International Journal of Informatics Society

06/23 Vol.15 No.1 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 quality and value as a resource. Informatics also referred to as Information science, studies the structure, algorithms, behavior, and interactions of natural and artificial 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 the study of informatics 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

Toshihiro Wakita

Guest Editor of the Forty-third Issue of the International Journal of Informatics Society

We are delighted to have the Fortieth issue of the International Journal of Informatics Society (IJIS) published. This issue includes selected papers from the Sixteenth International Workshop on Informatics (IWIN2022), which was held online, August 31-September 3, 2022. The workshop was the sixteenth 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, 25 papers were presented in eight 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 to IWIN2022 was reviewed in terms of technical content, scientific rigor, novelty, originality, and quality of presentation by at least two reviewers. Through those reviews, 18 papers were selected for publication candidates of IJIS Journal, and they were further reviewed as a Journal paper. We have three categories of IJIS papers, Regular papers, Industrial papers, and Invited papers, each of which was reviewed from different points of view. This volume includes papers among those 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.

Toshihiro Wakita is a Professor of Department of Vehicle System Engineering at Kanagawa Institute of Technology since 2018. He joined Toyota Central R&D Labs., INC. in 1985. He received his Ph.D. degree in Information Sciences from Nagoya University in 2006. He has been engaged in research and development of human machine interface of vehicle and Advanced driver-assistance systems. His current interests are interaction between autonomous mobility and human. He is a member of IPSJ, IEICE, IEEE and JSAE (Japanese Society of Automotive Engineering).

Regular Paper

Target Area Extraction for Object Recognition from Gesture Trajectory Detected Using YOLO

Tsukasa Kudo[†]

[†]Faculty of Informatics, Shizuoka Institute of Science and Technology, Japan
kudo.tsukasa@sist.ac.jp

Abstract -
Advances in mobile cameras such as smart glasses and object recognition technologies have made it possible to extract and utilize various types of information seen while working or walking. And, when the object is small in the camera's view, it is necessary firstly to perform object detection to extract the object area. Recently, object detection methods using deep learning have been actively studied, and their applications have spread to various fields. However, using deep learning requires preparing training data for each object, which is challenging when various objects are targeted. In this study, instead of these deep learning–based object detection methods, a method of extracting the target area from the trajectory of a gesture detected using You Only Look Once (YOLO) is proposed. An advantage of this method is that the YOLO model can be trained using the training data of only a specific body part, such as a fingertip. Furthermore, the experiments show that, contrary to the previously mentioned object detection methods, even the target area and video frame without special features can be extracted according to the purpose such as extracting an arbitrary area of an image.

Keywords: YOLO, gesture detection, object recognition, transfer learning, object detection

1 INTRODUCTION

Recently, object recognition for videos and images has been actively studied. Its applications are expanding into various fields, such as automatic car driving and immigration control using face recognition. And with the development of the Internet of Things (IoT), especially edge computing that performs computations as close to data sources as possible, its application field is expanding to mobile cameras such as smart glasses [15]. During object recognition, when the target in the image is small, the target area must first be extracted using object detection. For example, face detection using the Haar-like feature is performed during face recognition to extract the face area as a bounding box, and then face recognition is performed on this face area [20].

With the advancements in deep learning, various methods for object detection have been actively studied [12]. For example, You Only Look Once (YOLO) detects the target as a bounding box and simultaneously recognizes the target [17]. Furthermore, videos can be processed in real-time by YOLO [21]. However, these deep learning–based methods require training data for each target, so it is difficult to apply them to fields where various objects are targeted.

As such a field, the author has been working on the automation of inventory management in machine factories. The aim of this research is to automatically determine the target parts and their inventory number from the videos of smart glasses worn by workers. However, since there are thousands of different parts in such factories, the above-mentioned challenge of preparing the training data arose [4].

In this paper, a method of extracting the target area from the trajectory of a gesture detected using YOLO is proposed. The gesture is performed so that, for example, the target area is a closed area surrounded by its trajectory. Hence, the target area extracted from this trajectory can be the same as the object detection result. Notably, since gestures can be performed by a specific body part, such as a fingertip, this method requires only one type of training data for various object detections.

The detected trajectory usually includes erroneously detected points (subsequently called noises) and extra gesture sections. Therefore, in this method, the noises are removed using the median of nearby points, and the target section or feature point is indicated by the stop motion of the gesture. Furthermore, the experiments on actual use cases show that the arbitrary area of an image and frame of a video can be extracted. Consequently, object recognition can be done using simple methods, such as optical character recognition (OCR) and template matching.

The remainder of this paper is organized as follows. Section 2 presents related work and the aims of this study, and Sec. 3 proposes a method for extracting the target area from a gesture trajectory. Section 4 shows the implementation of the proposed method and the extraction of the target area through experiments. Section 5 shows the applications and evaluations of this method, and Sec. 6 discusses the evaluation results. Finally, Sec. 6 concludes this paper.

2 RELATED WORKS AND AIM OF THIS STUDY

In a machine factory, parts are stored in bulk containers as shown in Fig. 1 (3). However, as can be seen from Fig. 1 (3), the number of parts cannot be visually counted, which has been a problem in inventory management. For this problem, the authors have shown that deep learning can be used to estimate the number from images of containers with practical accuracy and that computer graphics (CG) can be used to efficiently prepare training data [3], [7].

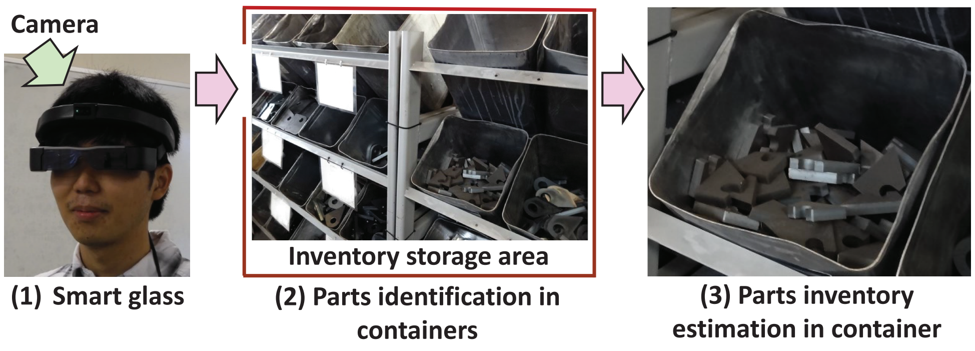

Camera
Inventory storage area
(1) Smart glass
(2) Parts identification in containers
(3) Parts inventory estimation in container

Figure 1: Inventory management of parts with smart glass

However, in such factories, there are usually several thousand types of parts stored dispersedly in the inventory storage areas in the factory. To manage such widespread objects, methods combining smart glasses and object recognition such as deep learning have been proposed and systems have been studied to assist factory workers and the visually impaired [14], [22].

So, I focused on the fact that the parts inventory fluctuates due to replenishment and picking (i.e., taking out the parts to be used) by workers, and attempted to automatically extract inventory information from videos continuously captured by smart glasses worn by workers shown in Fig. 1 (1). Concretely, object recognition with deep learning was used to identify the worker's entry into the inventory storage area, the target container, and parts from the videos, as shown in Fig. 1 (2). However, since the parts are small in the camera's view, the accuracy of the discrimination degraded [4]. In such a case, the target area must be extracted using object detection prior to object recognition.

Various methods for joint object detection and recognition have been proposed. Faster R-CNN performed them in a lump through collective end-to-end training of both models [18], and YOLO executed them with a single neural network to improve efficiency [17]. Regarding differently scaled objects, SSD could process them collectively [11]. RetinaNet improved the efficiency by introducing the Feature Pyramid Network (FPN) and improving the loss function [9], [10]. Subsequently, M2Det further improved accuracy and efficiency by introducing the new FPN and loss function [23].

Among these methods, YOLO has been improved repeatedly through version upgrades, with several models currently available [2]. YOLO has a high detection efficiency and accuracy and can be applied to real-time object detection and recognition [21].

However, since the above methods use deep learning, training data (i.e., ground truth) comprising images and the correct labels for model training must be prepared. For example, YOLO requires ground truth comprising the images and their corresponding labels of bounding box positions and object classification for each target. Here, object classification indicates the object type to be recognized. Therefore, this creates a significant burden in preparing training data when various objects are to be detected.

In the above-mentioned machine factory, since there are

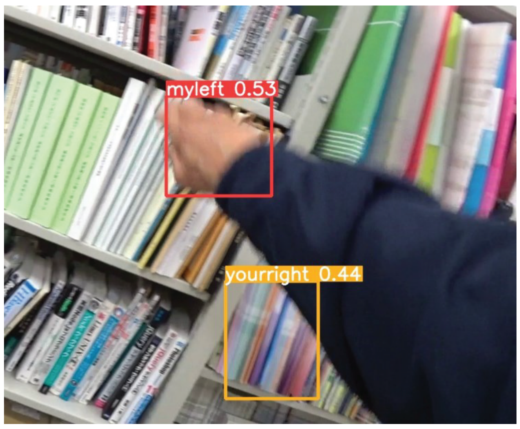


Figure 2: Object detection result using YOLO

thousands of types of parts, it is not practical to apply object detection using deep learning. So, I focused on the fact that the operator picks up the parts when working, and tried a method of using optical flow for parts detection. However, it was found that the detection accuracy degraded when the background was complex [5]. In addition, using Cycle-GAN, which is a kind of generative adversarial network, I showed that the contour of a container can be detected [6]. However, this method did not cover the detection of parts in containers.

The motivation for this study is the idea that the target area of an arbitrary object can be extracted by detecting the trajectory of a specific object, such as the tip of a hand indicated by a gesture in a video. Using this idea, I expected that randomly stored parts in containers shown in Fig. 1 (3) can be detected.

By the way, trained models and training data for YOLO for various objects, such as the coco dataset, are available on the Internet [1], [2], [8]. Therefore, YOLO can be easily used for real-time object detection and recognition from videos when targeting specific objects. That is, this trajectory detection can be performed in real-time by YOLO targeting only one object, and training the model is easy.

Furthermore, several applications have been proposed for hand gesture recognition using deep learning, such as conversation and device control [13], [16], [19]. However, application studies on extracting the target area, such as object detection, could not be obtained.

Figure 2 shows the resulting image of object detection for the right hand by YOLO from a video frame. The detected objects are indicated by bounding boxes, and the target type and estimation accuracy of the object is indicated above the upper side of each box. For the detected boxes, my left hand (myleft) was detected with an accuracy of 53% (0.53) on the upper side. However, there is falsely detected noise on its lower side, and its accuracy is 44%.

The aims of this study are stated below. The first is to develop a method that accurately extracts the target area from the trajectory of gestures detected using YOLO. To achieve this, the noise shown in Fig. 2 and the extra gesture section should be removed to improve the accuracy of the detected trajectory. The second is to demonstrate the effectiveness of this method. For example, this method requires an extra action (a gesture) compared to object detection using YOLO directly. On the other hand, in this method, only a single body part is required for model training, regardless of the number of target object types; this method has the advantage of extracting arbitrary areas for the purposes, such as areas without special features. Hence, to clarify what application fields are effective based on actual business use cases.

3 TARGET AREA EXTRACTION METHOD FROM GESTURE TRAJECTORY

Two steps are used to extract the target area using gesture trajectory: (1) accurate trajectory detection, for which noises and extra gesture sections are removed, and (2) target area extraction from the detected gesture trajectory.

3.1 Gesture Trajectory Detection Method

This method uses the trajectory of a gesture by a specific body part. Moreover, it indicates the target gesture section with stop motions. For example, to indicate the area of some books shown in Fig. 2, the gesture is paused when the hand moves to the target book; subsequently, it is moved along the outline of the books. Finally, it is paused at the end of the outline and moved away. The gesture section between these two pauses is used to extract the area of the book. In the following description, the upper left corner of the bounding box (Fig. 2) is used as the gesture indication for the detected point.

The following procedure detects gesture trajectory: (1) valid points are selected from the detected points using YOLO, (2) noise is removed using the medians of nearby points, and (3) the target section is extracted as described above.

3.1.1 Valid Point Selection

In this method, there is a maximum of one valid point in each frame. To select this valid point from the detected points, the following condition is adopted. Its accuracy is the highest in the frame and greater than the threshold (i.e., the specified value). If each pair of the detected point coordinates and accuracies of the i-th frame is indicated by c_{ij} and a_{ij}, the valid

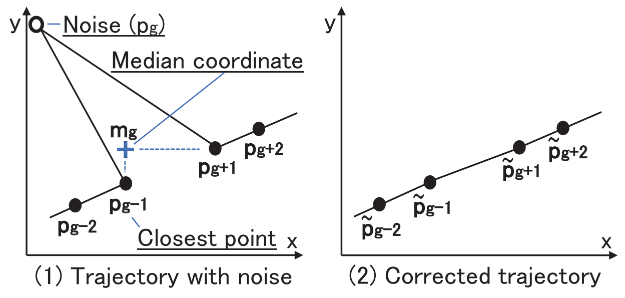


Figure 3: Noise elimination using median coordinates

point with a coordinate s_i in Eq. (1) is selected.

$$s_i = \begin{cases} c_{ik} & (\exists k(a_{ik} \geq V \wedge a_{ik} = max(\{a_{ij}\})) \\ \emptyset & (\forall k(a_{ik} < V)) \end{cases} \tag{1}$$

Here, $\emptyset$ indicates that the frame does not have a valid point; V indicates the threshold; $\{a_{ij}\}$ indicates the set of a_{ij}.

In Fig. 2, if $V = 0.4$, then the upper-left coordinate of the upper bounding box is selected because its accuracy is the highest in the two boxes and greater than the threshold. However, if $V = 0.6$, this frame is ignored.

3.1.2 Noise Elimination Using Median Coordinates

For each frame, the median coordinates are calculated with the valid points of the nearby frames to eliminate noises. Let $p_j(j = 1, 2, 3, \cdots)$ be the ordered set of coordinates with eliminating s_i if $s_i = \emptyset$ from $\{s_i\}$, namely the set of s_i in Eq. (1). Figure 3 (1) shows an example where p_g is a noise.

The median of p_g is constructed using the section before and after the index g. Let R_g indicate the set of indices of this section, and let p_{Rx} and p_{Ry} indicate the set of x-coordinates and y-coordinates, respectively. The median of this section is defined as follows:

$$m_g = (median(p_{Rx}), median(p_{Rx}))$$

Here, *median* is the function to get the median value of the coordinates. In Fig. 3, the coordinate m_g with the median of each of the x–y coordinates is selected.

The noises are eliminated using these median coordinates. The median coordinate $\tilde{m}_g$ of the frame g is defined as follows:

$$\tilde{m}_g = \{p_h | \exists h(dist(p_h, m_g) = min(dist(p_n, m_g)) \\ \wedge \forall n \in R_g)\} \tag{2}$$

$dist(p_n, m_g)$ indicates the distance between p_n and m_g. Conversely, $\tilde{m}_g$ denotes the coordinate $p_r(\exists r \in R_g)$ of the closest point to the median coordinate m_g. Hence, the median trajectory is defined as the set of points $\tilde{p}_g$ shown in Eq. (3).

$$\tilde{p}_g = \begin{cases} p_g & (\tilde{m}_g = p_g) \\ \emptyset & (\tilde{m}_g \neq p_g) \end{cases} \tag{3}$$

That is, if p_g is not the closest coordinate to m_g in R_g, then it is converted to $\emptyset$, that is, it is eliminated; otherwise, p_g is

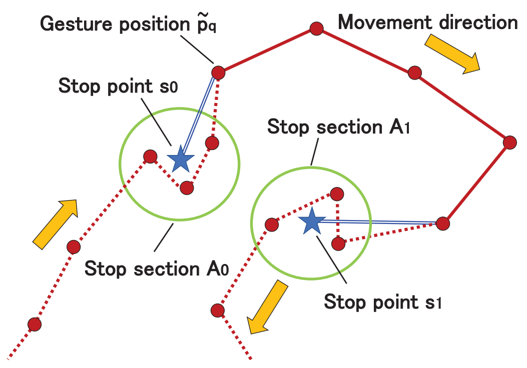


Figure 4: Target gesture section extraction way

adopted for $\tilde{p}_g$. Thus, noises are eliminated from the median trajectory.

In Fig. 3, the closest coordinate to the median $\tilde{m}_g$ is p_{g-1}, so the coordinate $\tilde{p}_g$ is set to $\emptyset$ and eliminated. The other coordinates are set as $\tilde{p}_n = p_n(r \in R_g)$. Consequently, a trajectory is corrected using the median coordinate, as shown in Fig. 3 (2).

3.1.3 Extraction of Target Section Using Stop Points

Let $q(q = 1, 2, 3, \cdots)$ be the coordinates of the q-th point in the set of $\tilde{p}_g$ denoted in Eq. (3), namely points other than noises. To extract the target section of a gesture, two stop motions are detected. The stop section is defined as the section where the distance between $\tilde{p}_q$ and $\tilde{p}_{q+1}$ is below or equal to a specified threshold H. This is the section in which the gesture is almost stopped and is shown as A_0 and A_1 in Fig. 4.

k-th stop section is expressed as follows:

$$A_k = \{\tilde{p}_q | (dist(\tilde{p}_{q+1} - \tilde{p}_q) < H) \wedge (\tilde{p}_q \notin A_l), l < k\} \quad (4)$$

Here, $dist$ is the same as in Eq. (2). Note that the threshold H is used because actual fingertip gestures cause slight fluctuations, even when the fingertip is stopped.

