival interval is 1 s, the server schedules a number of segments and does not always broadcast the segment that the new client requests.

Compared with the result when the average arrival interval is 1 s, the average interruption time under the G-MRB method changes greatly. This is because the probability that several clients request the same segment decreases as the average request arrival interval lengthens. The influence of the number of clients that request the same segment is larger with the G-MRB method than with other methods because the G-MRB method considers the number of segments requested directly.

When G is less than 25, the average interruption time increases as G increases. This is because the interval for creating broadcast schedules lengthens as G increases. However, when G is larger than 25, the interruption time decreases as G increases. This is because the server can broadcast many segments requested by clients at the time of creation of the broadcast schedule.

4.4.3 Cases When the Average Arrival Interval is 60 s

Figure 10 shows the average interruption times when the average request arrival interval is 60 s. In this case the proposed G-SET-C method also gives the shortest average interruption time.

Compared with the results for the other average arrival intervals, one of the main differences is that the average interruption time under the G-MRB method decreases as G increases when G is small. This is because clients can receive their requested segment from the communication system while the server broadcasts the scheduled segments. Because the server can broadcast more segments requested at the time of creation of the broadcast schedule, the average interruption time decreases as G increases. When G is larger than 10, the average interruption time is constant, because the number of segments requested is less than G. Therefore, in these cases the server actually does not schedule G segments.

5 CONCLUSION

In this paper, I proposed a segmented streaming data scheduling method for streaming delivery in hybrid broadcasting environments. Different from conventional methods, in the proposed G-SET-C method, the server schedules some segments considering the margin time until the next interruption. This approach leads to the broadcasting of segments that many clients have not yet received. The G-SET-C method thus contributes to further reduce interruption time. I also described how to determine the appropriate number of scheduled segments in this paper. My evaluation revealed that in many cases the proposed method could reduce the average interruption time further than with conventional methods.

In the future, I am planning to propose a method that considers stopping data play and applies the P2P data delivery techniques.

REFERENCES

[1] V. Gopalakrishnan, B. Bhattacharjee, K. Ramakrishnan, R. Jana, and M.K. Vernon, "CPM: Adaptive Video-on-Demand with Cooperative Peer Assists and Multicast," IEEE INFOCOM 2009, pp. 91-99 (2009).

[2] J.Y.B. Lee, "UVoD: An Unified Architecture for Video-on-Demand Services," IEEE Communication Letters, Vol. 3, No. 9, pp. 277-279 (1999).

[3] J.Y.B. Lee and C.H. Lee, "Design, Performance Analysis, and Implementation of a Super-Scalar Video-on-Demand System," IEEE Transactions on Circuits and Systems for Video Technology, Vol. 12, Issue 11, pp. 983-997 (2002).

[4] T. Taleb, N. Kato, and Y. Nemoto, "Neighbors-Buffering-based Video-on-Demand Architecture," Signal Processing: Image Communication, Vol. 18, Issue 7, pp. 515-526 (2003).

[5] M. Umezawa, T. Yoshihisa, T. Hara, and S. Nishio, "Interruption Time Reduction Methods by Predicting Data Reception for Streaming Delivery on Hybrid Broadcasting Environments," Proc. IEEE Pacific Rim Conference Communications, Computers and Signal Processing, pp. 185-190 (2011).

[6] S.W. Carter, J.F. Paris, S. Mohan, and D.D.E. Long, "A Dynamic Heuristic Broadcasting Protocol for Video-on-Demand," Proc. IEEE International Conference on Distributed Computing Systems, pp. 657-664 (2001).

[7] T. Yoshihisa, "Dynamic Data Broadcasting Methods for Streaming Delivery on Hybrid Broadcasting Environments," Proc. International Workshop on Advances in Data Engineering and Mobile Computing, pp. 470-475 (2015).

[8] D.L. Eager and M.K. Vernon, "Dynamic Skyscraper Broadcast for Video-on-Demand," Proc. of International Workshop on Advances in Multimedia Systems, pp. 18-32 (1998).

[9] H. Kim and H.Y. Yeom, "Dynamic Scheme Transition Adaptable to Variable Video Popularity in a Digital Broadcast Network," IEEE Transactions on Multimedia, Vol. 11, No. 3, pp. 486-493 (2009).

[10] J.B. Kwon. and H.Y. Yeom, "Adjustable Broadcast Protocol for Large-scale Near Video-on-Demand Systems," Computer Communications, Vol. 28, No. 11, pp. 1303-1316 (2005).

[11] Q. Zhang and J.F. Paris, "A Channel-based Heuristic Distribution Protocol for Video-on-Demand," Proc. IEEE International Conference on Multimedia and Expo, Vol. 1, pp. 245-248 (2002).

[12] T. Yoshihisa and S. Nishio, "A Division-based Broadcasting Method Considering Channel Bandwidths for NVoD Services," IEEE Transactions on Broadcasting, Vol.59, Issue 1, pp. 62-71 (2013).

(Received September 26, 2015)
(Revised March 10, 2016)

Tomoki Yoshihisa received his Bachelor's, Master's, and Doctor's degrees from Osaka University, Osaka, Japan, in 2002, 2003, 2005, respectively. Since 2005 to 2007, he was an assistant professor at Kyoto University. In January 2008, he joined Cybermedia Center, Osaka University as a senior lecturer and in March 2009, he became an associate professor. From April 2008 to August 2008, he was a visiting researcher at University of California, Irvine. His research interests include video-on-demand, broadcasting systems, and webcasts.

Log Data Collection of Real-time Control System using Fault Tree Analysis

Naoya Chujo†, Akihiro Yamashita‡, Nobuyuki Ito‡, Yukihiko Kobayashi‡, and Tadanori Mizuno†

†Faculty of Information Science, Aichi Institute of Technology, Japan
‡Mitsubishi Electronic Engineering Co., Ltd., Japan
ny-chujo@aitech.ac.jp
{Yamashita.Akihiro, Ito.Nobuyuki, Kobayashi.Yukihiko}@ma.mee.co.jp
mizuno@mizulab.net

Abstract - The increasing complexity of embedded systems in information and communication technology causes a problem with locating faults during system failures. One reason for this problem is that complicated systems consist of so many components that basic log data do not contain useful information about abnormal system behavior by faulty components. Since available time resources in real-time systems are limited, we cannot use much time for logging all data to specify the faulty components.

In this paper, we present a logging method of real-time control system using Fault Tree Analysis for locating the faulty components. Fault Tree Analysis is applied for assumed system failures, and then specific data in fault trees are defined to locate the faulty components. Log tasks are scheduled to collect the specified data in cooperation with system tasks. Once the assumed system failure is observed during system operation, the related log tasks wake up and collect the specified data to diagnose system faults. The experimental results have shown that specified data related to faulty components are collected by log task and the overhead for logging is predictable.

Keywords: Fault Tree Analysis, Log Data, Fault Diagnosis, Real-time, Embedded System

1 INTRODUCTION

In recent years, while the embedded software in systems such as those in automobiles or medical devices has grown increasingly complicated, numerous real-time control systems have been developed. These complexities have caused a range of problems, including reduced productivity and increasing difficulty in pinpointing fault origins. Thus, improving the reliability of such systems has become an important objective. Logging data has become a popular method to improve the reliability.

The primary role of the log data associated with faults is to record items such as fault occurrence time and the nature of the fault. In addition, fault-diagnosis functions allow the collection of log data describing the basic status of the system at the fault occurrence time. However, by using only this basic level of log data, it remains difficult to determine the factors that caused the fault to occur.

In this paper, we present a logging method for a real-time control system using a fault tree analysis to locate the faulty components. For assumed system failures, specific data in fault trees are used to locate the faulty components.

To summarize our contribution, we find that log data collection based on fault tree analysis is useful for the identification and tracking of the failed component through our case studies. Further, the time required for data collection is predictable, and the log task is able to be scheduled not to disturb control tasks. Although constructing a fault tree of complex system requires long time, fault trees are assumed to be given in this study.

The remainder of this paper is organized as follows.

In Section 2, we review related work and discuss case studies involving automotive fault diagnosis and reliability improvements by using a fault analysis model. In Section 3, we describe our proposed method to collect log data based on the fault tree analysis for a real-time system. In Section 4, experiments using a miniature car and a motor control system are presented. The results show that the faults of a real-time system can be detected by the proposed method. Moreover, the overhead for collecting log data is evaluated, because predictable overhead is important for a real-time system. In Section 5, we discuss these results. Our conclusions are presented in Section 6.

2 RELATED RESEARCH

Real-time control systems for applications such as automobiles and medical devices require extremely high reliability, and various methods exist for improving system reliability. In this section, we introduce a case study of automotive fault-diagnosis functionality. We also discuss the fault tree analysis (FTA) [1] to improve reliability.

2.1 Fault Diagnosis in Automobiles

The field of automotive fault-diagnosis functionality provides a case study of log data collection in a real-time control system. For example, on-board diagnosis (OBD) [2], which is a tool for diagnosing system status in automobiles, consists of an automatic diagnosis via the computers embedded in automobiles. Most automobiles in service today are equipped with OBD. Figure 1 shows a diagram of OBDII, which is a second-generation OBD system. OBDII monitors and diagnoses Electronic control units (ECUs) via the controller area network (CAN). ECU is an embedded system that controls the automotive electrical and electronic system.

The basic scope of OBD encompasses monitoring, data recording, and communication. Monitoring is checking for the flashing of malfunction indicator lamps (MILs) when rel-

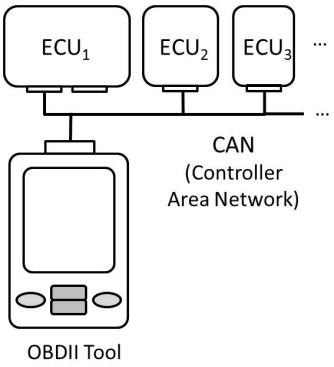


Figure 1: Second-generation on- board diagnosis system (OBDII).

evant items meet fault detection criteria. Data recording and communication are the recording of a code (in the event of a fault) and the specifying of the fault. A diagnostic tool can subsequently be used to read this code.

It is generally believed that monitoring frameworks in OBD systems should grow increasingly sophisticated in the future. Monitoring frameworks include an automotive fault-diagnosis function that allows computers to detect faults.

When an automobile detects a fault, it records diagnostic trouble code (DTC) that encodes information on the sensors involved in the fault, the events that have been diagnosed and the basic status of the automobile. The basic status data is called Freeze Frame Data (FFD).

However, information of the FFD is limited to basic automobile information, such as engine rotation, temperature of cooling water, and O_2 sensor output. Although the FFD includes much necessary information, it is not sufficient for diagnosing the individual controllers used in modern automotive systems. A high-end modern automotive system has dozens of ECUs using an estimated sixty-five million lines of code [3].

Moreover, the real-time nature of controllers makes it difficult to diagnose faults, because the period of a control cycle by sophisticated controllers is on the order of milliseconds. It is much shorter than the period of the FFD, which has a recording cycle time of hundreds of milliseconds, and most typically 500 milliseconds [4], depending on the system.

2.2 Fault Tree Analysis to Improve Reliability

FTA was developed for the reliability assessment and safety analysis of military systems [1]. To date, FTA has been widely applied to various types of industrial plants and transportation systems. In an FTA, the lower event and/or combinations of such events are investigated to determine whether they caused the higher event or the final top-level event, which is undesirable system failure. The tree format, called a fault tree, is defined to express the relation between the lower events and higher events.

As an example of an FTA, we consider the case of car halt. Figure 2 shows the fault tree (FT) diagram corresponding to the halt of an electric vehicle (EV).

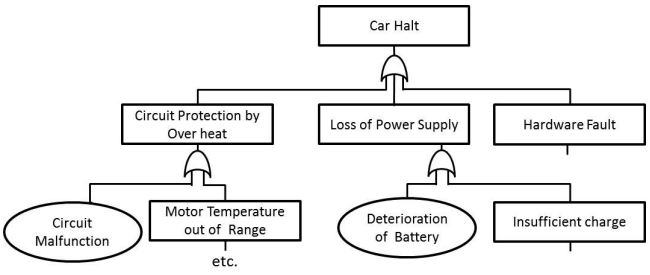


Figure 2: FT diagram for the case of car halt.