For the stop section A_k, stop point s_k is defined as the point whose coordinates are the simple average of the coordinates of all points in A_k. Figure 4 shows that the point $\tilde{p}_q \in A_k$ in the stop section is replaced with the stop point s_k, and the points before and after the stop section indicated by the dashed lines are eliminated.

Consequently, the target section of the gesture trajectory (subsequently called the detected trajectory) is extracted. This is indicated by the solid and double lines in Fig. 4.

3.2 Target Area Extraction Method Using Gesture Trajectories

In this section, the target area extraction method from the detected trajectories is shown. First, the basic target area extraction method, which extracts a specific area from the image same as object detection, is shown. Second, the following application methods are shown. The first is extracting frames for which gestures cannot be used, such as close-up photography; the second is extracting as a bounding box, which can also be used for tilt correction.

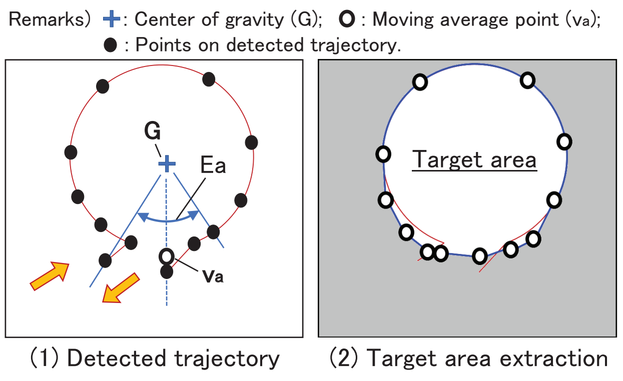

Remarks) +: Center of gravity (G); O: Moving average point (va);
●: Points on detected trajectory.
G
Ea
Va
Target area
(1) Detected trajectory
(2) Target area extraction

Figure 5: Target area extraction method using moving averages

3.2.1 Basic Target Area Extraction Method

The detected trajectory is accompanied by fluctuations of gesture motion. To compensate for these fluctuations, a moving average of the coordinates along the angle from the center of gravity is created. Similarly, doubles and omissions near the start and end of the detected trajectory are also compensated.

The x–y coordinate of the center of gravity G is obtained as an average of the x-coordinate and y-coordinate of the points on the detected trajectory, as shown in Fig. 5 (1). Subsequently, the coordinate of each point is transformed into a pair $\hat{p}_a = (\theta_a, r_a)$ of angle and distance from G. Let E_a be the angle section to calculate the moving average, and define the moving average v_a by Eq. (5).

$$v_a = (\theta_a, \bar{r}_a) \qquad (\theta_a \in E_a) \qquad (5)$$

Here,

$$\bar{r}_a = (\sum r_a)/n$$

n is the number of coordinates contained in E_a, and it is five in Fig. 5 (1).

Consequently, $\bar{r}_a$ is the moving average of the distances between G and points on the detected trajectory along the angle. By connecting the coordinates of this moving average set $\{v_a\}$ along the angle, the target area is extracted, as shown in Fig. 5 (2).

3.2.2 Close-up Target Frame Extraction Method

Since the labels, such as those attached to fixed assets and products, are small, an enlarged image must be obtained by close-up photography in order to use OCR. In this case, the target area should not be specified. However, the close distance from the camera makes it difficult to indicate the target frame with the gesture. To solve this problem, a target frame extraction method that combines camera movement and gestures is proposed.

Figure 6 shows that the camera is held still to view target (1) and zoom-in (a) for a close-up of the label (2). Subsequently, zoom-out (b) is performed to indicate the target frame by gestures (3). Consequently, the target frame where the label was taken was between this zoom-in and zoom-out. This operation can be performed using the mobile camera by approaching and leaving.

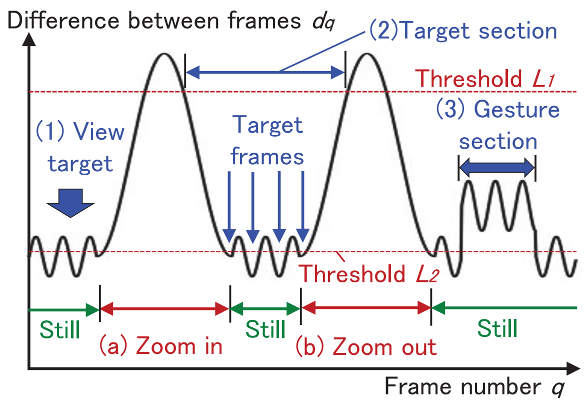


Figure 6: Target frame extraction method using differences between frames and gesture section

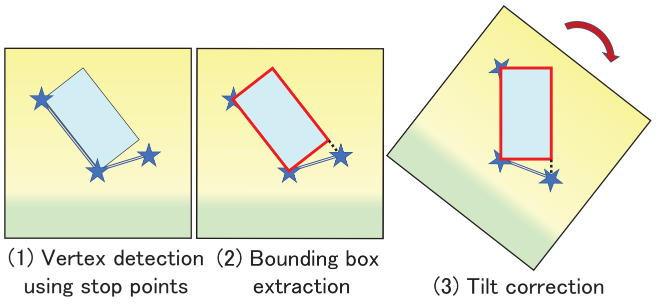

(1) Vertex detection using stop points
(2) Bounding box extraction
(3) Tilt correction

Figure 7: Extraction method as a bounding box with tilt correction

The vertical axis of Fig. 6 shows the variation of the difference d_q between frames along the frame number q in this operation. Since the field of view changes in the zoom-in and zoom-out sections, the difference between frames increases. Conversely, the difference is small in the other still section, but small fluctuations remain due to camera shakes for a wearable camera.

Set two threshold values L_1 and L_2 to extract the target frame based on the gesture section (3). L_1 is the threshold for detecting zoom-in and zoom-out; L_2 is the threshold for detecting still frames. The gesture section (3) is detected based on the stop points by the procedure shown in Fig. 4. Subsequently, the target section (2) is detected as the section between zoom-in and zoom-out, based on L_1. Finally, for the target section (2), the target frame is detected as the frame with the smallest difference d_q in each section where d_q is below L_2, as shown in Fig. 6.

3.2.3 Bounding Box Extraction Method

A method to extract the target area as a bounding box, similar to other object detection methods, is shown. In addition, the box's tilt can be corrected with this method. This tilt correction can be applied to rectangular target areas that may be placed at an angle, such as books on a bookshelf. Consequently, it is expected that the discrimination accuracy through template matching and others will be improved.

This method uses stop points; the gesture is sequentially stopped at three vertices, as shown in Fig. 7 (1). And, the

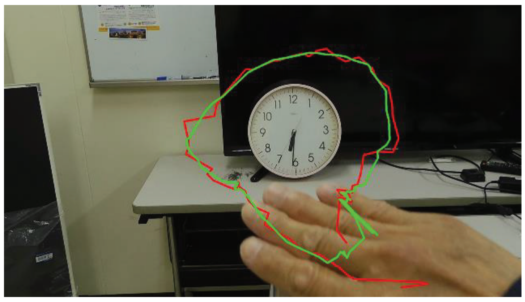


Figure 8: Indication accuracy of hand gestures

bounding box can be estimated by these stop points. When correcting the tilt, the edge that will become the bottom edge is specified first to indicate the direction of the rotation.

Figure 7 (2) shows that the longer of the two edges (subsequently called long edge), which should be accurate according to the tilt, is selected. Moreover, the other edge is estimated as a line segment perpendicular to the long edge. The remaining edges are estimated as parallels of these edges. Hence, the bounding box is estimated with these edges.

If the number of stop areas exceeds three, the three areas are selected based on the number of points $\tilde{p}_q$ on the detected trajectory included in each stop area. When the box's tilt must be corrected, the image is rotated so that the long edge is vertical or horizontal, according to the above-mentioned specified order of the edges, as shown in Fig. 7 (3).

4 IMPLEMENTATION AND EXPERIMENTS

4.1 Implementation of Gesture Trajectory Detection Method

An experimental system was constructed to evaluate the proposed target area extraction method. The PC used has a CPU of i9-10850K 3.6 GHz, 64 GB memory, and a GPU of Geforce RTX 3090 with 24 GB memory. In addition, the Logitech Web camera C920n was directly connected to the laptop computer, and videos were shot at a resolution of $1,920 \times 1,080$ pixels and 30 fps. This system was implemented on Windows 10, Python Ver. 3.8.13 as the program and Pytorch Ver. 1.7.1 with CUDA Ver. 11.5 to use YOLO, OpenCV-Python Ver. 4.5.5.64 for image and video manipulation, and TensorBoard Ver. 2.8.0 to analyze the model training results. YOLOv5s, a highly efficient model of YOLO, was implemented by adding the necessary functions to the publicly available program [2]. Similarly, the publicly available "Egohand Dataset" [1] was used for the training data, by which the model to detect each of own and opponent's left and right hands is trained, as shown in Fig. 2. It comprised 4,800 data taken at 48 locations, of which 3,840 were used as training data and 960 as validation data.

As a preliminary experiment, this trained model was used to evaluate the extraction accuracy of the target area from the gesture trajectory of the right hand. Figure 8 shows the results, where the red line shows the median trajectory and the yellow–green line shows the detected trajectory. Note that

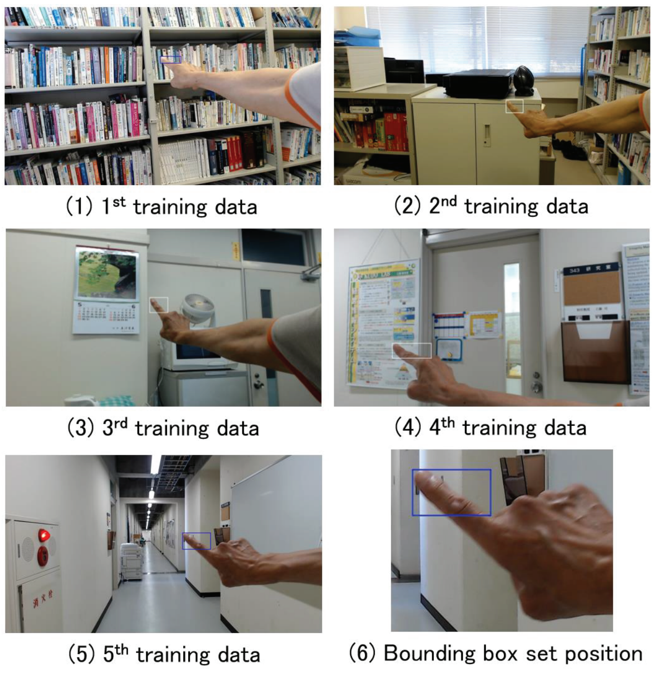


Figure 9: Training data of fingertip for transfer learning

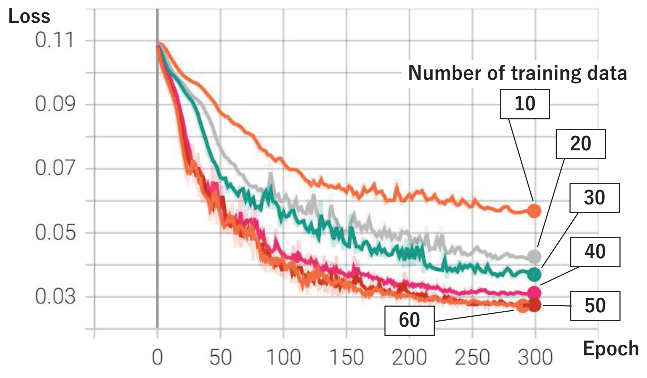


Figure 10: Changes in validation loss in transfer learning

the extraction of the target gesture section was omitted in this experiment.

Consequently, it was observed that the hand was too large to indicate the target area accurately and that the indication accuracy was insufficient.

4.2 Experiments on Improving Indication Accuracy Using Transfer Learning

For the issue mentioned in Sec. 4.1, the training efficiency of transfer learning utilizing this existing model was evaluated to efficiently use an arbitrary body part or instruction tool. A total of 75 images, 15 for each of the five different environments, were shot, as shown in Figs. 9 (1–5), and a bounding box label was created for each fingertip, as shown in Fig. 9 (6). This box includes the fingertip, and the lower right is the lowest position near the fingertip. In this experiment, the fingertip was pointed to the upper left. The data were divided into 60 training data and 15 validation data.

In YOLO's transfer learning, the number of layers to be fixed can be specified by the "freeze" parameter. In this experiment, it was set to 10, corresponding to the "backbone" of YOLO, where features were extracted. The number of training data was from 10 to 60 for every 10, and the number of validation data was fixed to 15. The number of training epochs was 300. Figure 10 shows the transition of the validation loss of the bounding box position obtained through TensorBoard. The numbers in the boxes indicate the number of training data. For 60 training data, the verification loss converged, so training was completed before 300 epochs.

Figure 10 shows that the accuracy improves as the training data increases, and the accuracy improvement converges at about 50 data.

Subsequently, 20 test data were prepared for the clock used in the preliminary experiment, as shown in Fig. 8, and evaluated the effectiveness of transfer learning. Figure 11 (1) shows the connecting result of the fingertip points detected by YOLO using the model before transfer learning, that is, the hand model; (2) shows the result using the model after transfer learning with 60 data.

In Fig. 11 (1), since some points are not detected, the right side is missing. In Fig. 11 (2), though there is one point that is far from the original position (a noise), this point can be removed in the median trajectory.

Figure 12 shows the number of undetected points and noises

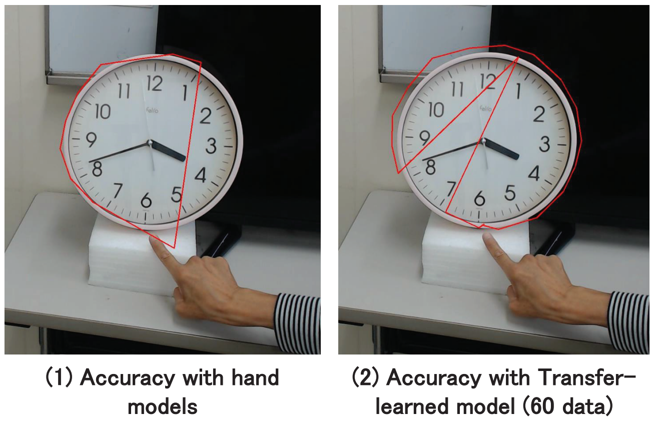

(1) Accuracy with hand models
(2) Accuracy with Transfer–learned model (60 data)

Figure 11: Detected trajectory of gesture using hand and transfer-learning model

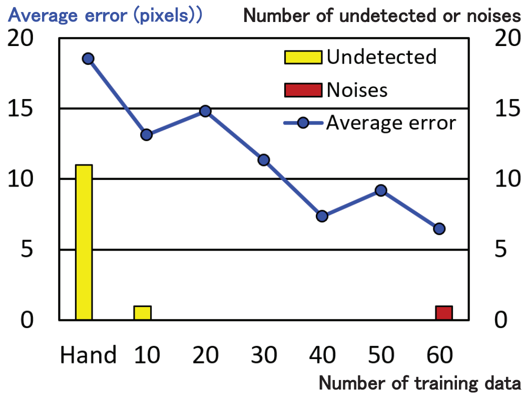


Figure 12: Accuracy improvement with an increase in transfer-learning data

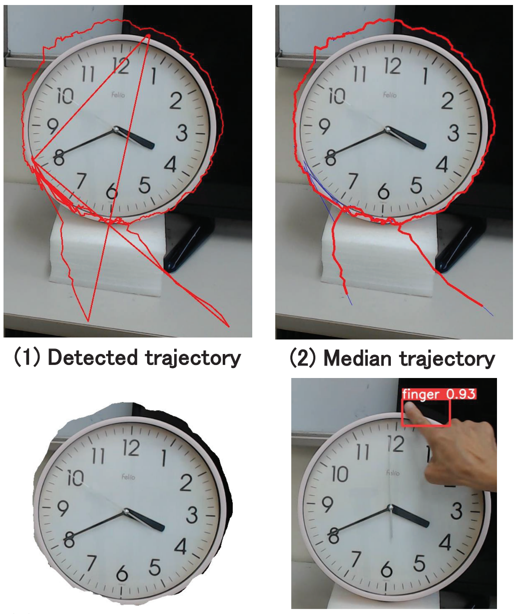

(1) Detected trajectory
(2) Median trajectory
finger 0.93
(3) Extracted target area
(4) Detected position and indication error

Figure 13: Accuracy evaluation of target area extraction using fingertip gestures

and the average error of detected coordinates in pixels for each number of transfer-learning data. Note that "hand" indicates the case where the original hand model was used. Undetected points and noises were excluded from the calculation of the average error.

While more than half of the points are not detected in the "hand" model before transfer learning, this number is reduced to one point after transfer learning with 10 data. This indicates that transfer learning with even a small number of data (about 10) can improve detection accuracy. As in Fig. 10, the accuracy improved as the number of training data increased, but in this experiment, the improvement in accuracy almost converged at 40 data.

In this experiment, the target area was extracted using the basic target area extraction method mentioned in Sec. 3.2.1. The accuracy threshold V in Eq. (1) was set to 0.4; the number of points to calculate the median and the moving average was set to five. Note that the end points of the trajectory were omitted for the median trajectory, and the next point was calculated with three points. In addition, for the stop area detection, the threshold H in Eq. (4) was set to 1% of the narrower axis of the screen (108 pixels).

Figure 13 shows the results. (1) shows the trajectory of the points detected by YOLO with the highest accuracy for each frame; (2) shows the median trajectory; (3) shows the extracted target area based on the detected trajectory where the outside of the target area was set to white.

Compared Fig. 13 to Fig. 8, the accuracy of the target area is improved using the fingertip. In addition, the area untargeted for detection at the bottom of the clock could be removed using the detected trajectory. However, the outline of the upper part of the clock deviated from the actual outline. Thus, I investigated the detected points and observed that the indicated position of the fingertip deviated, as shown in Fig. 11 (4). This was caused by the viewpoint error between the human eye and the camera because a fixed camera was used.

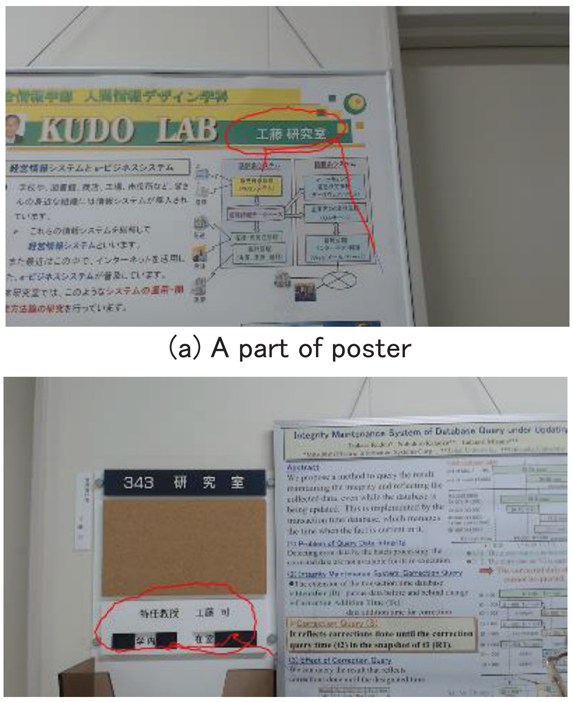


Figure 14: Evaluation target for basic target area extraction

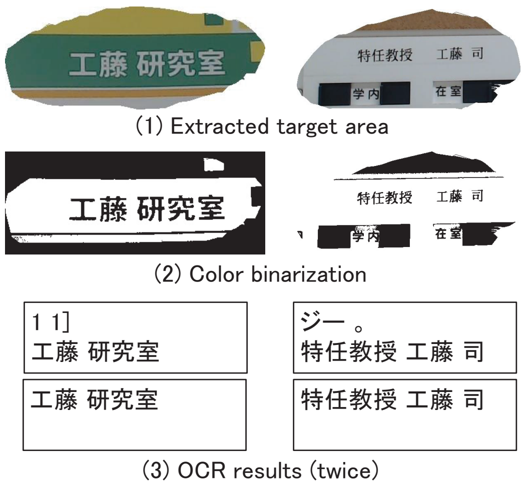
5 APPLICATIONS AND EVALUATIONS OF PROPOSED METHOS

To evaluate the effectiveness of the proposed method in actual business, object recognition accuracies were evaluated in the three cases assumed in Sec. 3.2. In addition, the settings are similar to the experiments shown in Fig. 13.

5.1 Evaluation of Basic Target Area Extraction: Recognition of Bulletin Boards

As the evaluation target of the basic target area extraction method mentioned in Sec. 3.2.1, the poster and the nameplate of the laboratory shown in Figs. 14 (a) and (b), respectively, were used, and OCR on the extracted target area was performed. The red line in Fig. 14 shows the median trajectory of the gesture.

First, the same procedure shown in Fig. 13 was used to extract the target area shown in Fig. 15 (1). Subsequently, the area was binarized, as shown in (2), to eliminate the influence of color. In this case, the area of the poster in 14 (a) is inverted because the text is white. This binarization was performed using the *threshold* method of OpenCV-Python, and the binarization threshold was set to 127 (the middle value). Finally, character recognition was performed using OCR from the binarized image shown in (2). OCR was implemented using Tesseract ver. 5.2.0 and OpenCV-Python's pyocr class. These gestures and procedures were performed twice.



```
1 1]
工藤 研究室
```

```
ジー 。
特任教授 工藤 司
```

```
工藤 研究室
```

```
特任教授 工藤 司
```

Figure 15: Procedure and results of object recognition by OCR

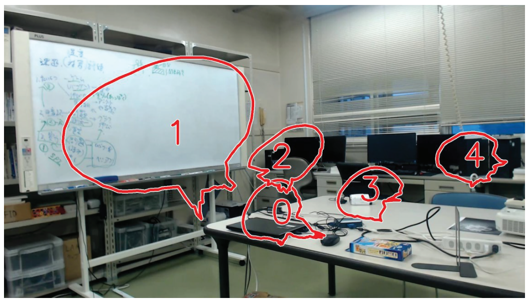


Figure 16: Indication results of fixed-asset locations

Hence, Fig. 15 (3) shows that although extra characters were included, the target characters could be recognized in all cases.

5.2 Target Frame Extraction: Fixed-Asset Location Management

To evaluate the target frame extraction method mentioned in Sec. 3.2.2, the method was applied to fixed-asset location management. The fixed-asset audit verifies the location of the assets and the fixed-asset label attached to each asset. In addition, the fixed-asset management database, which manages detailed information, such as fixed-asset names and administrators, can be retrieved with the fixed-asset number on the fixed-asset label. This experiment verified the possibility of information extraction from the video.

To record the location of the asset, a fixed camera was used, and the target area was indicated sequentially with the gestures, following the same procedure shown in Fig. 13. However, in this application, it was necessary to clarify the items' location, so the target area in Fig. 5 (2) was not extracted, but only the detected trajectory was drawn. In addition, the order in which each asset was indicated was described, in each trajectory.

Consequently, Fig. 16 shows that the position of each asset could be recorded only by gestures. Moreover, since this method does not require precise positions such as the outline of the targets, gesture indications were easy regardless of the type or view of the target.

Next, the recognition accuracy of fixed-asset labels was evaluated. Since the fixed-asset label needs to be taken by close-up photography, the target frames were extracted following the method mentioned in Sec. 3.2.2, and OCR was performed on the frames. Table 1 shows that a PC, a smart glass (Glass), and a tablet were used. Moreover, as shown in Fig. 17 (2), the fixed-asset number (BB170090) and registration date (20171109) in the second line were recognized.

In this experiment, the target object was taken along the procedure as shown in Fig. 6. That is, it was taken at a wide angle (1); then, the camera was moved closer to take the fixed asset (2); finally, the camera was moved away again, and the frame was indicated by gesture (3). The difference between frames (hereinafter, difference) in Fig. 6 was calculated using the OpenCV-Python methods. The frames were converted to monochrome images using the $cvtColor$ method, and each difference image between adjacent frames using the $absdiff$ method. Subsequently, the difference was calculated by Eq. (6).

$$d_i = \sum_{j=0}^{255} n_{ij} * j/N \tag{6}$$

Here, i is the frame number; j is the luminance; n_{ij} is the number of pixels with luminance j in the frame number i; N is the number of pixels in each frame. The variation d_i is divided by N for normalization.

Thresholds L_1 and L_2 are set to 10 and 5, respectively. From the gesture section (3) to backward, the section where ten consecutive frames exceed L_1 is detected as the zoom-out section. Similarly, with this zoom-out section as the starting point, a section where 15 consecutive frames were below L_1 was detected as the target section (2). Figure 18 shows the transition of d_i along the frame number and the detected results.

The frames with the minimum d_i for each section below L_2 in the target section were extracted to perform OCR to recognize the fixed-asset numbers and registration dates. The implementation of OCR was similar to Sec. 5.1. However, binarization was also omitted, since the target area extraction was omitted. Fixed-asset numbers began with "BB" in this case. Hence, strings matching the following regular expression were extracted for the OCR results.

$$\backslash nBB\backslash S\{6\}\backslash s * \backslash S\{8\}\backslash n$$

That is, the target line is surrounded by newline characters and comprises eight characters, starting with BB, one or more spaces, and eight characters. Additionally, in order to correct OCR errors, at most one space was allowed in the strings of each fixed-asset number ($BB\backslash S\{6\}$) and registration date ($\backslash S\{8\}$), and it was removed from the OCR results.

Table 1 shows the recognition results under this condition. In the "Result" column, "◯" indicates the correct result, and "NG" indicates the incorrect result. The "Target" column shows the number of frames with the target minimum difference in the target section, and the "Ratio" column shows the ratio of the frames that satisfy the above conditions.

Table 1: Target object in experiment

Target	Asset num.	Number	Result	Reg. date	Number	Result	Target	Ratio
PC	BB210104	9	◯	20211214	9	◯	20	45%
Glass	BB170090	8	◯	20171109	8	◯	16	50%
Tablet	BB15Z061	15	◯	20151022	18	◯	20	90%
	BB152061	3	NG					

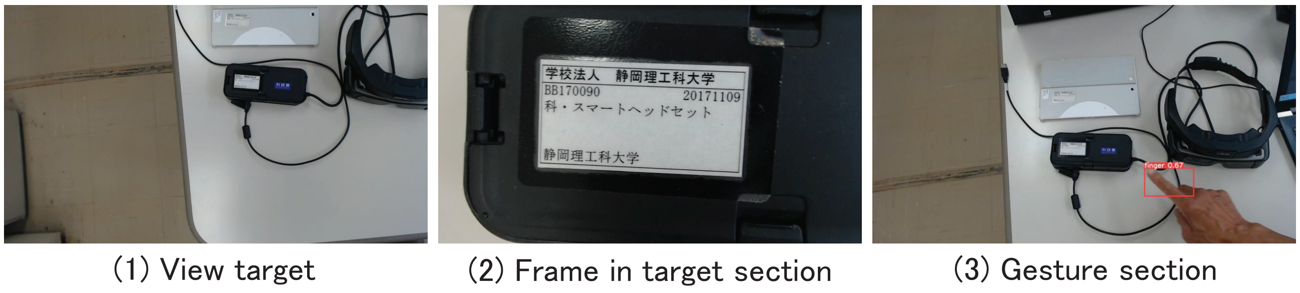


Figure 17: Target object-taking procedure to indicate target frame

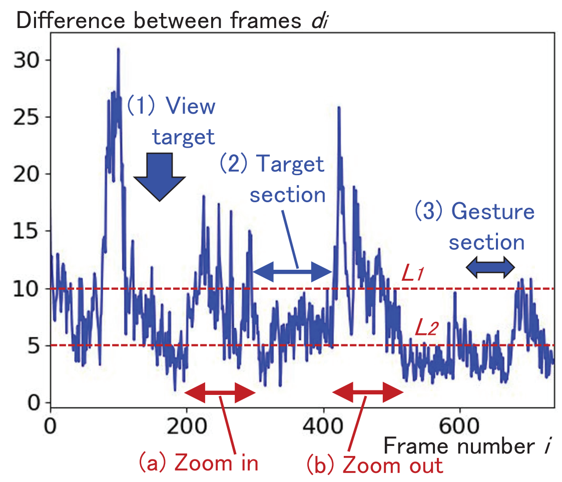


Figure 18: Transition of difference between frames for Glass

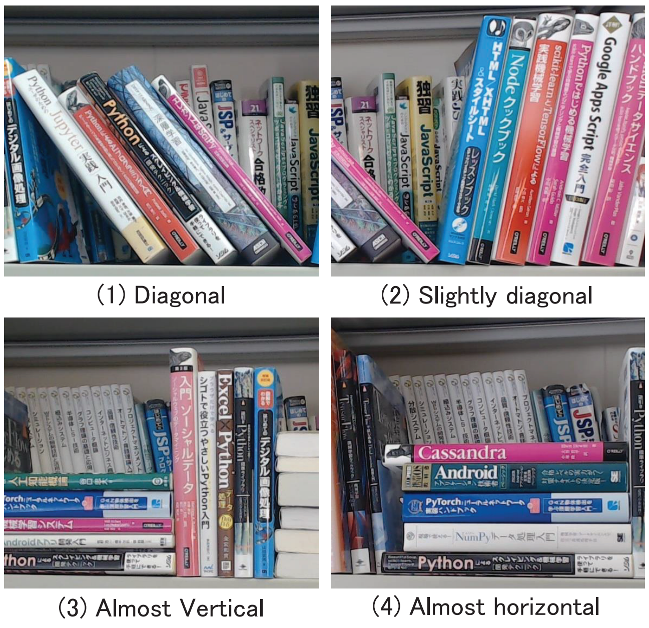


Figure 19: Evaluation target for book recognition with tilt correction

Consequently, while “Z” was misrecognized as “2” in 3 out of 18 fixed-asset numbers of the tablet, the others were correctly recognized. However, the percentage of frames satisfying the above conditions differed depending on the case.

5.3 Bounding Box Extraction and Tilt Correction: Book Recognition

In order to evaluate the bounding box extraction method with tilt correction, as shown in Sec. 3.2.3, 20 books arranged at four different tilts were used, as shown in Fig. 19. Moreover, the effect of tilt correction on object recognition using template matching was evaluated. The $matchTemplate$ function of OpenCV-Python was used for template matching, and the $normalized\ cross-correlation\ matching\ method$ was used for the matching method.

First, to create a template image, each target book was manually cut out from the image in Fig. 19 so that it was arranged vertically, as shown in Fig. 20 (1). Next, as shown in Figs. 20 (2) and (3), the target book was extracted as a bounding box with and without tilt correction, as mentioned in Sec. 3.2.3, and placed in a white image of 700 × 700 pixels, respectively. The green rectangles in Figs. 20 (2) and (3) are the results of template matching shown below.

For each image, template matching was performed between the template image shown in Fig. 20 (1) and each of the images shown in (2) and (3) to obtain the matching position and accuracy. Here, when the tilt was corrected, as shown in Fig. 20 (2), the template image was generally matched to the correct position. Similarly, other images were generally matched to the correct position when the tilt was corrected. However,

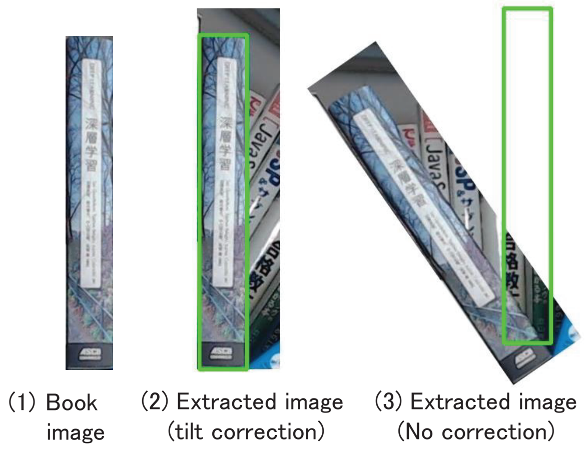

(1) Book image
(2) Extracted image (tilt correction)
(3) Extracted image (No correction)

Figure 20: Book template-matching results

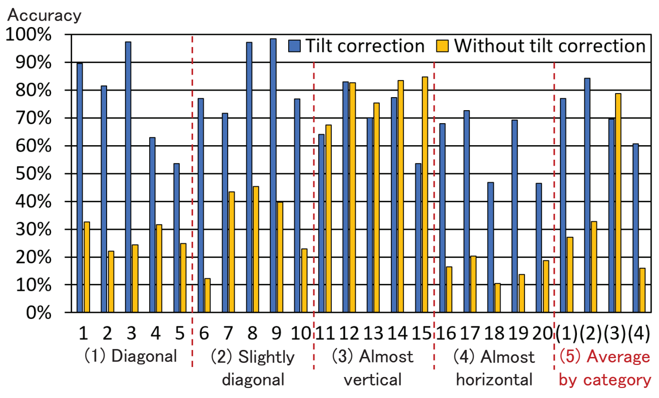


Figure 21: Evaluation of template-matching accuracy improvement through tilt correction

as shown in Fig. 20 (3), there was an error in the matching position of the template images in the case of this book.

Figure 21 shows the evaluation results of the matching accuracy for all books. The numbers (1) to (4) on the horizontal axis correspond to the numbers of the arrangement types in Fig. 19; (5) shows each average of the cases of (1) to (4). The accuracy of the tilt correction case was advanced for the tilted books, as shown in (1), (2), and (4) in Fig. 21. However, the accuracy of the no tilt correction case was advanced for the books placed almost vertically, as shown in (3). Furthermore, the accuracy was different for each book.

To analyze this phenomenon, template-matching positions were verified for typical cases. The maximum accuracy of the tilt correction case (horizontal axis number: 9) is shown in Fig. 22 (1); the minimum accuracy (also 20) is shown in (2), and the case where the tilt correction result deteriorated compared to the no tilt correction result (also 15) is shown in (3). In each case, the left image was the template image, and the right image was the extracted image. (3) was the only case of no tilt correction. In (1), the extracted image included the entire area of the target book, while in (2), the left side of the book was missing. In (3), the bounding box (extracted target area) was tilted due to an error in the gesture indication, although the books were placed vertically. Consequently, the accuracy deteriorated with tilt correction.