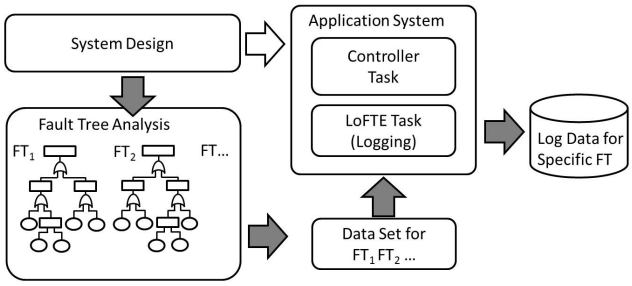


Figure 3: Schematic diagram of the LoFTE method.

We note first that the event at the top of the diagram is the undesirable event within the system. In this case, we positioned car halt as the top-level event. Possible causes for this top-level event include circuit protection by overheat, loss of power supply, and hardware fault. Among these events, possible causes for circuit protection by overheat include circuit malfunction and motor temperature out of range. We proceed in this way to trace the possible causes of each event.

The rectangles in the diagram show intermediate events that could be possible causes for the upper-level events. The circles in the diagram show basic events that could cause system faults.

By specifying events that could cause system faults (top-level events), an FTA enumerates the causes of lower-level events that lead to top-level events (system faults). This enumeration can then be used to analyze the causes and fault events that contributed to the system failure.

FTA was developed as an analysis technique in system design for the prevention of system failure, but it is often used for the purpose of finding the cause after a system failure. In this paper, FTA is used to data collection for the cause analysis of failures in real-time system.

Enhanced FTA approaches to analyze large scale complicated computer-based systems were developed. For example, dynamic fault tree (DFT)[5] is an effective method for the analysis of computer-based systems. DFT provides a means for combining FTA with Markov analysis for sequence dependent problems. DFT method works for fault tolerant computing systems by introducing the functional dependency gates and the spare gates. Another extension of FTA is condition-based fault tree analysis (CBFTA) [6]. CBFTA starts with the known FTA, but by condition monitoring system, CBFTA updates failure rates and applies to the FTA. CBFTA recalculates

periodically the top event failure rate, and then the system reliability is monitored in undergoing system. These enhanced FTA approaches also work for the cause analysis of failures in real-time system.

However, real-time control systems may have the unexpected failure during operation, and some failures are not reproduced in the off-line. We think that it is because of incompleteness or difficulty of modeling control systems including mechanical, electrical, and computer parts. Therefore, there are needs to perform the log data collection and off-line diagnosis.

3 PROPOSED METHOD FOR COLLECTING LOG DATA

In this section, we present our proposed method, which we named *Log data collection using Fault Tree Expansion* (LoFTE) [7]. We then describe our method by using a simple example of a system fault and the FT, after which we demonstrate a case of fault-event identification. Throughout this research, single fault was assumed.

3.1 Philosophy of LoFTE Method

The LoFTE method implements FTA at the system design stage and determines both the collection schedule and the data to be collected at the time of fault detection. Then, during the system operation stage, log data are collected, with proper consideration paid to the real-time nature of the system at the fault detection time. Thus, this method aspires to achieve real-time fault diagnosis by collecting the log data during system operation. More specifically, our method executes the following procedures at the system design stage and during system operation.

Procedure at the system design stage:

1. conduct FTA based on the system design specifications.
2. Within the FT, specify the data to be collected by software.
3. Store collection schedules for the specified data within the control software.

Procedure at the system operation stage:

1. Identify fault(s) that arise during control tasks.
2. Report the log-data-collection task responsible for information associated with the fault(s)
3. The data associated with the fault(s) are then collected as log data based on the relevant scheduling.

We further describe the system by referring to the example depicted in Fig. 3. The results of the FTA at the system design stage are stored as FT_1, FT_2, and so on. In this case, we assume that a fault in FT_2 has occurred and we initiate the log-data collection task. The log-data collection task collects the log data displayed for FT_2.

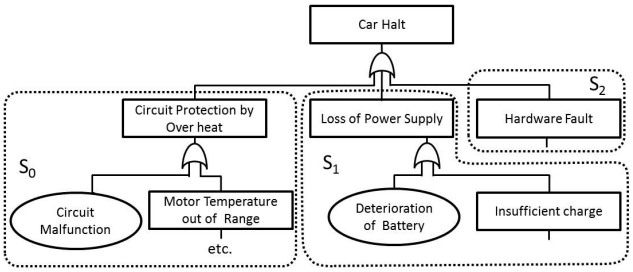


Figure 4: Subtrees for the case of car halt.

3.2 Collection of Log Data

In the LoFTE method, the collection of log data is executed based on information of the fault tree. Once a faulty event is detected, all events in the fault tree should be recorded. However, the log data of a large number of events are not preferable, because the work of collecting log data takes a long time and makes it difficult to analyze faulty events. To this end, the control software is analyzed to determine its module structure [8]. The collection of log data is executed based on the structure [9].

In this subsection, we use the example of the EV discussed in subsection 2.2 as an example of a real-time control system and consider the collection of log data.

The fault tree diagram in Fig. 2 has three subtrees: S_0 of the control circuit module, S_1 of the power supply module and S_2 of the other hardware module. These subtrees are shown in Fig. 4. We assume that the control software for each module is designed to be independent from the others. Therefore, the work of collecting log data for the fault tree is divided into three parts.

4 EXPERIMENTS

In this section, we describe our experiments conducted to test whether it is possible to identify the cause of a fault from the collected sensor data and the FT. The overhead of the proposed method is estimated through experiments.

4.1 Experiment with Miniature Car

In the first experiment, we used a miniature car as an experimental device.

4.1.1 Experimental Device

The experimental device was a 1/10-scale miniature car, Robocar 1/10 (hereafter referred to as RoboCar) for the automotive platform (AP) [10], which was designed to be a research platform for autonomous driving. Figure 5 show a photograph of the experimental device, and Figure 6 show a structural diagram of the RoboCar system.

RoboCar is equipped with a V850/FG4 CPU [11], multiple input devices, including a three-axis acceleration sensor, eight infrared range sensors, a three-axis gyro sensor, two field-effect transitor (FET) temperature sensors, a motor encoder,

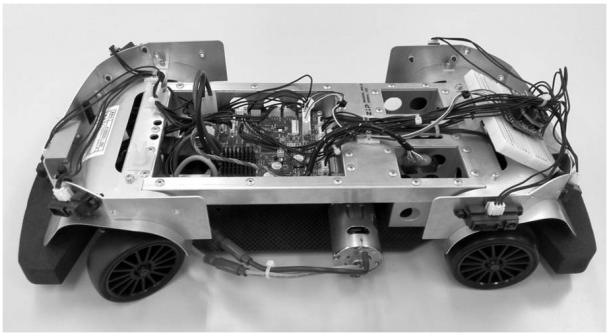


Figure 5: RoboCar 1/10 for AP.

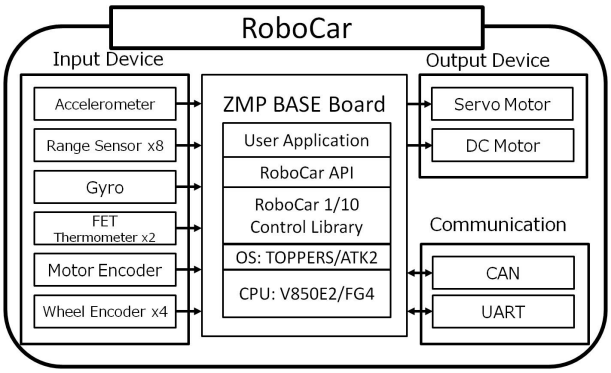


Figure 6: Structure of the RoboCar control system.

and four wheel encoders. Sensor data from all of these input devices are obtained from the RoboCar API. The device is also equipped with two output devices: a servo motor and a DC motor. These devices are controlled by the RoboCar API. The communications specifications correspond to the CAN [12] and the Universal Asynchronous Receiver Transmitter (UART).

4.1.2 Software Used in the Experiment

In this experiment, we used TOPPERS/ATK2 [13], a real-time OS designed for next-generation automotive embedded systems. This OS was designed by the Center for Embedded Computing Systems at Nagoya University (NCES) and was designed to comply with AUTOSAR [14], a standard specification for automotive embedded software.

4.1.3 Experimental System

The experimental system used in this work consists of the RoboCar (the system in which the fault occurs) and a computer that monitors sensor data transmitted from the RoboCar via Bluetooth. Figure 7 shows a schematic diagram of the experimental system[15].

Three tasks were implemented as TOPPERS/ATK2 applications: our proposed LoFTE task, a control task, and a communication task. The LoFTE task collects log data from the sensors installed on the RoboCar. The control task controls the various system actuators on the basis of the sensor data

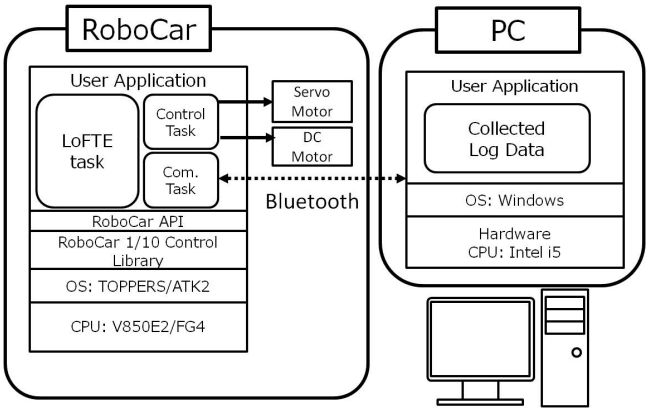


Figure 7: Schematic depiction of the experimental system.

collected by the sensor-data collection task. For example, this task controls the motor torque to ensure that the driving motor maintains a constant velocity. The communication task transmits the data collected to the computer. The system was realized by periodically executing these tasks at a cycle period of 100 msec.

In this experiment, Bluetooth wireless communication was used for the communication. The rate of data transmission via Bluetooth was at 115,200 bits per second (bps) to allow the computer to monitor data transmission from the RoboCar.

4.1.4 Assumed Fault and Experimental Procedure

In the first experiment, the RoboCar traversed a circular track in the clockwise direction until its motion was obstructed manually to bring the RoboCar to a halt. We take the halt of the RoboCar as our fault event in this experiment. We designed one specific fault event for this experiment.

At the system design stage, we anticipate the causes of the RoboCar halt and prepare the fault tree diagram, shown in Fig. 8. The shaded events indicate the causes of the fault assumed in our experiment.

We assigned codes for all events of the fault tree according to the level. The event of Level 0, car halt, was assigned the code, 0x01. Three events of Level 1, circuit protection by overheat, loss of power supply, and hardware fault, were assigned codes, 0x10, 0x11, and 0x12, respectively. Two events of Level 2, circuit malfunction and motor temperature out of range, were assigned codes, 0x20, and 0x21, respectively. Three events of Level 3, cooling fault, overload of motor, and fault of temperature sensor, were assigned codes, 0x30, 0x31, and 0x32, respectively. Two events of Level 4, overcurrent and overvoltage, were assigned codes, 0x40, and 0x41, respectively. The codes of the shaded events of Fig. 8 are listed in Table 1.

In this experiment, we measured the velocity of both wheels powered by the driving motor, the current, and the FET temperature to record the status of the RoboCar. We configured the overheat protection function to go into operation at 80.0 ℃, and the current limit function at 10 A.

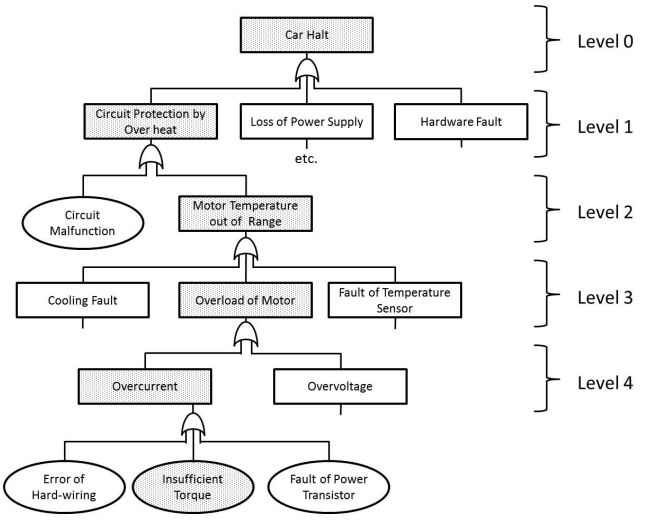


Figure 8: FT diagram for the case of RoboCar halt.

Table 1: Assigned codes for the case of RoboCar halt.

Event	Log Data	Code
Car Halt	Wheel Speed	0x 01
Circuit Protection by Overheat	Flag of Circuit Protection	0x 10
Out of Range of FET Temp.	FET Temperature	0x 21
Overload of Motor	Flag of Overload of Motor	0x 31
Overcurrent	Motor Current	0x 40

4.1.5 Experimental Results

Table 2 shows the experimental results. The RoboCar motion was obstructed at 3'59"7 after measurements began. The driving motor continued to operate after this time, but stopped 10 seconds later.

In the interval prior to 3'59"7, the drive velocity of the RoboCar (which was moving clockwise) was approximately 0.8 m/sec for the left wheel and 0.6 m/sec for the right wheel. At the time the car reached the obstruction, the velocity of both wheels fell to 0.3 m/sec. At 4'00"4, the left wheel velocity jumped to 1.4 m/sec, while the right wheel velocity fell to 0.0 m/sec, which shows the right wheel was locked. At 4'09"7, the driving motor halted and the velocity of both wheels fell to 0.0 m/sec.

The current of the RoboCar ranged from 1.0 A to 4.0 A in the interval prior to 3'59"7, but it jumped to a maximum value of 7.7 A after encountering the obstruction. At 4'00"4, the current rose to 10.2 A, which showed the overcurrent.

The temperature remained roughly constant until 3'59"7, but it jumped to 80.2 ℃ at 4'09"4 after encountering the obstruction. The driving motor halted at 4'09"7, and the temperature remained high.

The event codes 0x40 and 0x31, which corresponded to overcurrent and overload of motor, were detected at 4'00"04. In addition, 0x21and 0x10 , which corresponded to motor

Table 2: Detected events induced by RoboCar halt.

Event	Time	wheel Speed (Left) [m/sec]	wheel Speed (Right) [m/sec]	Current [A]	FET Temp. [℃]	Detected Event Code	Execution Time for Logging [μsec]
Normal	0'00 "0 – 3'59"6	0.8	0.6	1.0~4.0	49.5	-	4
Drive Obstruction	3'59"7	0.3	0.3	7.7	50.5	-	4
Overcurrent and Overload	4'00"4	1.4	0.0	10.2	52.2	0x40, 0x31	124
Out of Range of FET Temp.	4'09"4	1.4	0.0	7.6	80.2	0x40, 0x31 0x21, 0x10	139
Car Halt	4'09"7	0.0	0.0	9.9	79.9	0x40, 0x31 0x21, 0x10 0x01	149

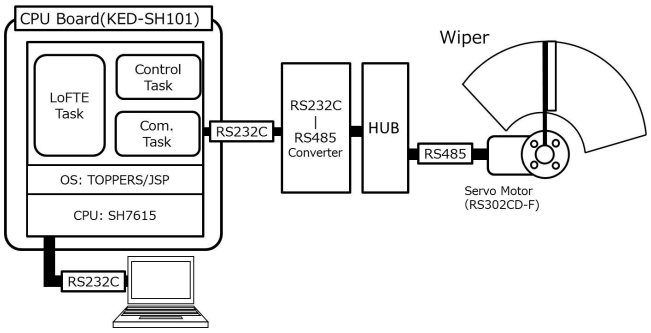


Figure 9: Wiper control system.

temperature out of range and circuit protection by overheat, were detected at 4'09"04. The code 0x01, which corresponded to car halt, was detected at 4'09"07.

In addition, we saw that the execution time for logging increased after detection of the codes. The execution time was 4 μsec at 3'59"7. The execution time jumped to 124 μsec at 4'00"4, and it increased with the number of detected codes.

4.2 Experiment with Wiper Control System

In the second experiment, we used a wiper control system as the experimental device.

4.2.1 Experimental System

Figure 9 shows a structural diagram of the wiper control system. The device is equipped with a CPU board (KED-SH101), a servo motor, and a monitoring computer. These are connected via RC232C and RS485 networks. The servo motor can be controlled from KED-SH101 via the networks.

In this experiment, we used TOPPERS/JSP [16], a real-time kernel designed for embedded systems.

Three tasks were implemented as TOPPERS/JSP applications (the LoFTE task, a control task, and a communication task), and the system was realized by periodically executing these tasks. The LoFTE task was executed every 700 ms. The LoFTE task collects log data from the servo motor, and the data are sent to the monitoring computer.

4.2.2 Assumed Fault and Experimental Procedure

For this experiment, we designed two specific fault events that cause a wiper halt. The moving wiper operation was obstructed manually, and it caused the servomotor to increase the motor current to keep it moving. Because the motor temperature increased due to the increased current, it finally brought about a wiper halt. We take the obstruction as the first fault event in this experiment.

As our second fault event, we designed a communication fault event. When the wiper system was moving to the left and right, the communication was interrupted intentionally by setting the disable flag of the communication register, bringing no response from the servo motor. The interruption also caused a wiper halt.

At the system-design stage, we anticipate the causes of a wiper halt and prepare the fault tree diagram shown in Fig. 10. The shaded events indicate the causes of the fault assumed in our experiment.

We assigned the codes for all events of the fault tree according to level. All of the assigned codes are listed in Table 3. The event of Level 0, wiper halt, was assigned code 0x01. Three events of Level 1, circuit protection by overheat, communication fault, and hardware fault, were assigned codes 0x10, 0x11, and 0x12, respectively. Six events of Level 2, motor temperature out of range, circuit malfunction, check sum error, parameter read error, response time out, and negative acknowledge, were assigned codes 0x20, 0x21, 0x22, 0x23, 0x24, and 0x25, respectively. Six events of Level 3, cooling fault, overload of motor, fault of temperature sensor, no response from servo motor, excess number of retries, and disconnection of network, were assigned codes 0x30, 0x31, 0x32, 0x33, 0x34, and 0x35, respectively. Four events of Level 4, overcurrent, overvoltage, loss of motor power supply, and communication disable flag, were assigned codes 0x40, 0x41, 0x42, and 0x43, respectively.

In this experiment we measured current, voltage, temperature and the control register of the servo motor. We configured the overheat protection function to go into operation at 60 ℃, and we set the maximum current at 100 mA. The communication control register and the number of retries were monitored in this experiment.

Table 3: Assigned codes for the case of wiper halt.

Event	Code
Wiper Halt	0x01
Circuit Protection by Over heat	0x10
Communication Fault	0x11
Hardware Fault	0x12
Motor Temperature out of Range	0x20
Circuit Malfunction	0x21
Check Sum Error	0x22
Parameter Read Error	0x23
Response Time Out	0x24
Negative Acknowledge	0x25
Cooling Fault	0x30
Overload of Motor	0x31
Fault of Temperature Sensor	0x32
No Response from Servo Motor	0x33
Excess Number of Retries	0x34
Disconnection of Network	0x35
Overcurrent	0x40
Overvoltage	0x41
Loss of Motor Power Supply	0x42
Communication Disable Flag	0x43

Table 4: Detected events induced by wiper obstruction.

	Time	Current [mA]	Temp. [℃]	Detected Event Code	Execution Time for Logging [msec]
No Operation	0'00 "00 – 0'3"32	0	41	-	1
Normal Operation	0'03"99 – 0'07"08	12~14	42	-	1
Overcurrent	0'07"68	611	42	0x40, 0x31	1
Wiper Halt	1'14"45	568	60	0x40, 0x31 0x20 0x10, 0x01	2

4.2.3 Experimental Results

Table 4 shows the experimental results for the first fault event. The wiper motion was started at 0'03"99 after measurements began. Then, the wiper's motion was manually obstructed at 0'07". The motor continued to operate after this time, but stopped at 1'14"45.

The current of the wiper system was in the range from 12 mA to 14 mA in the interval from 0'03"99 to 0'07"08, but it jumped to a value of 611 mA after encountering the obstruction, which showed the overcurrent because the current limit was 100 mA. The overcurrent was kept to 1'14"45.

The temperature remained roughly constant at 41 ℃ or 42 ℃ until 0'07"08, but it rose to 60 ℃ at 1'14"45 after encountering the obstruction.

Event codes 0x40 and 0x31, which corresponded to overcurrent and overload of motor, were detected at 0'07"68. In addition, 0x20, 0x10, and 0x01, which corresponded to temperature exceeding limit (60 ℃), circuit protection by overheat, and wiper halt, were detected at 1'14"45.

The execution time for logging was 1 msec after detecting two event codes at 0'07"68, then it increased to 2 msec when detecting five event codes at 1'14"45. Because the time resolution of TOPPERS/JSP kernel is 1 msec, we could not see that the execution time increased with the number of detected codes.

Table 5 shows the experimental results for the second fault event. The wiper started to swing at 0'07"18 after measurements began. The communication was obstructed at 0'16"47 after measurements began. The wiper halted at 0'17"07.

In the interval prior to 0'15"79, the register code of the communication register was 0x30 (00110000), which showed

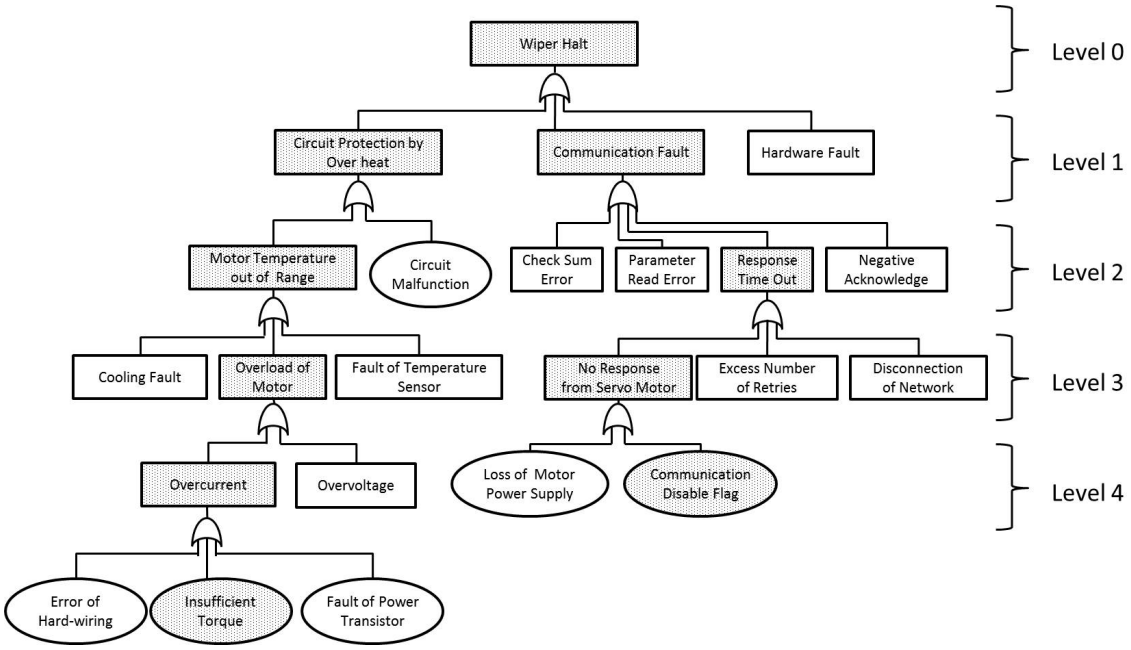


Figure 10: Assumed fault in wiper control system.

Table 5: Detected events induced by communication error.

	Time	Register Value	Num. of Retry	Detected Event Code	Execution Time for Logging [msec]
No Operation	0'00 "00 – 0'7"18	0x30	0	-	1
Normal Operation	0'07"19 – 0'15"79	0x30	0	-	1
Communication Fault	0'16"47	0x00	0	0x43, 0x33	1
Wiper Halt	0'17"07	0x00	6	0x43, 0x33 0x24 0x11, 0x01	2

that the transmit enable bit and the receive enable bit were set, and the communication was enabled. At 0'16"47, the register code changed to 0x00 (00000000), which showed that the communication was disabled. At 0'17"07, the number of retries for the communication reached 6, which showed that the maximum number of retries was exceeded.