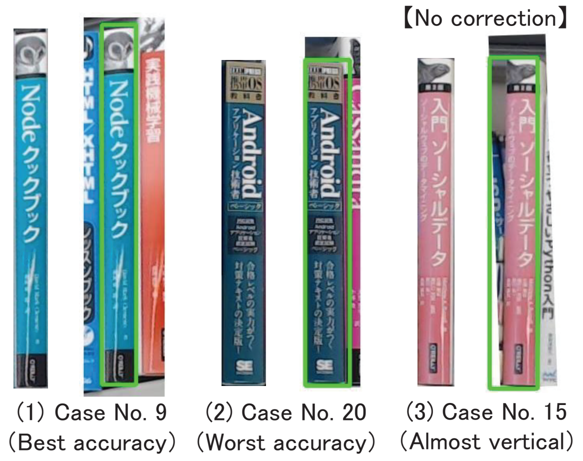

【No correction】
(1) Case No. 9 (Best accuracy)
(2) Case No. 20 (Worst accuracy)
(3) Case No. 15 (Almost vertical)

Figure 22: Influence analysis of gesture indication error on template-matching accuracy

The results show that the target area extraction with tilt correction is effective for template matching accuracy (recognition accuracy) for the target placed diagonally. Conversely, the gesture indication error caused an accuracy deterioration, especially when the target was placed vertically.

6 DISCUSSION

In this study, a method to extract the target area by gesture trajectory detection using YOLO was proposed, and its effectiveness was evaluated through several experiments. First, since only a specific part of the body, such as the fingertip, was used for gestures, it was easy to prepare the training data. Second, practical accuracy was achieved with 60 training data through transfer learning. This shows that this method can be easily applied to any body part and indication device according to the purpose.

However, the gesture trajectory detection was performed on videos and noises, and extra sections were included, as shown in Figs. 2 and 13 (1). Hence, this study showed that the noises were removed using medians, and the target sections were extracted using stop points. Consequently, as shown in Fig. 13 (3), it was possible to extract the target area without these influences. Furthermore, experiments showed that the target area could be extracted without being affected by noise, even in actual use cases.

Though object detection using deep learning can be performed automatically, this method requires the action of gestures. Conversely, this method can extract the target areas more flexibly, as shown by the actual application cases in Secs. 5.1 and 5.3. Any part of an image with no special features can be extracted, as shown in Figs. 14 and 15 (1). Furthermore, the target area can be extracted as a bounding box, and its tilt can be corrected when necessary, as shown in Fig. 20. In addition, as shown in Table 1, Figs. 15 (3) and 21, by extracting the target area in this method, object recognition using simpler recognition methods, such as OCR or template matching, is possible.

Though the experiments in this study were conducted in a laboratory, they targeted the use case of the factory operation shown in Fig. 1. First, the entry of workers to the inventory storage area, also the target container, can be identified by the recognition of the bulletin boards shown in Sec. 5.1. Second, since there are thousands of containers in the factory, the picking process requires instructions not only for the containers to be picked but also for the optimal route. This can be indicated by the method shown in Sec. 5.2 as shown in Fig. 16. Similarly, images of containers or container labels can be extracted and stored for confirmation during picking as shown in Fig. 17. Third, the images of the parts in the factory are stored in a database. Thus, parts in containers can be efficiently discriminated by using the tilt-corrected bounding box and template matching described in Sec. 5.3.

With the spread of IoT and mobile cameras, long tasks and operations are recorded through videos, and it is important to record the data to specify the target frame and area. Section 5.2 showed the method for this, by which the area and the target frame can be extracted. In addition, the frame can be specified even for close-up photography using gestures and differences between frames. For example, when inspecting a building while constantly taking video with the smart glass shown in Fig. 1 (1), the location of abnormalities can be recorded by gestures without operating the camera, as shown in Figs. 16 and 17 (2).

Conversely, when using a fixed camera, as shown in Figs. 13 (4) and 22, errors of gesture indication occur due to a viewpoint error between a human and a camera. For this problem, the use of the above-mentioned smart glass effectively performs gestures while checking the position displayed on the screen is considered. However, there is a challenge for such a mobile camera, since it moves, as shown in Fig. 18. To construct the trajectory of the target area shown in Fig. 5 (2), it is necessary to correct the angle, position, and magnification between frames. Therefore, applying this method to mobile cameras is the next challenge.

7 CONCLUSIONS

When performing object recognition with mobile cameras such as smart glasses, it is often necessary to perform object detection beforehand. In object detection, various studies using deep learning have been actively conducted. However, since it is necessary to prepare training data for each target, it is difficult to prepare training data when there are various targets.

To solve this problem, a method for extracting the target area from the gesture trajectory detected using YOLO is proposed. Furthermore, through experiments, it was shown that practical accuracy can be achieved from less training data by transfer learning and that targets can be extracted flexibly according to purposes.

Future studies will focus on improvements in the accuracy of gesture indication using mobile cameras.

REFERENCES

[1] S. Bambach, S. Lee, D. Crandall, and C. Yu, "Lending A Hand: Detecting Hands and Recognizing Activities in Complex Egocentric Interactions," IEEE Int. Conf. Computer Vision (ICCV), pp. 1949-1957 (2015), https://public.roboflow.com/object-detection/hands (referred May 19, 2022).

[2] G. Jpcher, et.al. "YOLOv5," https://github.com/ultralytics/yolov5 (referred May 17, 2022).

[3] T. Kawanaka, and T. Kudo, "Inventory satisfaction discrimination method utilizing images and deep learning," Procedia Computer Science, Vol. 126, pp. 937–946 (2018).

[4] T. Kudo, "A Proposal for Article Management Method Using Wearable Camera," Procedia Computer Science, Vol. 176, pp. 1338–1347 (2020).

[5] T. Kudo, "Moving Object Detection Method for Moving Cameras Using Frames Subtraction Corrected by Optical Flow," Int. J. Informatics Society, Vol. 13, No. 2, pp. 79–91 (2021).

[6] T. Kudo, "Contour Detection Method by CycleGAN Using CG Images as Ground Truth," Int. J. Informatics Society, Vol. 14, No. 1, pp. 3–12 (2022).

[7] T. Kudo, "Application Method of Deep Learning Model Trained with CG Images to Real Images," Procedia Computer Science, Vol. 192, pp. 1484–1493 (2021).

[8] T. Y. Lin, et.al., "Microsoft coco: Common objects in context," European Conf. Computer Vision, pp. 740–755 (2014).

[9] T.Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," Proc. IEEE Conf. Computer Vision and Pattern Recognition, pp. 2117–2125 (2017).

[10] T.Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection." Proc. IEEE Int. Conf. Computer Vision, pp. 2980–2988 (2017).

[11] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Y. Fu, and A. C. Berg, "SSD: Single shot multibox detector," European Conf. Computer Vision, pp. 21–37 (2016).

[12] L. Liu, W. Ouyang, X. Wang, P. Fieguth, J. Chen, X. Liu, and M. Pietikáinen, "Deep learning for generic object detection: A survey," Int. J. Computer Vision, Vol. 128, No. 2, pp. 261–318 (2020).

[13] A. Mujahid, M. J. Awan, A. Yasin, M. A. Mohammed, R. Damaševičius, R. Maskeliūnas, and K. H. Abdulkareem, "Real-time hand gesture recognition based on deep learning YOLOv3 model," Applied Sciences, Vol. 11, No. 9, 4164 (2021).

[14] M. Mukhiddinov, and J. Cho, "Smart glass system using deep learning for the blind and visually impaired," Electronics, Vol. 10, No. 22, p. 2756 (2021).

[15] M. S. Murshed, C. Murphy, D. Hou, N. Khan, G. Ananthanarayanan, and F. Hussain, "Machine learning at the network edge: A survey," ACM Computing Surveys (CSUR), Vol. 54, No. 8, pp. 1–37 (2021).

[16] M. Oudah, A. Al-Naji, and J. Chahl, "Hand gesture recognition based on computer vision: a review of techniques, " J. Imaging, Vol. 6, No. 8, 73 (2020).

[17] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," Proc. IEEE Conf. Computer Vision and Pattern

Recognition, pp. 779–788 (2016).

[18] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards real-time object detection with region proposal networks,” Advances in Neural Information Processing Systems, pp. 91–99 (2015).

[19] Y. Shi, Y. Li, X. Fu, K. Miao, and Q. Miao, “Review of dynamic gesture recognition,” Virtual Reality & Intelligent Hardwar, Vol. 3, No. 3, pp. 183–206 (2021).

[20] P. Viola, and M. Jones, “Rapid object detection using a boosted cascade of simple features,” Proc. 2001 IEEE Computer Society Conf. Computer Vision and Pattern Recognition, Vol. 1, pp. 511–518 (2001).

[21] Z. Wang, L. Jin, S. Wang, and H. Xu, “Apple stem/calyx real-time recognition using YOLO-v5 algorithm for fruit automatic loading system,” Postharvest Biology and Technology, Vol. 185, No. 111808 (2022).

[22] J. Wolfartsberger, J. Zenisek, and N. Wild, “Data-driven maintenance: combining predictive maintenance and mixed reality-supported remote assistance,” Procedia Manufacturing, Vol. 45, pp. 307–312 (2020).

[23] Q. Zhao, T. Sheng, Y. Wang, Z. Tang, Y. Chen, L. Cai, and H. Ling, “M2det: A single-shot object detector based on multi-level feature pyramid network,” Proc. AAAI Conf. Artificial Intelligence, Vol. 33, pp. 9259–9266 (2019).

(Received: October 29, 2022)
(Accepted: December 18, 2022)

Tsukasa Kudo received his BSc and ME from Hokkaido University in 1978 and 1980, and Dr. Eng. from Shizuoka University in 2008. In 1980, he joined Mitsubishi Electric Corp. He was a researcher of parallel computer architecture and engineer of business information systems. Since 2010, he has been a professor at Shizuoka Institute of Science and Technology. His current research interests include deep learning and database applications. He is a member of IEIEC and IPSJ.

Regular Paper

Improve Measuring Suspiciousness of Bugs in Spectrum-Based Fault Localization With Deep Learning

Hitoshi Kiryu[†], Nobutoshi Todoroki[‡], Satoshi Suda[‡], Shinpei Ogata*, and Kozo Okano*

[†]Graduate School of Engineering, Shinshu University, Japan
21w2025g@shinshu-u.ac.jp
[‡]Advanced Technology R&D Center of Mitsubishi Electric
{Suda.Satoshi@ay, Todoroki.Nobutoshi@dn}.mitsubishielectric.co.jp
*Faculty of Engineering, Shinshu University, Japan
{ogata, okano}@cs.shinshu-u.ac.jp

Abstract - Localizing Faults is integral for debugging in developing software. Spectrum-Based Fault Localization (SBFL) is a technique to localize faults. SBFL calculates the suspiciousness scores for each line in a source code using execution coverages of tests that represent which lines are executed in which tests. Some studies apply a deep learning techniques to SBFL. In these studies, the suspiciousness scores are calculated by giving a virtual coverage to a trained model. This paper proposes a method to calculate the suspiciousness scores of lines in source code from the execution coverages, and evaluate an effectiveness of a bootstrapping sampling method. The proposed method introduces the virtual coverage that activates consecutive lines whose execution is the same in any execution coverage. The method provides a ranking based on the score. As a result of the evaluation, it is confirmed that the proposed virtual coverage is better than a virtual coverage in previous researches and the sampling method effectively works in training model.

Keywords: SBFL, Fault Localization, Deep Learning

1 INTRODUCTION

When a problem caused by code is found in software development or maintenance, it is necessary to localize and fix bugs. Generally, such a debugging needs a lot of time and human works. Many techniques to localize faults and fix bugs have been studied to support developers in debugging.

One example of a bug-fixing technique is GenProg [1], which outputs code that passes all the test suites with a genetic algorithm. In techniques for localizing bugs, Various studies [2]–[5] have been conducted. These studies propose methods to identify statements that cause bugs by using information such as bug reports, trace information, and visualization.

Another technique for localizing the fault is Spectrum-Based Fault Localization (SBFL). SBFL localizes faults in the source code based on the execution coverage and test results of each test. The technique calculates the suspiciousness of each statement and provides a ranking based on the suspiciousness. The basic idea of SBFL is that a line executed in a failed test is more likely to contain a bug, while a line executed in a successful test is less likely to contain a bug.

As for difficulty of Fault Localization with machine learning, One survey [6] mentions that the lack of large labeled data sets and imbalance of training data makes the fault localization using deep learning more difficult. Especially, the problem of the imbalance of data often appears in fault localization using test cases. This is because a quantity of failed test cases are generally less than that of a quantity of passed test cases. In learning with imbalanced data, minority class can be ignored in prediction. For example, If A class occupies 99 percent of data, a neural network can get more than 99 percent of precision by just classifying inputs to the class.

In this paper, we propose a method based on SBFL techniques and deep learning techniques to support developers in debugging fault in source code. The method we propose also address handling imbalanced data. The proposed method shall localize a single fault in source code.

1.1 Related Work

Deep learning is a technology that has been showing results in a wide variety of fields, including image recognition and natural language processing. The field of fault localization is no exception either. Ikeda et al. [7] proposed the method that localizes a fault with a neural network. They trained the network to predict test results from execution traces and used ablated traces to localize a fault method.

Deep learning technique for NLP is utilized for fault localization. Guo et al. [8] proposed GraphCodeBERT that is a pre-trained model for programming language that considers the inherent structure of code. The model is based on BERT [9], and pre-trained in some tasks including Masked Language Modeling. Source code, text data including comments and data flow graph that represents variable relation are used as explanatory variables.

In the field of SBFL, the study [10] has been conducted to compute the suspiciousness using deep learning. In this study, three architectures, RNN (Recurrent Neural Network), CNN (Convolutional Neural Network), and MLP (MultiLayer Perceptron), are utilized to compute the suspiciousness of bugs, and CNN shows the best results. In another study [11], which calculates the suspiciousness using RBF networks (Radial Basis Function Network), concluded that RBF networks are more effective than existing methods such as Ochiai in fault localization. SBFL techniques with deep learning use execution coverage data that often consist of few failed tests and a lot of passed tests as explanatory variables to predict test results. Such an imbalanced data occurs deterioration of a performance of the deep learning model. This is because the model have to predict failed test, namely, the minority of data. Zhang et al. [12] proposes a SBFL method using deep learning technique to deal with data imbalance. The SBFL method applies upsampling technique that increase quantity of minority data in neural network training. The experiment shows the method performed best with an upsampled data that consist of failed and passed tests in the same ratio. These studies that address SBFL using deep learning train a neural network model to predict test results, and utilize a virtual coverage that is one certain line is executed to calculate the suspiciousness scores from the trained model.

1.2 The Approach

This paper proposes a method to compute the suspiciousness of lines in the source code from the program spectrum with deep learning. Using a trained model to predict test results from the execution coverages, suspiciousness scores are measured for each line. In training model process, a bootstrapping sampling method is utilized to deal with imbalanced data that composed with a lot of passed tests and a few failed tests. We propose a virtual coverage that differs to a virtual coverage in previous researches about SBFL with deep learning techniques. In the proposed virtual coverage, one certain block of lines that are commonly executed across coverage is executed rather than one certain line. The proposed virtual coverage is input to the trained model in order to measure the impact of each line on bug prediction in the trained model. We treat the impact as suspiciousness scores. A ranking of lines that are likely to cause bugs based on the suspiciousness scores are provided.

In evaluation experiment, the results indicates the proposed virtual coverage is better than the virtual coverage in previous researches, and bootstrapping sampling method contributes improving performance of fault localization. From the experiment, We conclude that the proposed virtual coverage and the sampling method effectively work in fault localization.

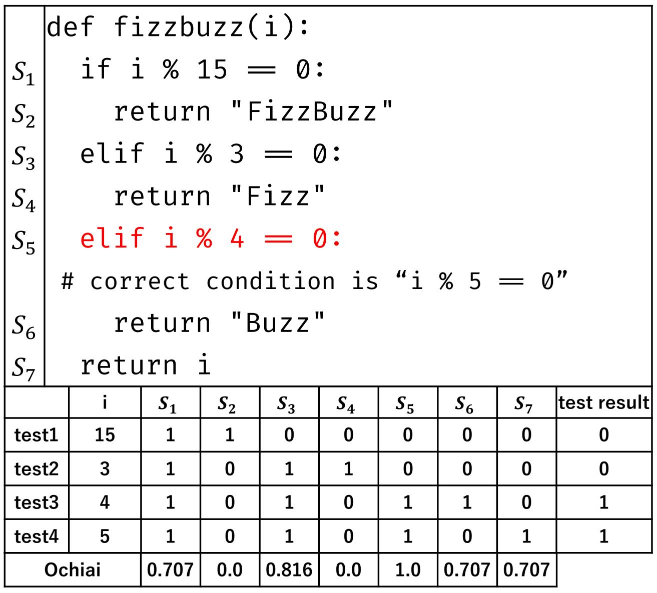
In the following sections, Section 2 describes the related techniques for this research. Section 3, explain about the proposed method. Section 4 shows the results in the evaluation experiments and Section 5 discuss the results in the experiments. Finally, we conclude in Section 6.

2 PRELIMINARIES

This section explains techniques, which compose the proposed method.