Event codes 0x43 and 0x33, which corresponded to disable flag of communication and no response from servo motor, were detected at 0'16"47. In addition, 0x24, 0x11, and 0x01, which corresponded to response timed out, communication fault, and wiper halt, were detected at 0'17"07.

5 DISCUSSIONS

Our experiments confirmed that the LoFTE method was capable of logging the sequences of faulty events up to the top-level event. The sequences of faulty events provide useful information for identifying the cause of the fault.

In the first experiment using the RoboCar, we see first that over current and overload of the motor can be detected, then the events on the path of the fault tree can be detected, and finally the RoboCar halt (the top-level event) can be detected.

In the second experiment using the wiper control system, we see that the communication disable flag (the basic event) and no response from the servomotor can be detected as the second fault event. After that, the events on the path of the fault tree can be detected, and finally the wiper halt (the top-level event) can be detected.

Regarding the load of LoFTE tasks, the maximum execution time was estimated at 149 μsec. This time was less than 1% of that of the control cycle period, 100 msec in the first experiment. However, more importantly, the overhead of real-time system was predictable [17]. Because the execution time for logging increased with the number of the detected codes, the load of the LOFTE task was predictable by the heights of the fault trees.

We assumed a single fault at a time in this study, and so the order of logged events was straightforward on the path of the fault tree. However, in the case of multiple faults, the order of logged events would be complicated.

It should be noted that constructing the complete FT of a complex system is difficult. It is also difficult to evaluate the validity of FT. However, even the FT is not complete, it does not mean useless. The incomplete FTs would have unexpected events or wrong edges between events. If the unexpected event had influence on system failure, they would have implicit edges to some FTs. Such events are considered to be detectable as the other events of FT. Otherwise, the events have no influence on the system, and they are negligible. In the case of the wrong edges, they make the fault analysis difficult, and another logging is necessary to verify the edges between the events.

For this reason, our future work will be to access data sets of the system for logging through remote networks, as shown in Fig. 11. We would be able to access real-time controllers by wireless communication and modify data sets for logging

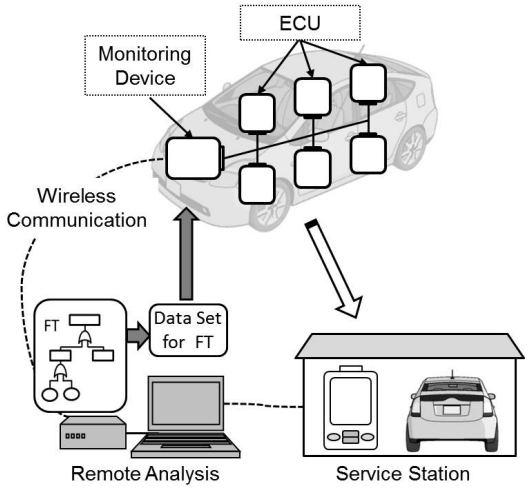


Figure 11: Remote log collection system.

unexpected faults depending on the designer's request. Fault diagnosis of real-time controllers through remote networks would be available. After detecting the faulty device or software, the system could be fixed at service stations or by a wireless network. This capability would reduce the problem of designing complete fault trees of complex systems prior to market release.

6 CONCLUSIONS

In this study, we proposed the LoFTE (Log data collection using Fault Tree Expansion) method for log data collection using a fault tree analysis. We then conducted experiments involving a miniature car and a wiper control system.

The results of our experiments indicate that the LoFTE method is capable of logging the sequences of faulty events. Therefore, it is useful for identifying the cause of a fault. The load of the logging task is small and predictable by using fault trees. The log task is able to be scheduled without disturbing control tasks.

Our future work will be to access and control data sets for the logging of fault trees through remote network.

ACKNOWLEDGMENTS

This research is supported by JSPS KAKENHI Grant Number 26330074.

We extend our deepest gratitude to Associate Professor Shinya Honda and the members of the Center for Embedded Computing Systems at Nagoya University (NCES), who provided extensive support and resources regarding TOPPERS/ATK2 during the completion of this work.

We are also particularly grateful for the assistance and support given by Shogo Fukuoka, Hiroki Kitagawa and Kazuhiro Tsujita.

REFERENCES

[1] N. Leveson, Safeware, System Safety and Computers, ACM, pp. 305-313 (1995).

[2] J. Shaeuffele, and T. Zurawka, Automotive Software Engineering - Principles, Processes, Methods and Tools, SAE International, pp. 118-125 (2005).

[3] Mentor Graphics Inc., "News and Views 2014 Spring/ Vol. 9," accessed June 13 (2015). http://www.mentorg.co.jp/news_and_views/automotive/2014/spring.html

[4] Toyota Motor Corp., Toyota Prius New Model Reference and Repair Manual, Parts Number NM12B1J, pp. IN-36-38 (2010).

[5] R. Gulati and J. Dugan, "A Modular Approach for Analyzing Static and Dynamic Fault Trees," IEEE Proc. of Reliability and Maintainability Symposium, pp. 57-63 (1997).

[6] D. Shalev and J. Tiran, "Condition-based Fault Tree Analysis (CBFTA): A New Method for Improved Fault Free Analysis (FTA), Reliability and Safety Calculations," Reliability Engineering and System Safety, Vol. 92, No. 9, pp. 1231-1241 (2007).

[7] N. Chujo, A, Yamashita, N. Ito, Y. Kobayashi, and T Mizuno, "Real-time Log Collection Scheme using Fault Tree Analysis," Proceedings of International Workshop on Informatics 2015, pp. 109-114 (2015).

[8] M. Takahashi, "A Study of Fault Tree Analysis for Embedded Software," IPSJ Technical Reports, Vol. 2013-SE-182 No. 24, pp. 1-8 (2013).

[9] S. Fukuoka, et al., "Scheduling for Logging of Real-time Control Systems," Proceedings of the 76th National Convention of IPSJ, No. 1, pp. 59-60 (2014).

[10] ZMP Inc., "RoboCar 1/10 for AP (Automotive Platform)," accessed June 13 (2015). https://www.zmp.co.jp/products/robocar-110_package_option#ap

[11] Renesas Electronics, "Data Sheet V850E2/FG4 32-bit Single-Chip Microcontroller (2013)," accessed June 13 (2015). http://documentation.renesas.com/doc/DocumentServer/R01DS0139ED0100_FG4.pdf

[12] Robert Bosch GmbH, "CAN Specification Version 2.0 (1991)," accessed June 13 (2015). http://www.kvaser.com/software/7330130980914/V1/can2spec.pdf

[13] TOPPERS Project, "TOPPERS/ATK2," accessed June 13 (2015). https://www.toppers.jp/en/atk2.html

[14] S. Frst, "AUTOSARTechnical Overview," AUTOSAR Open Conference. (2011).

[15] H. Kitagawa, et al., "Logging for Fault Diagnosis of Realtime Control System," Proceedings of Workshop on Informatics 2014, pp. 158-164 (2014).

[16] TOPPERS Project, "TOPPERS/JSP kernel," accessed June 13 (2015). http://www.toppers.jp/en/jsp-kernel.html

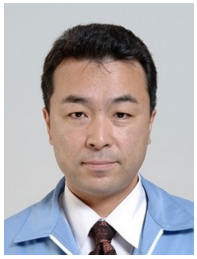
[17] P. Dodd and C. Ravishankar, "Monitoring and Debugging Distributed Real-time Programs," Software, Practice and Experience 22.10 pp. 863-877 (1992).

(Received September 30, 2015)
(Revised March 7, 2016)

Naoya Chujo received his M.S. degree in information science in 1982 and Ph.D. degree in electrical engineering in 2004 from Nagoya University, Japan. He is a professor in the Faculty of Information Science of Aichi Institute of Technology, Japan. His research interests are in the area of embedded system and automotive electronics. He is a member of IEEE, the Information Processing Society of Japan (IPSJ), the Institute of Electronics, Information and Communication Engineers (IEICE), the Institute of Electrical Engineers of Japan (IEEJ), and Informatics Society.

Akihiro Yamashita received the B.E and M.E. degree in Precision Engineering from the Kyoto University, Japan in 1977 and 1979, respectively. In 1979, he joined Mitsubishi Electric Corp. He received the M.E. degree in Mechanical Engineering from the Carnegie Mellon University, USA, in 1987. In 2012, he joined Mitsubishi Electric Engineering Corp. Since 2014, he has been a graduate school student at Shizuoka University, Japan. His research interests include control engineering and computer science. He is a member of IPSJ.

Nobuyuki Ito joined Mitsubishi Electric engineering Co.,Ltd. in 1979. He's a manager of the drive system control engineering department. His expertise field is in the factory automation for which a servo control system was utilized.



Yukihiko Kobayashi joined Mitsubishi Electric engineering Co., Ltd. in 1986. He is a senior engineer of the servo software engineering section. His specialized field is built-in software.

Tadanori Mizuno received the B.E. degree in Industrial Engineering from the Nagoya Institute of Technology in 1968 and received the Ph.D. degree in Engineering from Kyushu University, Japan, in 1987. In 1968, he joined Mitsubishi Electric Corp. From 1993 to 2011, he had been a Professor at Shizuoka University, Japan. From 2011 to 2016, he had been a Professor at the Aichi Institute of Technology, Japan. Since 2016, he is an Affiliate Professor at the Aichi Institute of Technology, Japan. His research interests include mobile computing, distributed computing, computer networks, broadcast communication and computing, and protocol engineering. He is a member of Information Processing Society of Japan, the Institute of Electronics, Information and Communication Engineers, the IEEE Computer Society and Consumer Electronics, and Informatics Society.

Invited Paper: An Anomaly Detection System for Equipment Condition Monitoring

Makoto Imamura*, Michael Jones**, and Daniel Nikovski **

*School of Information and Telecommunication Engineering, Tokai University, Japan
** Mitsubishi Electric Research Laboratories, USA
*imamura@tsc.u-tokai.ac.jp
**{mjones, nikovski}@merl.com

Abstract - In industrial domains, equipment condition monitoring (ECM) has attracted much attention as the Internet of Things (IoT) has been emerging and growing. This paper describes our anomaly detection system for ECM to solve the requirements in terms of development efficiency, sensor big data management, and feature extraction characteristic of sensor data, which we have experienced in the development of practical systems. First, we proposed a novel relation based query language TPQL (Trend Pattern Query Language) as a basis for declaratively describing the conditions that anomaly sensor data satisfy in order to improve development efficiency and the maintainability of programs. TPQL provides a convolution operator and a time interval join as important common operations for anomaly detection. The former is for extracting features of time series segments, and the latter is for time consuming preprocessing, such as missing value completion and merging data with different sampling periods. Second, we introduce function libraries for TPQL in order to solve the problems in terms of sensor big data management and feature extraction. In terms of sensor big data management, we select key-value store database for accumulating sensor data and provide data transformation functions among key-value, stream and relation to enable the selection of data type in accordance with various processes such as storage, aggregation among relations, and time series processing. Furthermore, we propose an exemplar learning method that can summarize the features of training time series with a smaller set of exemplars for enabling fast anomaly detection even for big sensor data. In terms of feature extraction, we propose a novel leg vibration analysis that can extract the global trend pattern in time series with local fluctuations, so that it can capture the vibration behavior depending on a given amplitude and a given window size.

Keywords: Equipment Condition Monitoring, Anomaly Detection, Feature Extraction, Sensor Data

1 INTRODUCTION

As the Internet of Things (IoT) [1] has been emerging and growing, sensor big data that are streamed from various equipment in power plants, industrial facilities, and buildings can be made available for monitoring, diagnosis, energy-saving, productivity improvement, quality management, and marketing. As a result, industry has paid much attention to the use of big sensor data generated from equipment or facilities in order to create a smarter society.

Equipment Condition Monitoring (ECM) is a typical service that uses big sensor data, and machine learning techniques for big sensor data are key technologies to make ECM smarter [2].

We have published elemental technologies for anomaly detection [3]-[6]. This paper illustrates a total anomaly detection system exploiting the elemental technologies. The rest of our paper is organized as follows. Section 2 describes the problems to be solved, and then introduces an anomaly detection system based on Trend Pattern Query Language (TPQL) to solve them. Section 3 and section 4 discuss leg vibration analysis and exemplar learning, respectively, as key technologies of our system. Section 5 shows an evaluation of our system.