2.1 Spectrum-Based Fault Localization (SBFL)

SBFL is a technique to localize faults in source code by calculating suspiciousness scores that represent probability of causing bugs for every statement. Generally, a ranking based



```
    def fizzbuzz(i):
S1    if i % 15 == 0:
S2      return "FizzBuzz"
S3    elif i % 3 == 0:
S4      return "Fizz"
S5    elif i % 4 == 0:
     # correct condition is “i % 5 == 0”
S6      return "Buzz"
S7    return i
```

	i	s_1	s_2	s_3	s_4	s_5	s_6	s_7	test result
test1	15	1	1	0	0	0	0	0	0
test2	3	1	0	1	1	0	0	0	0
test3	4	1	0	1	0	1	1	0	1
test4	5	1	0	1	0	1	0	1	1
Ochiai		0.707	0.0	0.816	0.0	1.0	0.707	0.707	

Figure 1: Example of SBFL

Table 1: Four Values for Calculation of Suspiciousness

e_f	Number of failed tests that execute the program element.
e_p	Number of passed tests that execute the program element.
n_f	Number of failed tests that do not execute the program element.
n_p	Number of passed tests that do not execute the program element.

on the scores are built to support developers to find the faults out.

For example, Ochiai [13] and Dstar [14] are metrics to calculate the suspiciousness scores. Here, ‘N’ is the exponent variable of e_f. The score tends to be higher if a line is executed more frequently when the test fails or less frequently when the test succeeds.

$$Ochiai = \frac{e_f}{\sqrt{(e_f + n_f)(e_f + e_p)}} \tag{Ochiai}$$

$$Dstar(* = N) = \frac{{e_f}^N}{e_p + n_f} \tag{Dstar}$$

The metrics for calculating the suspiciousness of a bug are based on four values described in Table 1.

Figure 1 shows an example of SBFL. The source code in the figure is a function that returns the result of FizzBuzz for a given number i as input. The fifth line of the function contains a bug. This is because the conditional statement is incorrect. The suspiciousness scores of each line is calculated by Ochiai based on the results and the coverage of the tests. The suspiciousness is the highest on lines fifth, which are executed only when the test fails. This indicates that the suspiciousness of the buggy lines is calculated properly. Thus,

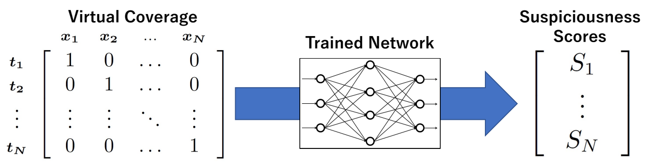


Figure 2: Overview of Score Calculation in The Previous Researches

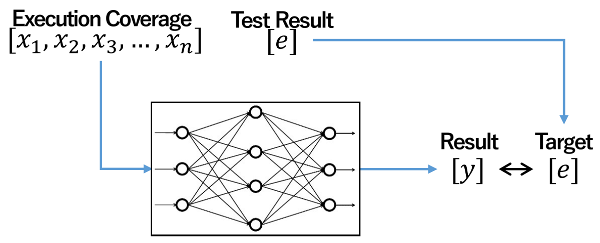


Figure 3: Overview of Training Model

from the test results and the coverage, it is possible to identify the location of bugs. Note that in the example shown in the figure, the score of S7 will get closer to that of S5 when the test cases which input is a multiple of 5 and not a multiple of 3 increase. Furthermore, the score of S3 will decrease when test cases which execute S3 and result pass increase. SBFL focuses on the frequency of test failure in execution of line.

2.2 SBFL using Deep Learning Techniques

Some studies [10]–[12] focus on SBFL that utilize deep learning techniques. In SBFL using deep learning, a virtual coverage is input to a trained neural network model to calculate suspiciousness scores. The model is trained to predict test results from execution coverages. Figure 2 shows overview of a virtual coverage. The virtual coverage in the figure is a execution coverage based on a scenario that only one certain line is executed in a virtual test. In order to measure the contribution that the execution of statement affect to the result prediction, the virtual coverage is input to the trained model. It is considered that the output of the virtual coverage indicates the impact of the statement on the test results. Therefore, the output of virtual coverage is treated as suspiciousness scores.

3 PROPOSED METHOD

This section describes the methodology of SBFL using a Neural Network in detail.

In the proposed method, first, a neural network model learns about a relation between the execution coverage of lines in the source code for each test and the test result. The model takes the coverage as input, and output respectively probability of a test result, pass and fail. In order to measure the impact on decision-making about test results in the trained model, a virtual coverage is input to trained model. The impact on decision-making is considered as the suspiciousness of bugs. The trained model cannot be applied to other source code. This is because the model learns only the relation between a execution of lines in a certain source code and test results.

3.1 Training Network

Figure 3 shows the overview of the training step. The neural network model learns implicit relations between the execution coverages and test results to predict test results.

Generally, the execution coverages collected by tests are composed with a lot of passed test and a few failed tests, namely imbalanced data. A model trained with such imbalanced data tend to ignore the minority data. In SBFL, the model have to emphasize minority data that is a few failed tests. Therefore, the model have to deal with imbalanced data in learning.

In order to handle the imbalanced data, we utilize the bootstrapping sampling method [15] that Yan et al. proposed. In the bootstrapping sampling method, the ratio of minority to majority is set to 1:1 for each mini-batch. Data of the majority group is divided into N pieces and create tentative mini-batches. For each tentative mini-batches, the same quantity of the mini-batch is randomly extracted from the minority group and joined with the tentative mini-batch as a mini-batch.

3.2 Virtual Coverage

We introduce a block of code for a virtual coverage. Source code is segmented into blocks. The block is defined as: For any execution coverage C and consecutive lines $\forall S_i, S_{i+1} \in C$, S_i and S_{i+1} are in the same block if $S_i = S_{i+1}$. Figure 4 shows the example of block segmentation for statements S_n in the example coverages of the tests t_n. The check marks in the Figure means the statement is executed in the test.

The matrix of Virtual Coverage shows the example virtual coverage of the blocks shown in Fig. 4. In the matrix, 1 means that the line is executed and 0 means not executed. Therefore, lines contained within the block are only executed. As shown in Fig. 2, the virtual coverage in previous research is executed by line. The proposed virtual coverage is executed by block.

Virtual Coverage

$$\begin{array}{c|cccccccccc} & S_1 & S_2 & S_3 & S_4 & S_5 & S_6 & S_7 & S_8 & S_9 & S_{10} \\ \hline block_1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ block_2 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ block_3 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ block_4 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ block_5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ block_6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{array}$$

We apply the block segmentation for lines in source code. In the virtual coverage, one certain block, i.e. lines in the same block, are executed. The neural network model learns to focus on patterns that appear in the data and predict the results. Therefore, by inputting the patterns that appear in data as the virtual coverage, the contribution of each pattern to the test results can be effectively extracted. This is the reason why we segmented the source code into the blocks.

Suspiciousness scores are given to each blocks. Hence, lines in the same block have same suspiciousness scores each other. Since lines in the same block have the same execution pattern, SBFL have a limit that difficulty of distinguishing

Table 2: Bugs for The Experiment

Project	Description	Bug Versions	LOC(Lines Of Code)	Tests
Chart	JFreeChart	8	5798	711
Math	Apache Commons Math	24	20803	6265
Lang	Apache Commons Lang	7	13901	677

	t_1	t_2	t_3	t_4	Block Divisions
s_1	✓		✓	✓	1
s_2	✓	✓	✓	✓	2
s_3	✓	✓	✓	✓	
s_4		✓	✓		3
s_5		✓	✓		
s_6	✓			✓	4
s_7	✓			✓	
s_8			✓		5
s_9	✓	✓	✓		6
s_{10}	✓	✓	✓		

Figure 4: Example of Block Segmentation

such lines from each other. The proposed virtual coverage gives blocks suspiciousness scores followingly the limit.

As for granularity of block segmentation, for example, an exception handling can interrupt the execution in middle of a code block. The block segmentation can be more smaller segmentation than code block. Hence, the proposed method can localize faults with the same or smaller granularity than that of code block.

3.3 Extract Suspiciousness of a Bug

The virtual coverage is input to the trained model to caclulate the suspiciousness scores. We consider the output means impact of patterns of lines on results prediction. In other word, the value represents probability of causing bugs. Thus, We treat the values as suspiciousness scores.

The suspiciousness scores are ranked in descending order and output as a result of fault localization. The ranking presents lines and the corresponding suspiciousness scores. As example of Fig. 4, If the second block get the highest suspiciousness score, The ranking has two lines ranked first in the suspiciousness score. Lines that are not executed in failed tests are excluded from the ranking, because they are not likely to contain bugs.

4 EVALUATION EXPERIMENT

In order to evaluate the proposed method, we applied the proposed method to actual bugs in some OSS projects. We collect bugs from Defects4J [16] that is a database and extensible framework providing real bugs to enable reproducible studies in software testing research. We applied the proposed method for bugs in three projects, that is Chart, Math, and Lang. Defects4J has real bug source code and fix patches for each bug. We collected bugs whose fix patch satisfies the following conditions.

- The fix patch has not only add changes : If the patch has only add changes, the original buggy source code have no lines to be fixed.

- Changes in the fix patch spread across multiple chunks : The proposed method aims localizing a single fault, Thus we don't collect such bugs.

The proposed method is applied to the coverage collected from the above three projects. The bugs we collected are shown in Table 2. The table shows bug versions of each project and entire tests and entire LOC(Lines Of Code) of the versions.

Following two types of virtual coverage are compare to evaluate which method gives higher suspiciousness scores to bugs:

- Block (proposed Virtual coverage) : lines in the same block is executed.

- One-Hot (Virtual coverage in the previous researches) : only a certain line is executed.

We collect four rankings based on the scores calculated following four methods:

- One-Hot

- Block

- One-Hot with Bootstrapping Sampling

- Block with Bootstrapping Sampling

The average processing time in the experiment for 79 buggy source code shown in Table 2 was 24.33 seconds. The experiment is conducted with following environment:

- OS : Windows 11 Pro

- CPU: Intel i5-9400F

- Memory: 16GB

- GPU: GTX1660

The cumulative sum charts in the Fig. 5, 6, 7, 8 show the comparison of the experiment results. The charts describe how many y percent of faults are ranked in top x percent. Therefore, when a chart is above another chart, the method of the chart can rank faults more higher.

4.1 Compare of The One-Hot and The Block

Figure 5 shows the comparison of the One-Hot and the Block. The chart of Block is always above one of the One-Hot. In Fig. 6, The chart of Block with bootstrapping sampling is always above one of the One-Hot with bootstrapping sampling.

From the two comparison, The curves of Block always exceed the curves of One-Hot regardless of whether bootstrapping sampling is used.

4.2 Compare of The Effectiveness of Bootstrapping Sampling

Figure 7 shows the comparison of One-Hot and One-Hot with bootstrapping sampling. The chart of One-Hot with bootstrapping sampling is always above or same as another. In Fig. 8, The chart of Block with bootstrapping sampling is mostly above or same as another except 35% of the statements.

From the two comparison, The curves of bootstrapping sampling mostly exceed the curves of method without bootstrapping sampling.

4.3 Statistical Hypothesis Test for TopN%

In order to compare the result of each method, we conducted statistical hypothesis test to TopN% values of each test. Table 3 describes p-values of the Wilcoxon signed-rank test for each combination of the methods. Wilcoxon signed-rank test is a non-parametric statistical hypothesis test for paired data to compare the locations of two populations. In the test, The null hypothesis is that the locations of two populations from paired data are not different significantly, and the alternative hypothesis is that the locations are different significantly. In the right-tailed test for method A vs method B, the alternative hypothesis is that the location of population of method A is greater than the one of B, and in the left-tailed test, the alternative hypothesis is that the location of population of method A is less than the one of B.

If a p-value of a test is greater than a given significance level, the hypothesis of the test is accepted. The hypothesis of right-tailed test means the method A tend to give the statements greater values as rank than the method B, in other word, the method B gives statements higher rank than the method A. This means the method B is more effective to localizing faults.

In order to counteract the multiple comparisons problem, we utilize a significance level corrected by the Bonferroni correction in each testing. the Bonferroni correction tests each individual hypothesis at significance level $\frac{\alpha}{m}$, where α is desired overall significance level and m is the number of hypotheses. This allows the probability of type I error of m testings to be less than or equal to α. In this paper, we test 4 hypotheses at $\alpha = 0.05$, hence, significance level of each test is set to 0.0125.

5 DISCUSSION

As results of Section 4.1, the method using the proposed virtual coverage always above another curve in both results. The results say the proposed virtual coverage gave more lot of bugs higher ranks. This means the proposed virtual coverage can extract the impact on result prediction in trained model. Furthermore, the top two items "One-Hot vs Block," "One-Hot with bootstrapping sampling vs Block with bootstrapping sampling" in Table 3 in Section 4.3 also suggest the rank of buggy line given by method using the block tend to be higher than another one. These results comfirmed that the virtual coverage based on the block segmentation works more effectively in localizing fault compared with the virtual coverage based on one-hot. The lines in the same block is the pattern that the executions of each line are same. Therefore, the virtual coverage based on the pattern can activate the pattern that trained model learned, and enhance the suspiciousness score of a pattern containing buggy lines.

About results of bootstrapping sampling in Section 4.2, the methods using bootstrapping sampling is mostly same or above to the others that don't use the sampling method. In another result in Section 4.3, the bottom two items "One-Hot vs One-Hot with bootstrapping sampling," "Block vs Block with bootstrapping sampling" in Table 3 also suggest the sampling method contribute training of model. The bootstrapping sampling is originally proposed to deal with imbalanced data for classification of movies. These results say the sampling method can prevent the model to excessively emphasize the minority in imbalanced data. Therefore, the results indicate the bootstrapping sampling method is effective in Spectrum-Based Fault Localization.

In order to verify difference of the localization among three projects in the evaluation experiment, Kruskal-Wallis test is conducted. Kruskal-Wallis test is a non-parametric statistical hypothesis test to verify the significant difference between the medians of more than three data. As a result of the test, p-value is 0.788. The value exceeds 0.05, generally used as the significance level, by a wide margin, and the alternative hypothesis is rejected. In other words, no significant difference among the projects. In addition, no significant structual features in source code (e.g., wrong variable reference, wrong if statement) in higher ranked faults are found. Since the execution coverages don't have context of the source code, such features seem to hardly give a feature to execution coverages.

As for the conditions for the proposeed method, there are several points to be considered. First, on the premise that The

Table 3: P-Value of Wilcoxon Signed-Rank Test in Each Combination

Combination	One-Tailed(Left)	Two-Tailed	One-Tailed(Right)
One-Hot vs Block	1	1.711E-5	8.556E-6
One-Hot with bootstrapping sampling vs Block with bootstrapping sampling	9.91E-1	1.881E-2	9.403E-3
One-Hot vs One-Hot with bootstrapping sampling	9.923E-1	1.607E-2	8.037E-3
Block vs Block with bootstrapping sampling	9.537E-1	9.536E-2	4.768E-2

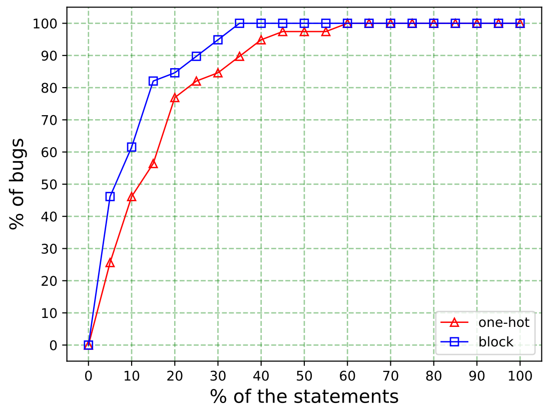


Figure 5: One-Hot Versus Block

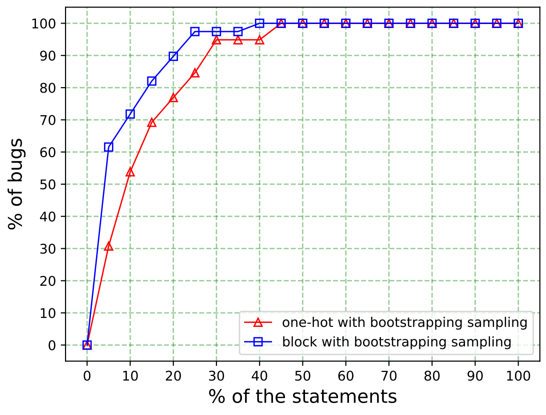


Figure 6: One-Hot with Bootstrapping Sampling Versus Block with Bootstrapping Sampling

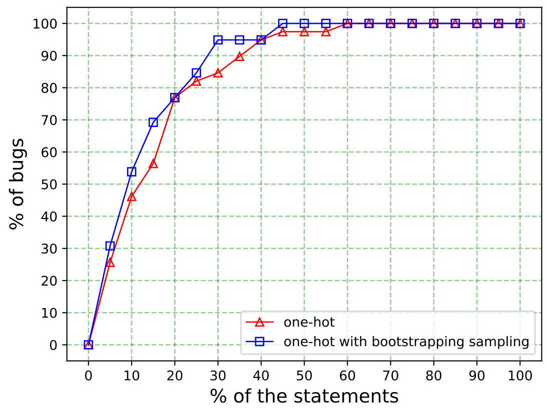


Figure 7: One-Hot Versus One-Hot with Bootstrapping Sampling

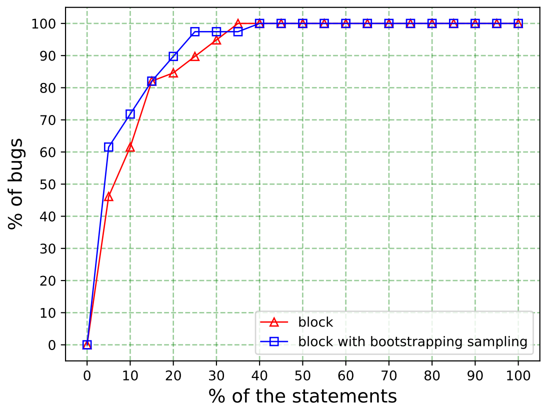


Figure 8: Block Versus Block with Bootstrapping Sampling

execution coverage is line-by-line in the proposed method, projects written in Java is used in the evaluation experiment. It is considered that proposed method can also work in a project written in a programming language which can measure execution coverages by line. Second, a test suite has to be composed with sufficient test cases. The proposed method extracts suspiciousness scores from a trained model. The model cannot learn well from insufficient test cases. Hence, a test suite which has adequate test cases is essential for the model training. Finally, white-box testing may be appropriate test for the proposed in terms of a granularity of localizing faults. The block segmentation in the proposed method divides lines whose execution are same into same blocks. A test suit that branch coverage is 100% makes the block segmentation minimum granularity, and the segmentation is similar to the code block. Therefore, testing methods such as white-box testing, which can cover branches in the source code, allow us to localize faults at smaller granularity.

In the OSS projects on the evaluation experiment, branch coverages are 62.33%, 82.72%, and 88.57% respectively correspond to Chart, Math, and Lang. Branch coverages of Math and Lang exceed 80%, while 62.33% of Chart is not desirable as branch coverage. For test suites whose branch coverage is not desirable, test generation techniques based on static code analysis can emphasise the branch coverage.

The proposed method gives each block suspiciousness scores. Thus, suspiciousness scores of lines in the same block are not different to each line. This means the proposed method cannot distinguish lines that is same pattern in the coverage. This is similar to other SBFL technique, which calculates the probability of bugs based on the coverage. Suspiciousness scores of lines that is identically executed tend to hardly differ each other. This results in a problem that impacts on bugs for such lines is determined equally. Therefore, it is necessary to distinguish such lines that implicit information that doesn't appear in execution coverage such as AST of source code and extracted semantics from documents, etc.

6 CONCLUSION

This paper proposes a Spectrum-Based Fault Localization method with deep learning technique, and the proposed method utilizes a virtual coverage to calculate suspiciousness scores and bootstrapping sampling. The proposed method trains a neural network model that predicts test results based on execution coverages, and measures the impact of each line on bugs from the trained model as suspiciousness scores, and provides a ranking based on the suspiciousness scores. We introduce a block segmentation for the virtual coverage. The block contains consecutive lines whose execution is the same in any execution coverage. In the training model, the bootstrapping sampling generates minibatches that balanced minority data and majority data.

In the evaluation experiments, we obtained results that shows the methods using virtual coverage, the block segmentation, and bootstrapping sampling ranked buggy lines higher rank. The authors concluded that the block segmentation and bootstrapping sampling effectively work in fault localization.

In future work, we are going to utilize implicit information that doesn't appear in execution coverage. The proposed method cannot distinguish lines that is same pattern in the coverage. This is because a execution coverage is a form which dispose of context of source code. Hence, the method have to make such lines different by implicit semantics of source code.

ACKNOWLEDGEMENT

Part of this work is supported by fund from Mitsubishi Electric Corp.

The research is also being partially conducted as Grant-in-Aid for Scientific Research A (18H04094) and C (21K11826).

REFERENCES

[1] C. Le Goues, M. Dewey-Vogt, S. Forrest, and W. Weimer, "A Systematic Study of Automated Program Repair: Fixing 55 out of 105 Bugs for $8 Each," ICSE, pp.3-13 (2012).

[2] J. Nam, S. Wang, Y. Xi, and L. Tan, "A bug finder refined by a large set of open-source projects," Information and Software Technology, Vol.112, pp.164–175 (2019).

[3] S. Kim, T. Zimmermann, K. Pan, and E. James Jr. Whitehead, "Automatic Identification of Bug-Introducing Changes," 21st IEEE/ACM International Conference on Automated Software Engineering (ASE'06), pp.81-90 (2006).

[4] S. Tsakiltsidis, A. Miranskyy, and E. Mazzawi, "Towards Automated Performance Bug Identification in Python," 2016 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW), pp.132-139 (2016).

[5] K. Matsushita, M. Matsumoto, K. Ohno, T. Sasaki, T. Kondo, and H. Nakashima, "A Debugging Method Based on Comparison of Execution Trace," Symposium on Advanced Computing Systems and Infrastructures (SACSIS), Vol.2011, pp.152-159 (2011) (in Japanese).

[6] A. Elena, B. Alexander, D. Artem, K. Konstantin, K. Anton, M. Ilya, and M. Vladimir, "A Survey on Software Defect Prediction Using Deep Learning," Mathematics, Vol.9, No.11, 1180 (2021).

[7] T. Ikeda, K. Okano, S. Ogata, and S. Nakajima, "Localization of Fault Methods and Ablation of Execution Traces Using A Machine Learning Model to Classify Test Results," IEICE Technical Report, Vol.121, No.416, pp.13-18 (2022) (in Japanese).

[8] D. Guo, S. Ren, S. Lu, Z. Feng, D. Tang, S. Liu, L. Zhou, N. Duan, A. Svyatkovskiy, S. Fu, M. Tufano, S. Kun Deng, C. Clement, D. Drain, N. Sundaresan, J. Yin, D. Jiang, and M. Zhou, " GraphCodeBERT: Pre-training Code Representations with Data Flow," arXiv:cs.SE/2009.08366. (2021).

[9] J. Devlin, M. Chang, K. Lee, and K. Toutanova, " BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," Proceedings of the 2019 Conference of the North American Chapter of the Association

for Computational Linguistics: Human Language Technologies, Vol.1, pp.4141-4186 (2019).

[10] Z. Zhang, Y. Lei, X. Mao, M. Yan, L. Xu, and X. Zhang, "A study of effectiveness of deep learning in locating real faults," Information and Software Technology, Vol.131, No.1, pp.1–16 (2021).

[11] W. Eric Wong, V. Debroy, R. Golden, X. Xu, and B. Thuraisingham, "Effective Software Fault Localization Using an RBF Neural Network," in IEEE Transactions on Reliability, Vol.61, No.1, pp.149-169 (2012).

[12] Z. Zhang, Y. Lei, X. Mao, M. Yan, L. Xu, and J. Wen, "Improving deep-learning-based fault localization with resampling," Journal of Software: Evolution and Process, Vol.33, No.3 (2021).

[13] R. Abreu, P. Zoeteweij, and A. J. C. van Gemund, "On the accuracy of spectrum-based fault localization," Testing: Academic and Industrial Conference Practice and Research Techniques, pp.89-98 (2007).

[14] W. Eric Wong, V. Debroy, Y. Li, and R. Gao, "The DStar method for effective software fault localization," IEEE Transactions on Reliability, Vol.63, No.1, pp.290-308 (2014).

[15] Y. Yan, M. Chen, M. Shyu, and S. Chen, "Deep Learning for Imbalanced Multimedia Data Classification," 2015 IEEE International Symposium on Multimedia (ISM), pp.483-488 (2015).

[16] R. Just, D. Jalali, and M. D. Ernst, "Defects4J: A database of existing faults to enable controlled testing studies for java programs," in Proceedings of the 2014 International Symposium on Software Testing and Analysis, pp.437–440 (2014).

(Received: November 6, 2022)
(Accepted: April 5, 2023)

Hitoshi Kiryu is a graduate student of Shinshu University. His areas of interest include formal verification.

Nobutoshi Todoroki received his M.E. degrees in Information and Computer Sciences from Osaka University in 2001. He joined Mitsubishi Electric Corp. Currently he is a senior manager of Solution Engineering Dept. at Advanced Technology R&D Center. He is also a member of IPSJ.

Satoshi Suda received his M.S. degree in mathematics from Osaka University, Osaka, Japan, in 2016. He joined Mitsubishi Electric Corp.Currently he is a researcher of Solution Engineering Dept. at Advanced Technology R&D Center.

Shinpei Ogata is an Associate Professor at Shinshu University, Japan. He received his BE, ME, and PhD from Shibaura Institute of Technology in 2007, 2009, and 2012 respectively. From 2012 to 2020, he was an Assistant Professor, and since 2020, he has been an Associate Professor, in Shinshu University. He is a member of IEEE, ACM, IEICE, IPSJ, and JSSST. His current research interests include model-driven engineering for information system development.

Kozo Okano received his BE, ME, and PhD degrees in Information and Computer Sciences from Osaka University in 1990, 1992, and 1995, respectively. He was an Assistant Professor and an Associate Professor of Osaka University. In 2002 and 2003, he was a visiting researcher at the Department of Computer Science of the University of Kent in Canterbury, and a visiting lecturer at the School of Computer Science of the University of Birmingham, respectively. Since 2020, he has been a Professor at the Department of Electrical and Computer Engineering, Shinshu University. Since 2023, he has been the Director of Center for Data Science and Artificial Intelligence. His current research interests include formal methods for software and information system design, and applying deep learning to Software Engineering. He is a member of IEEE, IEICE, and IPSJ.

Regular Paper

An Analysis of Rescue Requests on Twitter in a Disaster

Yuki Koizumi[†], Junji Takemasa[†], Toru Hasegawa[†], and Yoshinobu Kawabe[‡]

[†]Osaka University, Japan
[‡]Aichi Institute of Technology, Japan
{ykoizumi, j-takemasa, t-hasegawa}@ist.osaka-u.ac.jp, kawabe@aitech.ac.jp

Abstract - During catastrophic disasters like the Japan floods 2018, phone-based emergency call systems may not work as expected due to heavy congestion and network disruption. People needing help use social media, e.g., Twitter and Facebook, for delivering rescue requests, which complements phone-based emergency call systems, in disasters. Machine learning is a promising approach to automated rescue request extraction from a vast amount of social media posts. Since it is a critical task, such classifiers should produce few false negatives (rescue requests identified as non rescue requests). The objective of the study is to learn lessons to develop better classifiers for rescue request extraction. Hence, we analyze rescue-related tweets, which are tweets containing rescue-related keywords like "rescue," conduct a classification experiment of rescue requests using a classifier based on the bidirectional encoder representations from transformers (BERT) model, and investigate the classification results, particularly focusing on why the classifier produce false negatives. Furthermore, we construct an annotation mechanism based on a recurrent neural network to understand why the classifiers produce false negatives.

Keywords: Social Media, Twitter, Disaster, Analysis, Machine Learning

1 INTRODUCTION

Delivering rescue requests from citizens in need of help to the right persons, such as rescue authorities and first responders, is a key to effective disaster management. Phone-based emergency call services, however, may not work as expected during and after catastrophic disasters because of network disruption and congestion [1].

Circumstances of rescue requests during disasters have been changing. Some citizens needing help use social media, e.g., Twitter and Facebook, for delivering rescue requests, especially when phone-based emergency call services are inadequate, and social media complements existing phone-based emergency call services [1], [2]. For instance, in the U.S., several studies reported the use of social media during Hurricane Harvey. Stelter [2] reported that hundreds of stranded Texas residents sought help by posting on Facebook and Twitter during Hurricane Harvey, and Facebook and Twitter were clearly used as a supplement for traditional emergency services. Jahanian et al. [1] also reported that people tweeted their addresses for seeking rescues during hurricane Harvey, especially, when they felt that traditional aid-seeking methods was not adequate. Japan, for instance, had several catastrophic floods in 2018 and 2019. One is the *heavy rain of July 2018*, also referred to as *Japan floods 2018* [3] and another is the *19th typhoon of 2019* [4]. We observed that many rescue requests were posted on Twitter during both the two disasters.

Few rescue requests on social media, nevertheless, contributed to actual rescue activities. Though our analysis, discussed in Section 3, reveals that 312 rescue requests were posted during the Japan floods 2018, we did not find any news that reported any of the rescue requests directly contributed to rescue activities. As another example, a local government, Nagano Prefecture, deployed several workers to capture rescue requests from Twitter [5] during the 19th typhoon. On the one hand this activity finally contributed to saving about 50 victims, but on the other hand it consumed many workers of the local government, who might have contributed to other tasks. These tales imply that it is indispensable to extract rescue requests on social media automatically to utilize them for rescue activities. Machine learning is a promising technique for filtering rescue requests on social media [6]–[8].

Rescue-related social media posts, which are defined as tweets with rescue-related keywords, such as rescue, are a mixture of good and bad; that is, rescue-related tweets contain non rescue requests as well as rescue requests, as described in Section 3. Understanding actual rescue-related tweets is a key to realizing good rescue request classifiers. Hereafter, we refer to social media posts as tweets since this study focuses on Twitter as social media. Several studies analyzed rescue requests on Twitter, especially focusing on tweets during flood disasters in Japan [9], [10]. Sato and Imamura [9] analyzed rescue requests found on Twitter during two flood disasters in Japan and claimed that tweets tagged with #rescue include many non-rescue requests. Song and Fujishiro [10] interviewed a news article writer and a member of the social media listening team of a Japanese television company about rescue requests observed during flood disasters in Japan. In addition to the effort, we in this study analyze rescue requests on Twitter from the perspective of classification with machine learning.

The main contribution of this paper is to draw lessons for developing machine learning based classifiers identifying rescue requests through analyses on actual tweets during flooding disasters. One of the severe requirements for automated rescue request classification is reducing (ideally minimizing) false negatives, which are rescue requests identified as non rescue requests, because missing rescue requests is a matter of life and death for citizens needing help. Although much effort has been devoted to machine learning based tweet classification (not rescue requests) [6]–[8], few studies focus on reasons for misclassification, including false negatives and even false positives. Kshirsagar et al. [8], for example, tried to detect states of crisis, such as suicide, self-harm, abuse, or eating disorders, using machine learning. However, they focused on identifying what phrases contribute to classification results using an attention mechanism. Similarly, Ma et al. [6] and Ruchansky et al. [7] tried to classify fake news on Twitter. However, they focused on selecting appropriate features fed to machine learning to improve classification accuracy. In contrast to the existing studies, we analyze classification results, aiming at understanding reasons for misclassification, particularly false negatives.

We in this study conduct analyses on actual tweets in the following steps: First, we captured tweets that include disaster-related keywords (e.g., flooding and rescue, as well as the names of disaster-affected areas) on Twitter during several recent flooding disasters in Japan. Second, we analyze the captured tweets similarly to the work in [9] to understand how rescue-related tweets evolve during a disaster. Third, we analyze rescue-related tweets to understand what kinds of tweets are included in rescue-related tweets. In this step, we categorize rescue-related tweets into several categories, such as rescue requests, disaster information, and sympathy. Finally, we conduct experiments classifying rescue requests from numerous tweets using machine learning. One of the essential lessons learned from the experiment is that the variety of textual contexts of rescue-related tweets is a reason for producing false negatives. The analysis in the third step helps investigate this lesson.

This paper is extended from its conference version [11] from the following two aspects: First, we have updated the analysis based on machine learning by using the state-of-the-art machine learning model, i.e., the bidirectional encoder representations from transformers (BERT) model [12]. Second, we provide more detailed analysis results and observations with introducing actual tweets in disasters.

The paper is organized as follows: We first analyze disaster-related tweets in Section 2 and rescue-related tweets in Section 3. Based on the observations found in the analyses, we build a classifier to extract rescue requests from Twitter in Section 4. Section 5 briefly summarizes related work and Section 6 finally concludes this paper.

2 ANALYSIS ON DISASTER-RELATED TWEETS

2.1 Objective and Overview of Analysis

This section analyzes disaster-related tweets captured during the recent floods in Japan. The objective of the analysis in this section is to understand tweets mentioning a catastrophic disaster, especially during and in the aftermath of the disaster. More precisely, we investigate how many tweets mentioning the disaster exist, how the number of such tweets evolves, and when citizens post tweets having rescue-related keywords. We should note that though our analysis method is similar to that in [9], our dataset is different from that in [9], i.e., our dataset contains both rescue-related tweets and disaster information (non rescue requests) unlike the dataset in [9] that only includes tweets tagged with #rescue. Thus, this section's results will add value to the existing studies. More precisely, the results in this section help us understand statistics and time-series information about disaster-related tweets and rescue-related tweets.

2.2 Dataset

We collected tweets from three flood disasters in Japan: the Japan floods 2018 [3], the 15th [13], and the 19th typhoons [4] of 2019 in Japan. We use the tweets from the Japan floods 2018 for the analyses in this section and those from the other two disasters for the classification in the next section.

The tweets were captured via the Twitter search API [14] by specifying disaster-related keywords, which were selected so that we were able to capture as many disaster-related tweets as possible. Tweets retweeted by means of the twitter official API were eliminated from the dataset. Let us note that all the keywords are Japanese, and therefore all the tweets are also written in Japanese. The keywords are categorized into five classes: rain disaster, rescue request, first responder, volunteer, and infrastructure, as summarized in Table 1. We refer to tweets containing at least one keyword in any of the classes as *disaster-related tweets*. In the same way, tweets containing keywords in the classes of rescue request, first responder, infrastructure, and volunteer are referred to as *rescue-*, *first responder-*, *infrastructure-*, and *volunteer-related tweets*, respectively. Since most disaster-related tweets contain keywords in the rain disaster class, we do not focus on tweets in this class. Each class may have tweets unrelated to the class because they were captured with keyword search. For instance, tweets that are not rescue requests, although having rescue-related keywords, are categorized into rescue-related tweets.