2 ANOMALY DETECTION SYSTEM-BASED ON TPQL

The problems of anomaly detection for ECM can be largely grouped into three categories: development efficiency, management of sensor big data, and accuracy of anomaly detection.

In order to solve the above problems, first, we have proposed a relation based query language TPQL (Trend Pattern Query Language) [3]-[4] to express constraints in time series data for anomaly detection. TPQL is an SQL-like language, and it can help programmers describe application-dependent conditions for anomaly sensor data with a common function library. And then, we implemented an anomaly detection system based on TPQL with Java, and applied it to real applications.

Figure 1 shows the overall structure of our system. A TPQL interpreter calls the common function library, key-value store database, and relational database, by means of a given TPQL script. A TPQL script describes anomaly conditions with the help of the function library, which is designed on the basis of the requirements that we have encountered through our experience in the development of practical ECM systems. Furthermore, configuration management information as input makes TPQL scripts separated from the parameters dependent on actual facilities. This separation improves maintainability of TPQL scripts.

This section mainly illustrates the problems to be solved, our solution to those, and the difference with respect to related work. In terms of the implementation details of TPQL, please refer to [3] and [4].

2.1 Development Efficiency

Programming for data preprocessing that is specific to each application has a large proportion in the development effort for anomaly detection functions. Therefore, we aim to propose a general relation based query language, TPQL, which has a common function library containing functions that are commonly used in our development of anomaly detection systems, while also maintaining the semantics of SQL. TPQL provides two basic common functions for preprocessing: a convolution operation and a time-interval join.

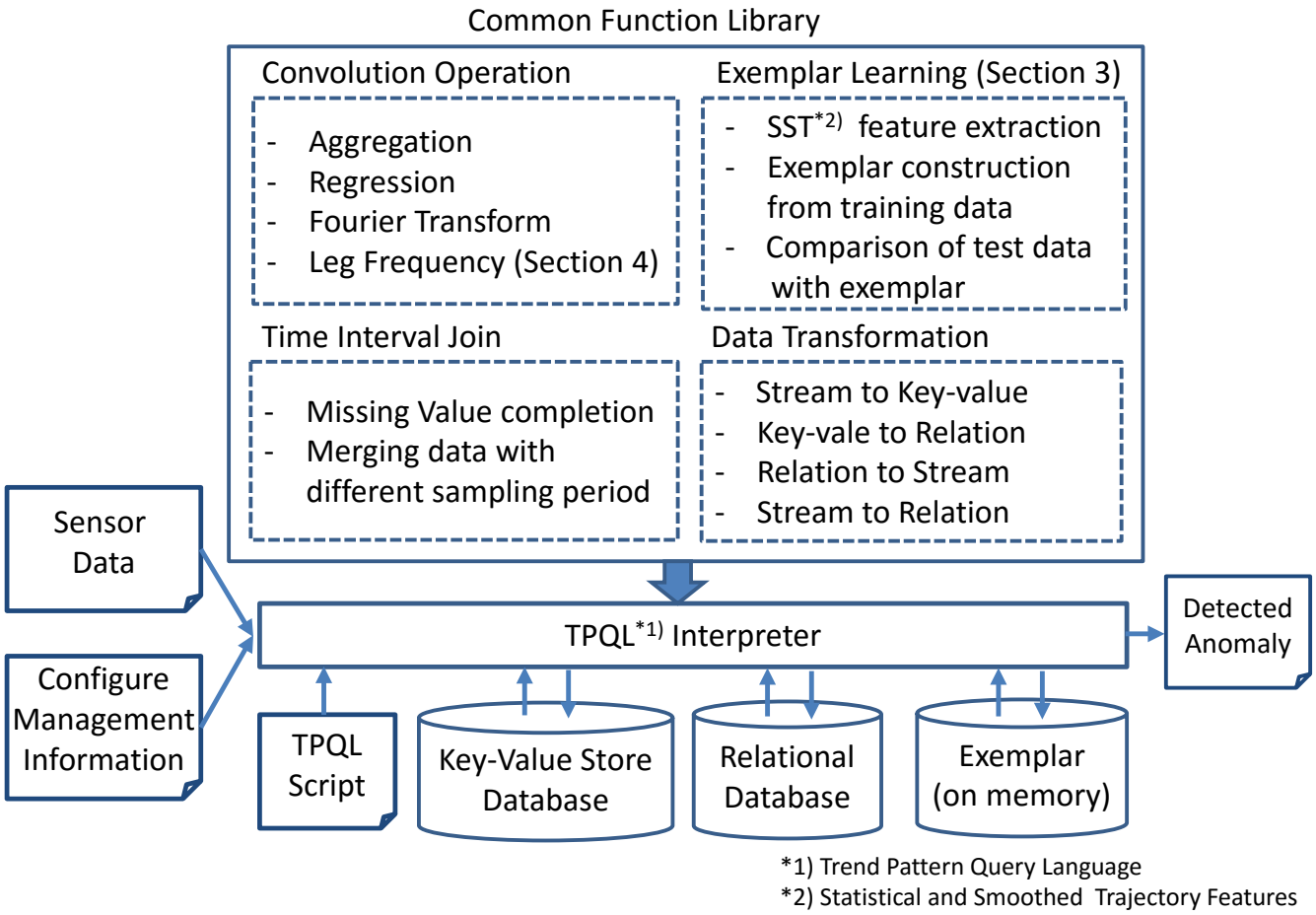


Figure 1: An Anomaly Detection System based on TPQL

(1) Convolution operation

One of the common operations of anomaly detection for sensor data is sliding window processing, because a time series segment for a given time period is a basic processing unit. In a typical anomaly detection procedure, features are extracted from each time segment, and then the extracted features from test data are compared with those for training data. If the features of the test data are different from those of the training data, the test data is determined to be anomalous.

A convolution operation is the cumulative sum of the repeated multiplication of sliding window segments and a given feature function. The output of a convolution operation is a time series which consists of feature values of each time segment of input time series. A convolution operation makes the description of a procedure clear by separating feature functions from the calculation.

Typical feature functions in convolution operations are aggregation, regression, and Fourier transform.

(i) Aggregation functions

TPQL provides standard aggregation functions for time series segments, such as max, min, average or standard deviation.

A typical query example with aggregation is as follows: Let f and g are time series of temperature. “Find the time t when average value of 25 minutes segment f starting from t is 5 degrees higher than that for g.”;

(ii) Regression function

Regression and auto-regression are used to estimate future value from past data. If the deviation between estimated values and actual measurement is large, it is judged to be anomaly.

A typical query example with regression is as follows: “Find the time when the difference between the actual measurement and the estimated value by auto-regression with window size 15 is larger than 5 degrees”.

(iii) Fourier transform

The Fourier transform is known to describe the spectral characteristics for steady-state sensor data. An anomaly is often detected as the change in the frequency spectrum. A typical query example with a Fourier transform is as follows: “Find the time when the spectral density in normal mode is below half that of usual data”.

(2) Time-interval based join

Typical pre-processing steps for raw data are as follows.

- Missing value completion
- Merging data with different sampling periods

We introduced time-interval based join for a time-series data table whose key is a pair of column “time” and column “time interval” to describe the above preprocessing steps, while maintaining the semantics of SQL. We use a method of constructing a subdivision that merges two different subdivisions on the same interval, used in the definition of the

Stieltjes integral, for joining tables with different time intervals.

A standard temporal query language TSQL [7] also supports operation over time intervals, such as intersection and inclusion and so on, but does not support time-interval join.

2.2 Sensor Big Data Management

There are a lot of sensors in a facility, so that a large amount of data will be accumulated as time passes. If there are 10,000 sensors, sampling period is one second and one byte per one point, the amount of data is about 1 Gbytes per one day. If the number of sensors per device is 50, the total number of 10000 sensors would be reched by as little as 200 devices in a facility, or 200 products in the consumer market. Therefore, 10,000 sensors is not an excessive assumption. Furthermore, missing values often occur in sensor data, so relational databases that are frequently used in enterprise domains, may not be suitable, because they have excessively strict data management functions. We propose two functions. One is for data transformation and the other is for fast processing.

(1) Data Transformation

With respect to storage, our system uses key-value store database, which is often used for big data, and provides mutual transformation functions among key-value data, relational data and stream data for developers to select a suitable data type in accordance with the purpose of processing in TPQL scripts. Generally speaking, key-value data are suitable for accumulating data, relational data are suitable for aggregation over relations, and stream data are suitable for data passing to external functions and time series analysis.

The typical stream query language CQL [8] is also a relation-based one, and has a sliding window process as one of its basic operations. CQL provides the transformation between relational data and stream data, that is, “Relation to Stream” and “Stream to Relation”. TPQL adds “Stream to Key-value” and “Key-value to Relation” as basic data transformation functions.

(2) Exemplar Learning

With respect to fast processing, we proposed a compression method that opearates by combining similar data segments into one segment, in order to speed up anomaly detection procedures. We call this method exemplar learning in this paper. Exemplar learning will be illustrated in the next section.

2.3 Feature Extraction Characteristics of Sensor Data

A lot of algorithms have been proposed for anomaly detection [9]. We have a policy to select existing algorithms in accordance with our application requirements. Preprocessing for sensor data is as important as the anomaly detection algorithms themselves. The techniques for preprocessing are sometimes called feature engineering [10], and are a very important factor in determining the success of anomaly detection systems.

Vibrational behavior is very important for anomaly detection. The Fourier transform is a useful and frequently used feature for steady state data, but it is not so useful for transient or non-periodical data from our experience. We introduce a novel feature which we call leg frequency in order to calculate the frequency of variations in time series that include an upward trend and a downward trend alternatively. Leg frequency is treated as a feature function in convolution operation in TPQL.

Rain flow method [11],[12] in material mechanics is a related work. It calculates the amplitude of deformation which causes the fatigues or the cracks form in materials. It calculates the maximal pair of upward trend and downward trend at each maximal point. But our leg vibration analysis calculates the frequency in a time series segment for given amplitude so that it can describe the degree of fluctuation for the given amplitude that is decided to distinguish anomaly from noise with domain knowledge.

A typical query with leg vibration analysis is used in detecting hunting in control systems. A query example is as follows: “Find the time when the number of the alternate occurrences of upward and downward trends whose amplitude are above 3 ℃ during 30 minutes window is larger than 2”. In this example, 3 ℃ and 30 minutes are the parameter that are decided by the application requirement.

Leg vibration analysis will be illustrated in section 4.

3 EXEMPLAR LERNING

3.1 Statistical and Smoothed Trajectory (SST) Features

We proposed statistical and smoothed trajectory (SST) features [5] that can capture the shape and the stochastic behavior of the time series within the window so that it can handle various types of sensor data.