The number of collected tweets, i.e., disaster-related tweets, during and in the aftermath of the Japan floods 2018 is 6,978,389, and the number of tweets in each class is summarized in Table 2. As we discuss later in Section 4, we captured 246,807 rescue-related tweets.

2.3 Temporal Analysis

We investigate how the number of tweets grows in accordance with situations in a flood disaster. Before explaining the result, we briefly explain the Japan floods 2018 to understand this analysis. From the end of June to the middle of July 2018, it had been raining heavily and steadily in western Japan. As a result, it caused widespread and catastrophic floods throughout western Japan, especially in Okayama and Hiroshima. From July 5th to 7th, the most severe floods occurred in Okayama and Hiroshima.

The time series of the number of disaster-related tweets is plotted in Fig. 1a. The horizontal and vertical axes represent the date and the number of tweets in each day. The number of disaster-related tweets steeply increases around July 5th as floods became severe, and its peak is on July 6th and 7th, the most intense days of floods. The numbers of tweets on July 6th and 7th are 591,514 and 584,692, respectively. Note that

Table 1: Search keywords used for collecting disaster-related tweets

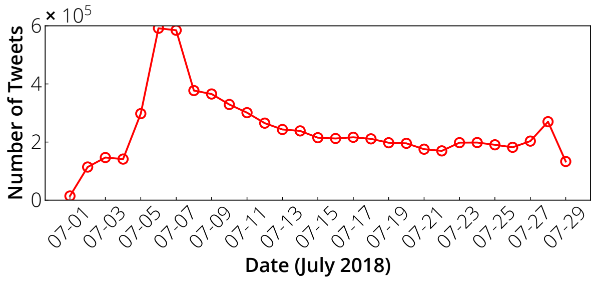
Category	Keywords
Rain disaster	Heavy rain, rain disaster, disaster, flood, flood disaster, disaster-hit area, flood-hit area, river burst, evacuation, and (confirmation of someone's) safety
Rescue request	Rescue, rescue request, SOS, and help (me)
First responder	Rescue team, fire fighting team, police, Japan self-defense forces, hospital, and local government
Infrastructure	Infrastructure, lifeline, water supply, electricity supply, gas supply, (network) disconnection, (network) congestion, (network) failure, and recovery
Volunteer	Volunteer, support, and relief supply

Table 2: The number of tweets captured during and in the aftermath of the Japan floods 2018 (July 1–29, 2018)

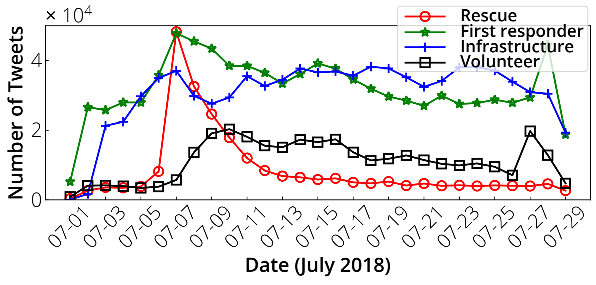
Category	Number
Total (disaster-related tweets)	6,978,389
Rescue-related tweets	246,807
First responder-related tweets	932,605
infrastructure-related tweets	889,889
Volunteer-related tweets	324,935

the number of disaster-related tweets again increases at the end of July because another typhoon came to Japan.

Figure 1b indicates the number of rescue-, first responder-, infrastructure-, and volunteer-related tweets. Though rescue-related tweets also increased as floods became severe, the trend is shifted slightly behind compared to that of disaster-related tweets. That is, the number of rescue-related tweets suddenly increases on July 7th, and the peak is also on the same day. We captured 48,272 rescue-related tweets on July 7th. First responder and infrastructure-related tweets were captured almost invariably throughout the disaster. In contrast, the number of volunteer-related tweets increases in the aftermath of the disaster.



(a) Number of disaster-related tweets



(b) Number of rescue, first responder, infrastructure, and volunteer-related tweets

Figure 1: Time series analysis of the number of tweets

3 ANALYSIS ON RESCUE-RELATED TWEETS

3.1 Objective and Overview of Analysis

This section explains observations obtained through reading tweets containing rescue-related keywords. The primary objective of the analysis is to understand what kinds of tweets are included in rescue-related tweets. We categorize rescue-related tweets into eight categories. In the classification experiment in Section 4, we will reveal that a mixture of texts in different contexts is one of the causes of misclassification in rescue request extraction. The categorization is indispensable for drawing such observations. Furthermore, we observe that the locations where real rescue requests in our dataset were posted were concentrated in a narrow area. The observation is also essential for drawing a lesson discussed in the next section.

3.2 Dataset

Since it is nearly impossible to read an enormous number of tweets containing rescue-related keywords (246,807, as shown in Table 2), we further sampled 5,304 tweets from the rescue-related tweets using hashtags. Specifically, we selected tweets that contains hashtags related with rescue requests (i.e., *#rescue*, *#rescue request*, *#SOS*, *#help*, and *#help_me*) from the rescue-related tweets. The reasons for filtering tweets with the hashtags are twofold. One is that Twitter Japan (@TwitterLifeline) recommended using the #rescue hashtag [15]. The other is that many of the actual rescue requests eventually had the hashtags since other Twitter users re-posted the tweets, especially if they did not contain the hashtags, by adding the hashtags.

3.3 Taxonomy

We found that a keyword search with rescue-related keywords is not a good approach for extracting rescue requests because it results in a mixture of rescue and non-rescue requests.

Therefore, as the first step of extracting rescue requests from Twitter, we need to understand what kinds of tweets are included in rescue-related tweets. We read all the sampled tweets to achieve the objective. As a byproduct of reading the tweets, we develop a taxonomy of disaster-related tweets. This section explains the taxonomy categories, introduces actual disaster-related tweets categorized according to the taxonomy, and discusses observations.

3.3.1 Categories

This section overviews the taxonomy categories, which are summarized in Table 3. Alam et al. [16] have similarly developed a taxonomy about tweets captured during disasters. While their taxonomy categorizes disaster-related tweets, our taxonomy focuses only on rescue-related tweets and analyzes them in detail. As shown in Table 3, rescue-related tweets are categorized into eight categories. We explain the criteria for categorizing tweets in the rest of this section, and we introduce several examples and discuss observations in the subsequent sections.

The rescue request category is based on a recommendation about rescue requests on Twitter, which Twitter Japan (@TwitterLifeline) posted [15]. The recommendation suggests that victims post a rescue request indicating the detailed current situation and their postal address (or their geographical location) as well as the hashtag #rescue. Hence, we define *rescue requests* as tweets containing the following information:

1. Help message: A message representing that the persons posting the tweet themselves are victims and are in need of rescue.
2. Disaster situation: A message explaining the situation of the victims.
3. Location: The location of the victims.

Based on the definition, we next define *incomplete rescue requests* as tweets containing a help message but lacking either a location or a disaster situation (or both). In contrast, *disaster situations* are defined as tweets containing disaster situations but not containing a help message. The fundamental idea behind these definitions is that people posting tweets in the incomplete rescue request category are likely to be facing a dangerous situation, whereas people posting tweets in the disaster situation category themselves do not face a dangerous situation. Nonetheless, tweets in the disaster situation category are helpful in knowing the disaster.

The categories of *sympathy*, *advice*, and *volunteer* are straightforwardly defined as the definitions mentioned above. In contrast, tweets in the categories of *exploitation* and *unrelated* are similar because tweets in both categories are unrelated to the disaster. A difference lies in the fact that rescue-related keywords used in tweets in the exploitation categories are intentionally used. Tweets in the exploitation category, for instance, include a political opinion and a service advertisement and intentionally put #rescue tag so that the tweets appear on the trend line. In contrast, tweets in the unrelated category use rescue-related keywords in non-disaster situations, such as smartphone games, books, movies, and TV shows.

3.3.2 Actual Tweets

This section introduces examples of actual tweets classified into each category. Note that all the tweet examples in this paper were originally written in Japanese, but we translated them into English. In addition, privacy information, such as complete postal addresses, building names, and victim names, is concealed, which is indicated by the dash sign (—).

Rescue Request: A typical example of rescue requests is as follows:

> #rescue_request URGENT! Please rescue us, **— (full postal address), — (name of this victim)**. My elderly father, mother, and I are isolated in our house. The first floor of the house is fully flooded, and we have taken refuge in a closet on the second floor. Please help us!

The tweet includes all the three kinds of information (a help message, a disaster situation, and a location).

Incomplete Rescue Request: The following tweets are categorized into incomplete rescue requests.

1. Please help me! **— (full postal address)**
2. #rescue The mountain is likely to collapse, and we are stuck. #Hiroshima

Although tweet #1 contains the exact location of this victim, the victim's situation is still unknown. Similarly, rescue teams need to know the exact location of the victim in tweet #2. In this way, tweets in this category lack any of the three kinds of information, and thus rescue teams (or local government officers) need to communicate with victims to collect the missing information.

Disaster Situation: Tweets in the disaster situation category do not contain a help message though the tweets contain somewhat helpful information for disaster management. Since tweets do not contain a help message, posters of the tweets themselves or acquaintances of the posters, such as their family, relatives, colleagues, and friends, are not likely to be in need of rescue. Another situation is that tweets describe general people rather than specific people even though they may face a dangerous situation. The following tweet is an example of tweets in this category.

> None of the news media have been reporting anything about Ehime. Rescue in mountainous areas has been delayed. People there are almost isolated. They are sending out lots of messages and requesting rescue. Please help them. #rescue #Ehime #— (town name)

This is a borderline tweet between the incomplete rescue request and disaster situation categories. The reason why we categorize it into the disaster situation category is that it mentions that some people are requesting rescue but they seem to be general people in disaster-stricken areas.

Sympathy: Posters of tweets in the sympathy category express in their tweets that they are praying for the safe rescue of the victims. Though such tweets have rescue-related keywords, they seldom contain information helpful for disaster management. A typical example of tweets in this category is as follows:

Table 3: A taxonomy of rescue-related tweets

Category	Tweets	Ratio
Rescue request	Tweets indicating all the following information a rescue request, situations of victims, and locations of victims.	24.5%
Incomplete rescue request	Tweets indicating a rescue request but lacking either the situation and the location of victims (or both).	8.7%
Disaster situation	Tweets reporting a situation of the disaster.	7%
Sympathy	Tweets praying for the safe rescue of victims.	25.9%
Advice	Tweets giving advice to victims or adding supplementary information to existing tweets.	20.0%
Volunteer	Tweets offering or requesting voluntary contributions, e.g., voluntary work and donations of supplies like foods, water, and clothes.	1.2%
Exploitation	Tweets mentioning something unrelated to rescue requests by intentionally exploiting rescue-related keywords so that the tweets are widespread.	3.5%
Unrelated	Tweets that is not related to the disaster while it contains rescue-related keywords, which are used in a different context from rescue requests, e.g., tweets regarding a smartphone game application.	9.5%

> A town I have visited several times faces a catastrophic situation. Please rescue people there. #— (town name)

Advice: Tweets in the advice category offer supplementary information, particularly for victims requesting rescue. Since such tweets are for victims needing rescue, they contain rescue-related keywords. However, they seldom offer helpful information for disaster management. An example of tweets in this category is as follows:

> People who are waiting for rescue should indicate something conspicuous so that rescue teams easily find you.

Exploitation: An example of tweets in the exploitation category is as follows:

> @— (politician account name) Please search Twitter for the hashtags "#rescue" instead of having a dinner party, and you will know the current situation that would be more critical than your party.

Tweets in this category tend to exploit rescue-related hashtags, particularly for satisfying their own purposes. For example, a typical type of tweet expresses criticisms of politicians like the example above. Another type of tweet exploits rescue-related hashtags to satisfy the posters' desire, such as advertising their services, increasing the number of followers, and disseminating their tweets.

Unrelated: Other tweets are categorized into the unrelated category. An example in this category is as follows:

> Please help me with assignments in my university lectures. #SOS

Tweets in this category use rescue-related keywords in a context different from disaster management, such as daily life, smartphone games, TV shows, and movies.

3.3.3 Observations

The taxonomy implies that extracting rescue requests via keyword searches is challenging since many tweets are unrelated to rescue requests while containing rescue-related keywords. As shown in Table 3, more than 65% of tweets in our dataset are not rescue requests though they contains rescue-related keywords.

Another observation is that many rescue requests and incomplete rescue requests were describing the same victims and the same incidents. We discovered 312 original rescue requests in the tweet dataset, whereas more tweets are categorized into rescue requests. The reason is that voluntary citizens re-posted such an incomplete rescue request by adding missing information, e.g., the hashtag #rescue.

3.4 Spatial Analysis of Rescue Requests

We further analyze the 312 original rescue requests, focusing on the locations where they were posted. Because few tweets have geographical metadata like geotag information, we extract a postal address in the text field of the rescue requests. We put pins pointing the extracted postal addresses on the map in Fig. 2.

A crucial observation is that rescue requests were concentrated in severely flood-hit areas, which were very narrow, in the case of the Japan floods 2018. Although we present a coarse-grained map in Fig. 2 to protect the privacy of victims, the 312 rescue requests contain the exact location of the victims. For instance, there were 235 rescue requests indicating a postal address within a 7×5 km rectangle area in a certain town in Okayama.

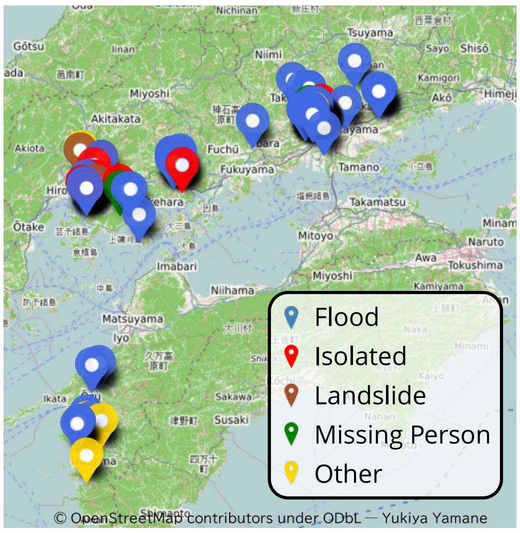


Figure 2: A map pointing rescue requests during the Japan floods 2018

4 CLASSIFICATION BASED ON MACHINE LEARNING

4.1 Objective and Overview of Classification Experiment

Rescue request classification is a critical task that directly impacts human lives, and hence reducing (ideally minimizing) false negatives, which are rescue requests classified into non rescue requests, is one of the severe challenges compared to the classification of other types of tweets. Thus, we focus on understanding why classifiers produce false negatives, which contributes to developing classifiers that produce fewer false negatives.

Note that the objective of the analysis is neither to develop a new machine learning technique nor to review the feasibility of state-of-the-art machine learning techniques for rescue request classification. Instead, we focus on learning lessons for extracting rescue requests from Twitter using machine learning. Thus, we build our classifiers using an existing machine learning model for natural language processing and analyze why the classifier identifies rescue requests as non rescue requests in two ways. First, we manually read misclassified tweets. Second, we use the attention mechanism of neural network models in the same way as Kshirsagar et al. [8], which contributes to understanding the reasons for the misclassification. The lessons drawn through the experiments are summarized in Section 4.6.

4.2 Dataset

We use the same dataset as used in Section 2. We use tweets from the Japan floods 2018 and the 19th typhoon for training and testing because the two disasters caused more severe damage than recent other typhoons in Japan. For training and testing, we further extract tweets containing rescue-related hashtags, i.e., #rescue, #rescue request, #SOS, #help, and #help_me. We should note that all the hashtags are specified in Japanese and they are translated into English in this paper. We finally selected 3424 and 1501 rescue-related tweets from the Japan floods 2018 and the 19th typhoon dataset, respectively.

Each tweet t is labeled by z_t, where 1 indicates t is a rescue request and 0 otherwise. One of the authors gives labels to tweets, and the labels are verified by two of the other authors. There are 328 and 74 rescue requests in the the Japan floods 2018 and the 19th typhoon dataset, respectively. The datasets are summarized in Table 4.

4.3 Model of Classifier

We use two classifiers in this study, and the two classifiers are based on a neural network: One is the bidirectional encoder representations from transformers (BERT) model [12] and the other is a recurrent neural network consisting of gated recurrent unit (GRU) cells. Hereafter, we refer to the recurrent neural based on GRU cells as the GRU model. We use the BERT model because it is a state-of-the-art neural network model for natural language processing. We use the GRU model because we adopt the same analysis method as used by Kshirsagar et al. [8].

We use the BERT Japanese pre-trained model developed by National Institute of Information and Communications Technology, Japan [17]. The BERT model is pre-trained using Japanese Wikipedia pages, and the number of its vocabulary is 100 thousand. The BERT model is fine-tuned using the Japan floods 2018 dataset.

We build another classifier using the GRU model to analyze why machine learning misclassifies rescue requests. Specifically, we adopt the same analysis method as Kshirsagar et al. [8], where they use an attention mechanism to analyze what parts of a tweet contribute to the classification result. They analyzed hidden states produced by GRU cells to investigate what phrases contribute to the classification results. Hidden states are referred to as attention values, hereafter. A high attention value represents that the corresponding phrase (or word) is related to a rescue request. In the same way, we use the attention values of the BERT classifier as well to analyze why machine learning misclassifies rescue requests.