To detect anomalies in a time series, we first learn a model of the time series given normal time series data. To learn a model we use a fixed-size sliding window over the training time series and compute a feature vector representing each window. Our model consists of a set of exemplars representing the variety of feature vectors that exist over all windows in the training time series. The feature vector that is computed for each window consists of a trajectory component that captures the shape of the time series within the window and a statistical component that captures the stochastic component. The trajectory component is designed to capture the low frequency information in the time series window. It is computed using a simple fixed window running average of the raw time series to yield a smoothed time series after subtracting the mean of the window. Because of smoothing, half of the values in the smoothed time series can be discarded without losing important information. Thus, the trajectory component has w/2 elements where w is the number of time steps in the window. Figure 2a) shows a noisy sine wave time series and the corresponding smoothed time series with half the values discarded is shown in Fig. 2b). The statistical component is a small set of statistics computed over time series values in the window which are mainly designed to characterize the high frequency information in the raw time series window. There are many possible choices for a set of statistics that well characterize the high frequency information in a time series window. The statistics used for experiments in this paper are mean, standard deviation (std), mean of the absolute difference $|z(t) - z(t+1)|$ (mean abs diff), number of mean crossings divided by window length (num. mean crossing/w), percentage of positive differences (% pos.diff), percentage of zero differences (% pos.zero), and the average length of a run of positive dif-

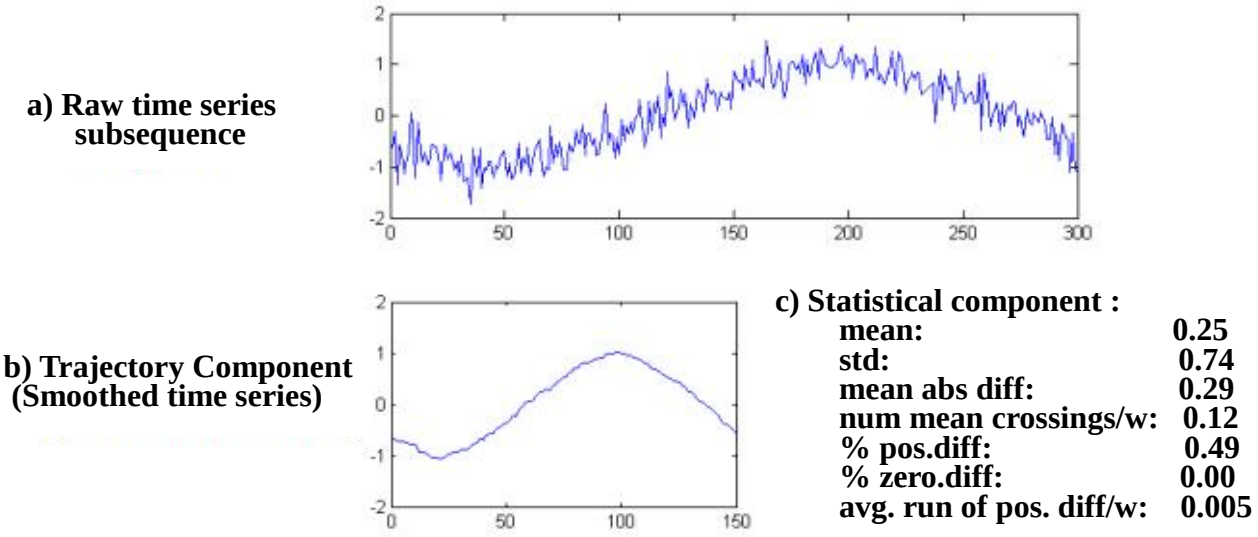


Figure 2: Example time series window (a) along with its trajectory (b) and statistical components (c) Statistical components

ferences divided by window length (avg. run of pos. diff/w). Here, $z(t)$ is the value of the raw time series at time t. Figure 5(c) shows the vector of statistics for an example window. This choice of statistics has worked well in practice across a variety of different time series, but as mentioned before other statistics would likely also work well. The trajectory component is half the length of the window (w/2 time steps), and the statistical component is 7 real numbers for a total of w/2+7 real values. We call this novel representation Statistical and Smoothed Trajectory (SST) features.

3.2 Anomaly Detection using Exemplar Learning

One possible model for a time series is simply the set of all SST features that are computed from all overlapping windows of the time series. This model would be an inefficient representation because the overlapping windows would produce many very similar feature vectors. A much more efficient model is created by finding a small set of exemplars that compactly represent the set of all SST features from the time series. An exemplar in this context is a representation of the SST features of a group of similar windows (overlapping or not) from the training time series. We use an agglomerative clustering algorithm to select SST exemplars from the set of all SST features for a time series.

The agglomerative clustering algorithm works as follows. After computing SST features for every window of the training time series, a set of exemplars is learned by initially assigning each SST feature as its own exemplar and then iteratively combining the two nearest exemplars until the minimum distance between nearest exemplars is above a threshold. This is illustrated in Fig. 3.

We use Euclidean distance to measure the distance between two exemplars:

$$dist(f_1,f_2) = \sum_{i=1}^{\frac{w}{2}} \left(f_1.t(i) - f_2.t(i)\right)^2 + \frac{w}{14}\sum_{i=1}^{7}\left(f_1.s(i) - f_2.s(i)\right)^2$$

where f_1 and f_2 are two feature vectors, $f_j.t$ is the length w/2 trajectory component of f_j, and $f_j.s$ is the length 7 statistical component of f_j. The w/14 coefficient causes the statistical and trajectory components to be weighted equally.

Two exemplars are combined by a weighted average of the corresponding elements. The weight is the count of the number of feature vectors that have already been averaged into each exemplar divided by the total count. Each resulting exemplar is thus simply the overall average of the feature vectors that went into it. The threshold that determines when to stop combining exemplars is set to $\mu + 3\sigma$ where μ is the mean of the Euclidean distances ($dist(f_1,f_2)$) between each initial SST feature vector and its nearest neighbor among the initial SST feature vectors and σ is the sample standard deviation of these distances. The running time of this exemplar selection algorithm is $O(n^2w)$ (where n is the length of the training time series and w is the chosen window size).

After exemplar selection, each exemplar is associated with a set of original SST features that were averaged together to form the exemplar. The standard deviation of each element of the $w/2+7$ length feature vector is then computed and stored with each exemplar. These standard deviations are computed over the set of SST feature vectors associated with a particular exemplar. An exemplar is thus represented by $w/2 + 7$ mean elements and $w/2 + 7$ standard deviation elements. In our experiments, the final exemplar set is typically between 1% and 5% of the total number of features (windows).

After the model is learned, anomalies are found in a testing time series as follows. For each window of the testing time series, an anomaly score is computed. This is done by first computing the SST feature of the window. Then the nearest neighbor exemplar to the SST feature is found. The distance function used is

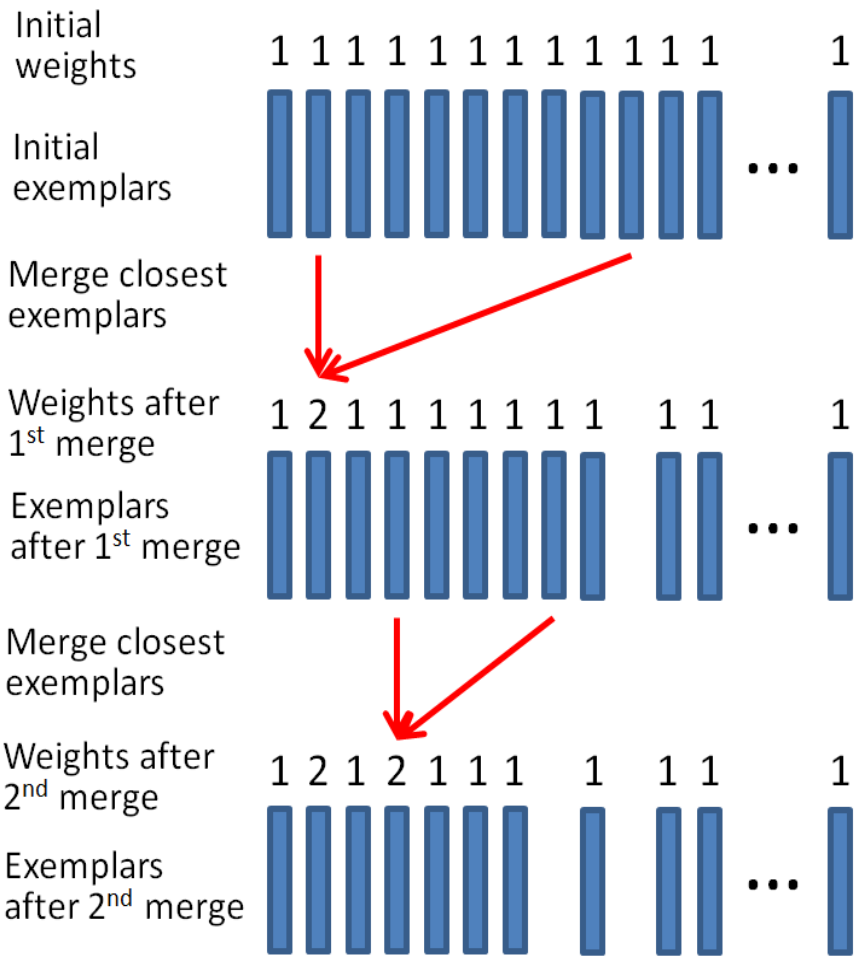


Figure 3: Illustration of agglomerative clustering for learning exemplars. The exemplars (which are SST feature vectors) are represented by blue rectangles. At each iteration the exemplars with minimum distance between them are averaged together using a weighted average. This process is repeated until the minimum distance is above a threshold.

$$d(f,e) = \sum_{i=1}^{\frac{w}{2}} \max\left(0, \frac{|f.t(i) - e.t(i)|}{e.\sigma(i)} - 3\right) + \frac{w}{14} \sum_{i=1}^{7} \max\left(0, \frac{|f.s(i) - e.s(i)|}{e.\varepsilon(i)} - 3\right)$$

where f is the SST feature vector for the current window consisting of a trajectory vector, $f.t$ and a statistical vector $f.s$, e is an exemplar for the current dimension consisting of trajectory $(e.t)$ and statistical $(e.s)$ vectors as well as the corresponding standard deviation vectors, $e.\sigma$ for the trajectory component and $e.\varepsilon$ for the statistical component.

This distance corresponds to assigning 0 distance for each element of the trajectory or statistical component that is less than 3 standard deviations from the mean and otherwise assigning the absolute value of the difference divided by the standard deviation for each element that is more than 3 standard deviations from the mean. In equation 2 and in our experiments, the statistical component is given equal weighting to the trajectory component, although this weighting can be changed based on the application.

4 LEG VIBRATION ANALYSIS

Facility maintenance in buildings, plants, or factories needs to calculate the frequency of variations in sensor data in order to detect a sign of failure or deterioration. Fink et al. proposed a leg search method [13] to find a global trend in a time-series including small variations such as noise. The dotted lines in Fig. 4 are examples of legs. Both lines show the global upward trend that includes local up-down segments.

However, their method treats only single legs so that it can find an upward or downward trend, but can't catch the frequency of variations. We developed leg vibration analysis that can calculate the frequency of variations in time-series that includes upward trends and downward trends that can appear alternately and iteratively. We showed an algorithm whose calculation order is linear in the window size. In contrast, the computational order of a naïve algorithm is factorial in the square of window size.

Definition: time series X, sbsequences X[p:q]

A *Time Series* $X=[x_1,\cdots,x_m]$ is a continuous sequence of real values. The value of the i-th time point is denoted by $X[i] = x_i$.

A Time Series *subsequence* $s = [x_p, x_{p+1},...,x_q] = X[p:q]$ is a continuous subsequence of X starting at position p and ending at position q. We denote the starting time point, the ending time point, and the length of a subsequence l by start, end and length respectively:

$$start(s) \equiv p$$
$$end(s) \equiv q$$
$$length(s) \equiv q\text{-}p\text{+}1$$

Definition: Leg

Let X be time series.

An *upward leg* l = X[p:q] is a subsequence of X that satisfies the following conditions from (1) to (3).

$$\forall i.\ p \le i \le q \quad X[p] < X[i] < X[q] \tag{1}$$
$$X[p-1] \ge X[p] \tag{2}$$
$$X[q] \ge X[q+1] \tag{3}$$

A *downward leg* $l = X[p:q]$ is a subsequence of X that satisfies the following conditions from (4) to (6)

$$\forall i.\ p \le i \le q \quad X[p] > X[i] > X[q] \tag{4}$$
$$X[p-1] \le X[p] \tag{5}$$
$$X[q] \le X[q+1] \tag{6}$$

If l is an upward leg or downward leg, l is called a *leg*.

Definition: Sign and amplitude of a leg

Let X[p:q] be a leg l . We define the *sign* and *amplitude* of a leg l by the following. We denote them by amp and sign respectively:

$$\text{sign}(l) \equiv \text{sign}(X[q] - X[p])$$
$$\text{amp}(l) \equiv \text{abs}(X[q] - X[p])$$

Definition: Leg Vibration Sequence

Let $l_1, l_2,.., l_n$ be legs, and A be a positive real number. A *leg vibration sequence with amplitude A* is a leg sequence $u = [l_1, l_2,.., l_n]$ that satisfies the following conditions from (7) to (9).

$$\text{For } 1 \leqq i \leqq n\text{-}1 \quad \text{end}(l_i) \leqq \text{start}(l_{i+1}) \tag{7}$$
$$\text{For } 1 \leqq i \leqq n \quad \text{amp}(l_i) \geqq A \tag{8}$$
$$\text{For } 1 \leqq i \leqq n\text{-}1 \quad \text{sign}(l_i) \times \text{sign}(l_{i+1}) < 0 \tag{9}$$

Definition: Frequency of a leg vibration sequence

Let $v = [l_1, l_2,.., l_n]$ be a leg vibration sequence. The *start*, *end*, *length*, *sign*, *amplitude* and *frequency* for a leg vibration sequence v are defined as follows. We denote them by start, end, length, amp and freq respectively:

$$\text{start}(v) \equiv \text{start}(l_1)$$
$$\text{end}(v) \equiv \text{end}(l_n)$$
$$\text{length}(v) \equiv n$$
$$\text{sign}(v) \equiv \text{sign}(l_1)$$
$$\text{amp}(v) \equiv \min_i \text{amp}(l_i) \text{ for } 1 \leqq i \leqq n$$
$$\text{freq}(v) \equiv \text{sign}(v) \times \text{length}(v)$$

Definition: Leg vibration sequence set of a subsequence for amplitude A

Let $X[p:q]$ and A be a subsequence of time series X and amplitude respectively. *Leg vibration sequence set for amplitude A* $V(X[p:q], A)$ is defined by a set of leg vibration sequences $v = [l_1, l_2,.., l_n]$ that satisfy the following conditions from (10) to (12).

$$\text{amp}(v) \geqq A \tag{10}$$
$$p \leqq \text{start}(v) \tag{11}$$
$$\text{end}(v) \leqq q \tag{12}$$

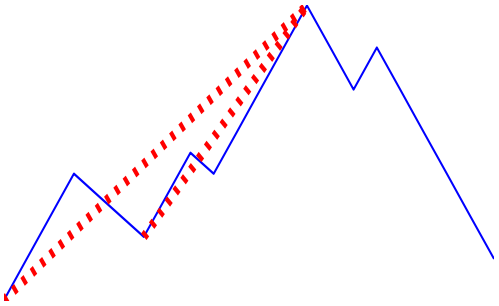


Figure4: Leg

Definition: Leg frequency of a subsequence for amplitude A

Let $V(X[p:q], A)$ be a leg vibration sequence set of a subsequence $X[p:q]$ for an amplitude A. *Leg frequency* $freq_A$ *of a subsequence X[p:q] for A* is defined by below.

$$\text{freq}_A(X[p:q]) \equiv \text{sign}(v_{\max}) \times \text{length}(v_{\max})$$
$$\text{where } v_{\max} = \underset{v \in V(X[p:q],A)}{\text{argmax}}\ \text{length}(v) \tag{13}$$

Leg frequency is well defined because $v_{\max}$ is not unique but the sign of $v_{\max}$s is unique due to the lemma below.

Lemma: Let $V(X[p:q], A)$ be a leg vibration sequence set of a subsequence $X[p:q]$ for amplitude A. The signs of leg vibration sequences that satisfy (13) are the same.

Proof. We assume that $u = [l_1, l_2,.., l_n]$ and $v = [m_1, m_2,.., m_n]$ are leg vibration sequences where both u and v have the maximal length n in $V(X[p:q], A)$ and have different signs. We will show this assumption implies contradiction. Without loss of generality, we can assume that sign(u) is positive and sign(v) is negative; leg l_1 is upward leg and m_1 is downward leg.

Since l_1 and m_1 cross, either one is included by the other, or either one proceeds the other, one of the following conditions is true.

$$\text{start}(l_1) < \text{start}(m_1) < \text{end}(l_1) < \text{end}(m_1) \tag{14}$$
$$\text{start}(m_1) < \text{start}(l_1) < \text{end}(m_1) < \text{end}(l_1) \tag{15}$$
$$\text{start}(l_1) < \text{start}(m_1) < \text{end}(m_1) < \text{end}(l_1) \tag{16}$$
$$\text{start}(m_1) < \text{start}(l_1) < \text{end}(l_1) < \text{end}(m_1) \tag{17}$$
$$\text{start}(l_1) < \text{end}(l_1) \leq \text{start}(m_1) < \text{end}(m_1) \tag{18}$$
$$\text{start}(m_1) < \text{end}(m_1) \leq \text{start}(l_1) < \text{end}(l_1) \tag{19}$$

The definition of leg implies that the above magnitude relations in the formulas from (14) to (17) satisfy not equality but inequality.

First, we will deduce the contradiction when the condition (14) is true. Since l_1 is an upward leg, X[start(m_1)] < X[end(l_1)]. Since m_1 is an downward leg, X[start(m_1)] > X[end(l_1)]. These equations contradict each other. When the condition (15) is true, a proof is in the same way.

Secondary, we will deduce the contradiction when the condition (16) is true. Since amp(m_1) $\geqq A$ and l_1 an upward leg, l_{1_1} = X[start(l_1): start(m_1)] is an upward leg whose amplitude is greater than or equal to A. Therefore, [l_{1_1}, m_1, m_2,.., m_n] is a leg sequence whose length is $n + 1$. It contradicts that v has the maximal length n in $V(X[p:q], A)$. When the condition (17) is true, a proof is in the same way.

Lastly, we will deduce the contradiction when the condition (18) is true. Since [l_1, m_1, m_2,.., m_n] is a leg sequence whose length is $n + 1$. It contradicts that v has the maximal

length n in $V(X[p:q], A)$. When the condition (19) is true, a proof is in the same way.

Therefore, the initial assumption – u and v have different sign – must be false. □

Definition: Leg Frequency $freq_{X,A,W}(t)$

Let X, A, W are time series, amplitude and window size respectively. A *Leg Frequency of X at time t with A and W*, $freq_{X,A,W}(t)$, is defined as follows:

$$freq_{X,A,W}(t) \equiv freq_A(X[t:t+W-1])$$

Leg frequency qualifies the vibration of a time series subsequence, when amplitude and window size are given. Larger amplitude means larger width of variation. The sign of frequency is the sign of the first leg of the longest leg sequence. That is, a positive frequency at time point t with window size W means that the first leg has an upward trend in the subsequence X[t:t+w-1]. Similarly, a negative frequency means that the first leg is downward. A frequency of 2 means that the subsequence has one convex pattern whose amplitude is larger than A, while a frequency of -2 means that it has one concave pattern. If the absolute value of the frequency for a subsequence is larger than 4, we know that the subsequence has at least 4 consecutive up-down trends with the specified amplitude within the specified window size. This rule is often used for detecting vibration over specified amplitude.

Figure 5 shows the leg sequences in the same subsequence for several different amplitudes. The amplitudes are 1 for the top, 2 for the middle, and 4 for the bottom, respectively. The frequencies are 6 for the top, - 5 for the middle and 2 for the bottom respectively.

Let n, W and A be the length of time series, window size and amplitude respectively. The computational order of calculating leg frequency $freq_{X,A,W}$ by an algorithm derived directly from the definition is O($n \times (W!)$), because searching all the possible legs in constructing $V(X[t:t+W-1], A)$ needs the computation whose order is O((W!)) in the worst case. However, we can obtain faster algorithm whose computational order is O($n \times W$) by using left most vibration sequence and the theorem which it satisfies.

Definition: Leftmost Leg Vibration Sequence

Let X and L be a time series and a set of legs whose amplitude are greater than or equal to amplitude A respectively. The *leftmost vibration sequence* in $V(X[p:q], A)$ is a leg sequence $[m_1,.., m_i,\ldots, m_n]$ when m_i is selected repeatedly until we can not select it in the below way:

$$m_1 = \text{argmin}_{l \in L}\ \text{end}(l)$$
$$m_{i+1} = \text{argmin}_{l \in L_i}\ \text{end}(l) \text{ for } \geq 1$$
$$\text{Where } L_i \equiv \{l \in L | \text{start}(l) \geq \text{end}(m_i) \text{and } \text{sign}(l) \cdot \text{sign}(m_i) < 0\}$$

Theorem: The leftmost leg vibration sequence is a leg sequence that has the maximal length in $V(X[p:q], A)$.

Proof. Let u = [l_1, l_2,.., l_n] be the leftmost leg vibration sequence whose length is n. We will reduce contradictory if v = [m_1, m_2,.., m_k] is assumed a maximal leg vibration sequence in $V(X[p:q], A)$ whose length k is greater than n.

First, we will prove that leg l_1 and m_1 have the same sign. Let us suppose that leg l_1 and m_1 have different sign. Since u is leftmost, we can prove [l_1, m_1, m_2,.., m_k] is a leg vibration sequence whose length is m + 1 by exploiting the reasoning

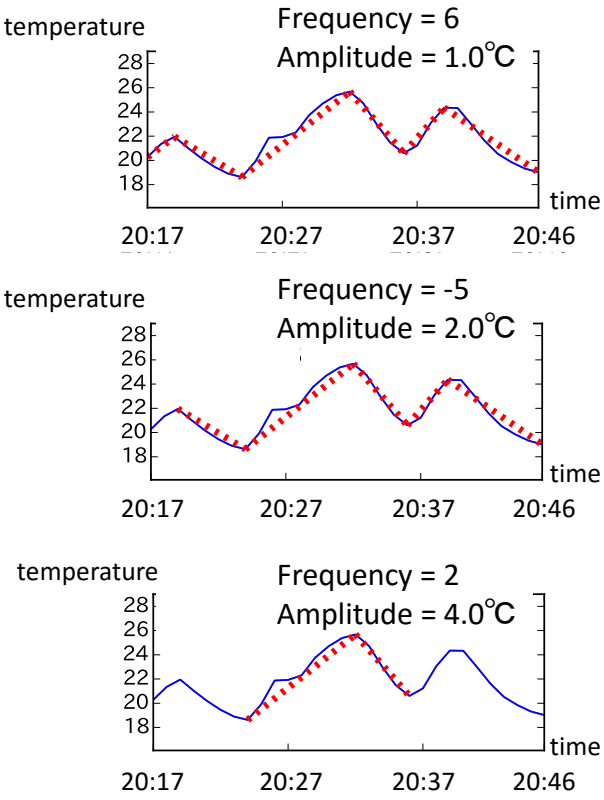


Figure 5: Trend Graphs of Experimental Data

method in the previous lemma. It contradicts the length of v is maximal. Therefore, the assumption – l_1 and m_1 have different sign – must be false.

Since l_1 and m_1 have the same sign and u is leftmost, it is true that end(l_1) ≤ end(m_1) ≤ start(m_2). Therefore [l_1, m_2,.., m_k] is a leg vibration sequence. In the same way, [l_1, l_2, m_3,.., m_k] is a leg vibration sequence. Since $k > n$, [l_1, l_2,.., l_n, m_{n+1},.., m_k] is a leg vibration sequence where the amplitude of each leg is greater than or equal to A. Therefore, a subsequence X[end(l_n) : end(m_k)] must include the leftmost leg l_{n+1} whose amplitude is greater than or equal to A and whose sign is the same as that of m_{n+1}, that is, different from that of l_n. This means that [l_1, l_2,.., l_n, l_{n+1}] is the leftmost vibration sequence. It contradict that u is the leftmost vibration sequence.

Therefore, the initial assumption – the length of v is greater than that of u – must be false. □

The computational order of constructing leftmost leg vibration sequence is O(W), therefore that of leg frequency $freq_{X,A,W}$ is O($n \times W$).

5 EVALUATION

We have developed a TPQL based anomaly detection system and applied it to a practical application system. With regard to development efficiency, we compared the number of lines of programs by TPQL to those by Java. We confirmed that the convolution operation for describing feature extraction for sliding windows can reduce about 100 lines to 1 line. However, enough evaluation of development efficiency for constructing real applications remains as a future

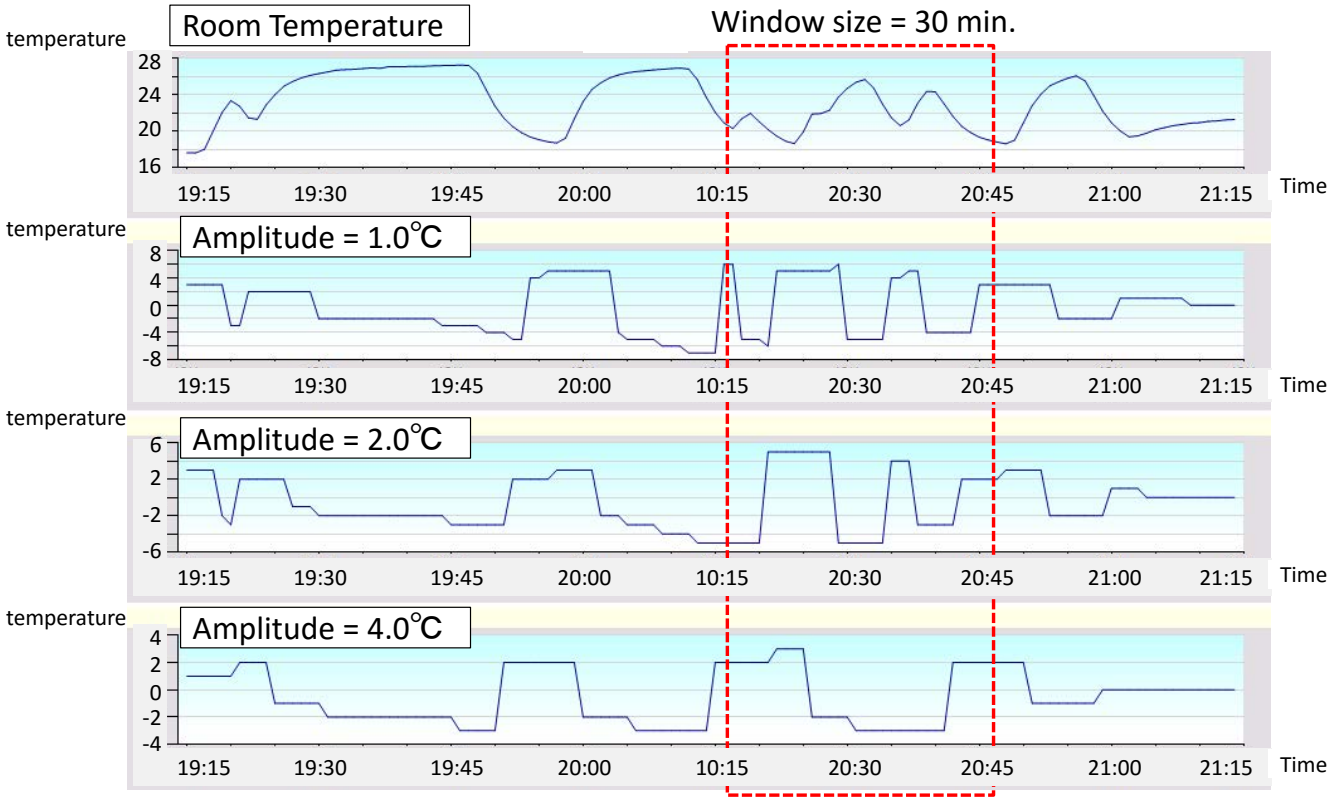


Figure 6: Trend Graphs of Experimental Data

work. The rest of this section shows the result of processing time.

(1) TPQL

We confirmed that our system satisfies the processing time that are required by our real application. Our requirement of ECM for buildings is that the processing must be completed within one day for the following conditions:

- 3 anomaly detection scenarios for each signal
- 5,000 signals with sampling period one minute in half a year

The detail results and the conditions in the experiment are described in [4].

(2) Exemplar Learning

We compared our algorithm to the simple yet effective Brute Force Euclidian Distance (BFED) algorithm [14] which has proven to be the most accurate over a variety of different testing time series [15]. Data for the experiment are 24 data sets that are available from the paper [16] and 2 synthesized data sets. The result shows that our algorithm is about from 5 times to 100 times faster than the BFED algorithm without losing accuracy. The detail results and the conditions in the experiment are described in [5].

(3) Leg vibration analysis

We applied leg vibration analysis to anomaly detection for HVAC (Heating Ventilation, and Air Conditioning) systems. It is known that a significant variation of room temperature during operation often shows an anomaly in the control and/or sensor system.

The data for the experiment consisted of room temperature readings with sampling period one minute, for a total duration of three years. The total number of time points is thus 1,578,239. We obtain leg frequencies with window size 30 minutes and with amplitude 1.0℃, 2.0℃, and 4.0℃, respectively. A higher amplitude means a higher warning level. Leg vibration analysis can enable adaptive monitoring by selecting an appropriate window size and amplitude.

Our leg vibration analysis software detected 1901 points (0.12%), 454points (0.029%), and 69 points (0.0044%) for amplitudes of 1.0℃, 2.0℃, and 4.0℃, respectively. Fig. 4 shows a snapshot of leg frequencies as a function of time for each amplitude. The top, the middle, and the bottom graphs correspond to amplitudes of 1.0℃, 2.0℃, and 4.0℃, respectively. Fig. 5 above shows the leg sequences for the subsequences that are surrounded by the rectangular area in Fig. 6.

The processing times for amplitudes 1.0℃, 2.0℃, and 4.0℃ were 0.612 sec., 0.554sec., and 0.489 sec. respectively. We set the threshold on the absolute value of leg frequency to 4, in order to detect anomalies for each temperature. All computational times satisfy the requirements of our application. The detail results and the conditions in the experiment are described in [6].

6 CONCLUSIONS

This paper describes a TPQL-based anomaly detection system whose main features are convolution operation, time interval join, data transformation, exemplar learning and leg vibration analysis.

Our efforts have been mainly focused on feature extraction and machine learning based anomaly detection, that is, a data-driven approach. A data-driven approach has a problem that it can detect the differences from the ordinary behavior of sensor data, but it cannot distinguish the symptoms of failures from only unusual behavior. Our future direction is to develop a hybrid method to combine a data-driven ap-

proach with a physical model-based approach with equipment domain knowledge in order to explain whether anomalous behavior is actually a symptom of a failure.

REFERENCES

[1] J. Zheng, D. Simplot-Ryl, C. Bisdikian, H.T.Mouftah: "The Internet of Things [Guest Editorial]," Communications Magazine, IEEE , Vol.49, No.11, pp.30-31 (2011).

[2] M. Imamura, D. Nikovski, Z. Sahinoglu, M. Jones: "A Survey on Machine Learning for Equipment Condition Monitoring Using Sensor Big Data," IIEEJ Transactions on Image Electronics and Visual Computing Vol.2 No.2, pp. 112-121 (2014).

[3] M. Imamura, S. Takayama, and T. Munaka: "A stream query language TPQL for anomaly detection in facility management," The 16th International Database Engineering & Applications Symposium (IDEAS '12), pp. 235-238 (2012).

[4] M. Imamura,T. Takeuchi, S. Kitagami, M. Kanno, T. Munaka: "Time Series Data Query Language TPQL for anomaly detection in facility," Journal C of Electronics and Communications in Japan,Vol.134, No.1, pp. 156-167 (2014). (in Japanese).

[5] M. Jones and D. Nikovski and M. Imamura and T. Hirata: "Exemplar Learning for Extremely Efficient Anomaly Detection in Real Valued Time Series," Data Mining and Knowledge Discovery (DAMI), First online: 25, pp 1-28 (2016).

[6] M. Imamura, T. Nakamura, H. Shibata, N. Hirai, S. Kitagami, T. Munaka: "Leg Vibration Analysis for Time Series," IPSJ Journal, Vol. 57, No.4, pp.1303-1318 (2016). (in Japanese).

[7] R. T. Snodgrass (Ed.): The TSQL2 Temporal Query Language. Kluwer (1995).

[8] A. Arasu, S. Babu, J. Widom: "The CQL continuous query language: semantic foundations and query execution," The International Journal on Very Large Data Bases archive, Vol. 15, No. 2, (2006).

[9] V. Chandola, A. Banerjee, V. Kumar: "Anomaly detection: a survey," ACM Comput Survey, Vol. 41, No. 3 (2009).

[10] W. Yan: "Feature Engineering for PHM applications," The 7th Annual Conference of the Prognostics and Health Management Society, http://www.phmsociety.org/sites/phmsociety.org/files/FeatureEngineeringTutorial_2015PHM_V2.pdf (2015).

[11] T. Endo, M. Matsuishi, K. Kounaga, K. Kobayashi, K. Takahashi: "Rain Flow Method and Its Application," Research Report of Kyushu Institute of Technology, http://www-it.jwes.or.jp/qa/details.jsp?pg_no=0040020170 (1974).

[12] I. Rychlik: "A new definition of the rainflow cycle counting method," International journal of fatigue Vol. 9. No. 2, pp. 119-121 (1987).

[13] E. Fin, B. P. Kevin: "Indexing of Compressed Time series," DATA MINING IN TIME SERIES DATABASES, World Scientific, pp. 43-65 (2004).

[14] T. RakthanmanonT, B. Campana B, A. Mueen, G. Batista, B. Westover, Q. Zhu, J. Zakaria, E. Keogh Searching and mining trillions of time series subsequences under dynamic time warping. 18th ACM SIGKDD international conference on knowledge discovery and data mining, pp 262–270 (2012).

[15] H. Ding, G. Trajcevski, P. Scheuermann, X. Wang, and E. Keogh: "Querying and Mining of Time Series Data: Experimental Comparison of Representations and Distance Measures," VLDB 2008, pp.1542-1552 (2008).

[16] E. Keogh, J. Lin, A. Fu: "HOT SAX: finding the most unusual time series subsequence: algorithms and applications," The Fifth IEEE international conference on data mining, pp. 226–233, www.cs.ucr.edu/eamonn/discords/ (2005).

(Received September,30,2015)
(Revised June 10,2016)

Makoto Imamura He received a M.E. degree from Kyoto University of Applied Mathematics and Physics in 1986 and a Ph.D. degree from Osaka University of the Information Science and Technology in 2008. He is a professor of the school of Information and Telecommunication Engineering at Tokai University. He had been worked as a research staff at the Information Technology R&D Center in Mitsubishi Electric Corporation until 2015. He has worked on datamining methods for prognostics and health management and cyber-physical production system.

Daniel Nikovski He received a PhD in robotics from Carnegie Mellon University in 2002, and is presently a senior member of research staff and group manager of the Data Analytics group at Mitsubishi Electric Research Laboratories. He has worked on probabilistic methods for reasoning, learning, planning, and scheduling, and their applications to hard industrial problems. He is a member of IEEE.

Michael Jones He received a Ph.D. degree from the Electrical Engineering and Computer Science department of the Massachusetts Institute of Technology (MIT) in 1997, and is currently a senior principal member of the research staff at Mitsubishi Electric Research Laboratories. He has worked mainly on problems in computer vision, but recently has also focused on time series analysis.

Submission Guidance

About IJIS (International Journal of Informatics Society)

International Journal of Informatics Society (ISSN 1883-4566) is published in one volume of three issues a year. One should be a member of Informatics Society for the submission of the article at least. A submission article is reviewed at least two reviewer. The online version of the journal is available at the following site: http://www.infsoc.org.

Aims and Scope of Informatics Society

The evolution of informatics heralds a new information society. It provides more convenience to our life. Informatics and technologies have been integrated by various fields. For example, mathematics, linguistics, logics, engineering, and new fields will join it. Especially, we are continuing to maintain an awareness of informatics and communication convergence. Informatics Society is the organization that tries to develop informatics and technologies with this convergence. International Journal of Informatics Society (IJIS) is the journal of Informatics Society.

Areas of interest include, but are not limited to:

Computer supported cooperative work and groupware
Intelligent transport system
Distributed Computing
Multi-media communication
Information systems
Mobile computing
Ubiquitous computing

Instruction to Authors

For detailed instructions please refer to the Authors Corner on our Web site, http://www.infsoc.org/.

Submission of manuscripts: There is no limitation of page count as full papers, each of which will be subject to a full review process. An electronic, PDF-based submission of papers is mandatory. Download and use the LaTeX2e or Microsoft Word sample IJIS formats.

http://www.infsoc.org/IJIS-Format.pdf

LaTeX2e

LaTeX2e files (ZIP) http://www.infsoc.org/template_IJIS.zip

Microsoft Word™

Sample document http://www.infsoc.org/sample_IJIS.doc

Please send the PDF file of your paper to secretariat@infsoc.org with the following information:

Title, Author: Name (Affiliation), Name (Affiliation), Corresponding Author. Address, Tel, Fax, E-mail:

Copyright

Publisher

Address: Informatics Laboratory, 3-41 Tsujimachi, Kitaku, Nagoya 462-0032, Japan
E-mail: secretariat@infsoc.org

CONTENTS