4.4 Classification Results

We use accuracy, precision, recall, and F-measure as performance metrics. In our case, a true positive (negative) corresponds to a (non-)rescue request that is (not) identified as a rescue request. A false negative corresponds to a rescue request that is not identified as a rescue request and a false positive corresponds to a non-rescue request that is identified as a rescue request. The most critical metric for classifiers of rescue requests is recall, which represents the ratio of tweets identified as rescue requests among tweets that are actual rescue requests. A low recall value means that many rescue requests will be missed, i.e., many false negatives. Since rescue requests must not be missed, the recall must be high for classifiers of rescue requests.

We first investigate the performance of the classifier by using the same dataset. Specifically, we use 80%, 10%, and 10% of the tweets in the Japan floods 2018 dataset for fine-tuning, validating, and testing purposes, respectively. Table 5 summarizes the classification results and Table 6 shows the

Table 4: Summary of datasets used for classification analysis

Dataset	Number of tweets	Number of rescue requests
Japan floods 2018	3424	328
19th typhoon	1501	74

Table 5: Classification result

Predict\Label	1	0
1	34	6
0	17	321

Table 6: Performance metrics

Accuracy	Precision	Recall	F-measure
0.94	0.85	0.67	0.75

Table 7: Classification result

Predict\Label	1	0
1	45	26
0	28	1402

Table 8: Performance metrics

Accuracy	Precision	Recall	F-measure
0.96	0.63	0.62	0.63

four performance metrics. Although the accuracy is very high, the recall is low, as shown in Table 6. This result implies that the classifier misses several rescue requests.

Next, we investigate the applicability of the trained classifier to other disaster situations. We use the 19th typhoon dataset for testing, while we use the same classifier as the previous evaluation, which is fine-tuned using the Japan floods 2018 dataset. The classification results and the performance metrics are summarized in Table 7 and Table 8, respectively. The recall value is further decreased compared to the classification results in Table 5.

4.5 Analysis based on Attention Values

To understand the reasons why the recall is low, we investigate attention values of these classification results. Attention values indicate where the classifier focuses to filter (non-)rescue requests. Analyzing attention values, we obtain four observations regarding the misclassification: The first observation is that words of high attention values for non-rescue requests is one of the causes of false negatives. The second observation involves that similar expressions are used in both rescue requests and tweets of in the three categories, disaster situation, sympathy, advice and supplement. The third one involves that many citizens retweeted rescue requests by adding text, which is likely to belong to the three categories. Finally, the last observation involves that location names cause misclassification. The rest of this section explains the observations. We should again note that all tweets analyzed in this paper are written in Japanese and they are translated in English for this manuscript.

The results regarding the first observation are shown in Table 9, which summarizes the words with high attention values for the four classification results. The words are selected in the following steps. First, we selected words of the top five highest attention values from each tweet. Second, we counted the number of appearances of the selected words. Finally, we sorted the words in descending order of the number of appearances and selected frequently used words. Note that we eliminated postpositional particles, and privacy information, such as postal addresses, commas, and periods from the table. Both tweets classified as rescue requests and those not classified as rescue requests have words indicating postal addresses, such as -town and -city. However, the hashtag mark # is more frequently used in tweets classified as non-rescue requests regardless of true and false negatives than tweets classified as rescue requests. We observed that many Twitter users quote actual rescue requests and add some information to the quoted tweets, which often contains the hashtag mark and the mention mark. Hence, rescue requests are misclassified as non-rescue requests if the poster unintentionally use the hashtag mark and the mention mark.

Regarding the second observation, for example, some rescue requests use guessing expressions, which are often used in tweets reporting a disaster situation. The following tweet is a typical example of such rescue requests:

> Help me. My house is flooded above the 2nd floor level. Nobody comes to my rescue. *Roads appear to be blocked due to landslide.*

The last sentence is similar to tweets reporting disaster situations. Such rescue requests tend to be misclassified.

The third observation comes from the fact that there were incomplete rescue requests during the Japan floods 2018, as discussed in the previous section. Many voluntary citizens added several pieces of information, such as hashtags, to incomplete rescue requests to make them complete and retweeted them. On retweeting rescue requests, the citizens often add comments praying for the safety of the victims. The following tweet is a typical example:

> *I hope the victim will be immediately rescued.* #rescue RT: Help me. My house is flooded above the 2nd floor level.

Another example is as follows:

> *Add #rescue in case you need rescue.* RT: Help me. My house is flooded above the 2nd floor level.

Table 9: Examples of words of a high attention value

Classification Result	Words of a high attention value
True positive	town, -, floor, #, people, left (leave), flooded, water, city, isolated, -Chome (Japanese postal address)
False positive	town, #, -, floor, city, San (a Japanese title for general people, like Mr./Mrs.), -Chome
True negative	#, @, town, http, please, city, people, dissemination
False negative	#, evacuation, name, help, city

> This rescue request may not be delivered to rescue authorities unless you put hashtags like #rescue #SOS. RT: Help me. My house is flooded above the 2nd floor level.

Such tweets are misclassified into non-rescue requests since the added texts are similar to tweets in the category of either sympathy or advice and supplement.

The fourth observation is that tweets containing the names of disaster-hit locations tend to be misclassified into rescue requests.

> The situation of — *(town name)* — *(city name)* looks terrible.

During the Japan floods 2018, many rescue requests are generated in a town, which is one of the severely damaged areas, and therefore many rescue requests contain the name of the town and the city of the area.

Finally, we pick up typical phrases contributing to prediction of rescue requests. The following list summarizes phrases of a high attention value of rescue requests:

- Characteristics of victims like age and sex (e.g., elderly persons, babies, children, woman, man, octogenarian (80s), septuagenarian (70s), grandfather, and grandmother).
- Situations of flood-hit buildings (e.g., the water level has been rising gradually and my house is flooded above the 2nd floor level).
- Location names of flood-hit areas during the Japan floods 2018.
- Numbers in postal addresses (e.g., x-y (x and y represent a block and a house number and they are used like Mabi-town x-y in Japanese style postal addresses)).

In contrast, the attention values of phrases often used in rescue requests (e.g., rescue request, SOS, help, share it if you can, and retweet it if you can) are very low because they are also used in non-rescue requests. Furthermore, the attention values of location names of typhoon-hit areas during the 19th typhoon are also very low. If we replace a location name in a false negative rescue request of the 19th typhoon to that of a flood-hit area during the Japan floods 2018, they are identified as a rescue request.

4.6 Lessons Learned

This section summarizes four lessons learned through the classification experiments.

1. First, the entire text of rescue requests should not be used for training classifiers because a part of a rescue request is often unrelated to the rescue request but it is similar to tweets reporting disaster situations.

2. Second, retweeted rescue requests should be carefully handled in the same way as the first lesson because persons retweeting rescue requests often add several texts expressing sympathy for the victims. If texts added on retweeting are unrelated to rescue requests, they should be eliminated for training a classifier.

3. Third, the entire text of tweets should not be used for predicting whether they are rescue requests or not for the same reason behind the first and the second lesson. One way to realize this lesson is to classify tweets using attention values as well as a predicted result of classifiers. Using attention values allows us to identify texts related to rescue requests in a tweet and omit texts unrelated to the rescue requests, which cause misclassification, as demonstrated in the previous section.

4. Finally, location names should be handled independently for each disaster. One way to follow this lesson is to convert location names in tweets to a particular reserved word.

5 RELATED WORK

This section compares the present study with related studies.

Several studies analyzed disaster-related social media posts, especially focusing on Twitter. Alam et al. [16] analyzed disaster-related tweets in three hurricanes in the U.S. in 2017, i.e., Harvey, Irma, and Maria. They conducted both textual content analysis and multimedia content analysis on tweets from the three hurricanes. They define a taxonomy for disaster-related tweets and this study inspired us to categorize rescue-related tweets. Yang et al. [18] also analyzed tweets from the hurricane Harvey and proposed a framework to estimate the credibility of events reported by tweets. They also captured disaster-related tweets by searching Twitter for predefined keywords. While those studies focuse on disaster-related tweets, our study focuses on rescue-related tweets.

Next, we introduce studies that proposed classifiers for social media posts with machine learning. Studies in [6] and [7] propose classifiers based on neural networks for detecting fake information or rumors. Though both studies adopt recurrent neural networks, they develop slightly different models. Ma et al. [6] use intervals between social media posts reporting the same event as input values for their classifiers because of the fact that tweets are too short to identify their context with machine learning. Ruchansky et al. [6] propose to use relation between users who post tweets regarding the same event as well as texts of tweets to identify fake news. Kshirsagar et al. [8] propose a classifier for detecting crises like suicide, self-harm, abuse, or eating disorders by using a recurrent neural network. They also develop an annotation mechanism for explaining how social media posts are related to the crises. Their model uses texts of tweets as input values of their classifier. Our model is based on their model.

Finally, we introduce our previous study [19], which develops a communication framework for disaster management. The key idea behind the proposed framework is to utilize social media for collecting information regarding disaster situations. The framework delivers social media posts to right persons by using a machine learning technique. While the motivation of the previous study is to develop a communication framework that utilizes social media, the present study focuses on rescue requests in social media.

6 CONCLUSION

Circumstances of rescue requests during disasters have been changing. Citizens in need of help use social media, like Twitter, for expressing their rescue requests. To utilize such rescue requests on social media, it is a key to understand real rescue requests on social media. This study captured real disaster-related tweets from several flood disasters in 2018 and 2019 in Japan and analyzed the tweets. We observed that tweets having rescue-related keywords are classified into the eight categories, which include not only rescue requests but also non-rescue requests. Furthermore, most of the rescue-related tweets are unrelated to rescue requests. Next, we conducted preliminary experiments of classifying rescue requests from tweets using a BERT-based classifier. Moreover, we built a classifier based on GRU and LSTM-based recurrent neural networks, and analyzes the reason why our classifier based on machine learning misses rescue requests. The experiments revealed several lessons for building classifiers of rescue requests.

ACKNOWLEDGMENTS

The work has been supported by NICT under grant agreement no. 19303.

REFERENCES

[1] M. Jahanian, Y. Xing, J. Chen, K. K. Ramakrishnan, H. Seferoglu, and M. Yuksel, "The evolving nature of disaster management in the internet and social media era," in *Proceedings of IEEE International Symposium on Local and Metropolitan Area Networks* (2018).

[2] B. Stelter, "How social media is helping Houston deal with Harvey floods." https://money.cnn.com/2017/08/28/media/harvey-rescues-social-media-facebook-twitter/index.html (2018).

[3] Wikipedia, "2018 Japan floods (heavy rain of July 2018)." https://en.wikipedia.org/wiki/2018_Japan_floods (2018).

[4] Wikipedia, "Typhoon hagibis (the 19th typhoon of 2019 in japan)." https://en.wikipedia.org/wiki/Typhoon_Hagibis_(2019) (2019).

[5] Japan Broadcasting Corporation (NHK), "Nagano-prefecture collected rescue requests from Twitter and it contributed to saving about 50 victims." https://bit.ly/3741noG (in Japanese) (2019).

[6] J. Ma, W. Gao, P. Mitra, S. Kwon, B. J. Jansen, K.-F. Wong, and M. Cha, "Detecting rumors from microblogs with recurrent neural networks," in *Proceedings of International Joint Conference on Artificial Intelligence*, pp. 3818–3824 (2016).

[7] N. Ruchansky, S. Seo, and Y. Liu, "CSI: A hybrid deep model for fake news detection," in *Proceedings of International Conference on Information and Knowledge Management*, pp. 797–806 (2017).

[8] R. Kshirsagar, R. Morris, and S. R. Bowman, "Detecting and explaining crisis," in *Proceedings of ACM Workshop on Computer Linguistics and Clinical Psychology*, pp. 66–73 (2017).

[9] S. Sato and F. Imamura, "An analysis of tweet data tagged with "#rescue" in the 2018 west Japan heavy rain disaster: Comparative analysis with the case of 2017 north Kyushu heavy rain disaster," *Journal of Japan Society for Natural Disaster Science*, vol. 37 (2019). (in Japannese).

[10] C. Song and H. Fujishiro, "Toward the automatic detection of rescue-request tweets: analyzing the features of data verified by the press," in *Proceedings of International Conference on Information and Communication Technologies for Disaster Management* (2019).

[11] Y. Koizumi, J. Takemasa, T. Hasegawa, and Y. Kawabe, "An analysis of rescue requests on twitter in a disaster," in *Proceedings of International Workshop on Informatics* (2022).

[12] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," *arXiv1810.04805v2* (2019).

[13] Wikipedia, "Typhoon faxai (the 15th typhoon of 2019 in japan)." https://en.wikipedia.org/wiki/Typhoon_Faxai_(2019).

[14] Twitter, Inc., "Twitter api documentation—search tweets." https://developer.twitter.com/en/docs/tweets/search/api-reference/get-search-tweets.

[15] Twitter, Inc. (@TwitterLifeline), "Twitter post." https://twitter.com/TwitterLifeline/status/1016519147738419201 (2018).

[16] F. Alam, F. Ofli, M. Imran, and M. Aupetit, "A Twitter tale of three hurricanes: Harvey, Irma, and Maria," in *Proceedings of International Conference on Information Systems for Crisis Response and Management* (2018).

[17] National Institute of Information and Communications Technology, "NICT BERT japanese pre-trained model." https://alaginrc.nict.go.jp/nict-bert/index.html (2020).

[18] J. Yang, M. Yu, H. Qin, M. Lu, and C. Yang, "A Twitter data credibility framework—hurricane Harvey as a use case," *International Journal of Geo-Information*, vol. 8 (2019).

[19] M. Jahanian, T. Hasegawa, Y. Kawabe, Y. Koizumi,

A. Magdy, M. Nishigaki, T. Ohki, and K. K. Ramakrishnan, "Direct: Disaster response coordination with trusted volunteers," in *Proceedings of International Conference on Information and Communication Technologies for Disaster Management* (2019).

(Received: October 31, 2022)
(Accepted: April 7, 2023)

Yuki Koizumi Yuki Koizumi is an associate professor at the Graduate School of Information Science and Technology, Osaka University, Japan. Having obtained his master's degree and his doctoral degree from Osaka University in 2006 and 2009, respectively, he possesses a wealth of academic expertise. His primary research interests encompass the realms of programmable data plane, privacy, and security on the Internet.

Junji Takemasa Junji Takemasa is an assistant professor of Graduate school of Information and Science, Osaka University. He received the master's and Ph.D. degrees in information science from Osaka University in 2016 and 2019, respectively. After receiving the Ph.D. degree, he worked as a research engineer at KDDI Research, Inc. for one and half years and moved to Osaka University. His research interests include programmable network, future Internet and anonymous communication. He is a member of IEICE, IPSJ and IEEE.

Toru Hasegawa Toru Hasegawa received the B.E., the M.E. and Dr. Informatics degrees in information engineering from Kyoto University, Japan, in 1982, 1984 and 2000, respectively. After he worked as a research engineer at KDDI R&D labs., he has become a professor of Graduate school of Information and Science, Osaka University. His current interests include future Internet. He has published over 100 papers in peer reviewed journals and international conference proceedings. He is a member of IEEE, ACM, IEICE and IPSJ.

Yoshinobu Kawabe Yoshinobu Kawabe received his B.E., M.E., and D.E. degrees in information engineering from Nagoya Institute of Technology in 1995, 1997, and 2003, respectively. He joined NTT Communication Science Laboratories, Nippon Telegraph and Telephone Corporation in 1997. In 2002, he was a visiting research scientist at MIT Laboratory for Computer Science. Since 2008, he has been with Aichi Institute of Technology and is currently a Department of Information Science professor. His research interests include term rewriting systems, process algebras, network programming languages, formal methods, security/privacy verification, and computational trust. He is a member of ACM, JSSST, IPSJ, IEICE and SOFT.

Regular Paper

A Weather-aware Microclimate Prediction using Measurements of Meteorological Observatory

Genki Nishikawa[†], Takuya Yoshihiro[‡],

[†]Graduate School of Systems Engineering, Wakayama University, Japan
[‡]Faculty of Systems Engineering, Wakayama University, Japan
{s226183, tac}@wakayama-u.ac.jp

Abstract - Microclimates, the climate measurements near the ground, are helpful for agriculture, town management, etc. To measure microclimates such as temperature and humidity, it is necessary to locate sensors at all the observation points on the ground. However, its high cost makes it unfeasible to keep sensors installed at many points. Hence, although some studies use machine learning techniques to learn the effects of topography to predict microclimate, it is difficult to obtain sufficient data to predict microclimate measurement because of various factors that affect microclimate. In this paper, we extended the conventional method that used the differences in measurements between the nearby meteorological observatory and the observation point, and proposed two ways to predict microclimate more accurately; One is using weather classification, and the other is using the meteorological measurements such as sunlight strength and wind direction.

Keywords: IoT, microclimate prediction, town management, meteorology

1 INTRODUCTION

Microclimate is the climate observed in a small area near the ground. Microclimate data is useful in several applications. For example, cultivation management using the observed microclimate in farm field contribute to automation and increases agricultural crops. Also, urban microclimate data is useful to avoid the bad effect on our health such as heatstroke. Consequently, to grasp microclimate is important. Microclimate depends on the surrounding environment and it depends on locations [1]. Hence, although it is necessary to locate sensors at all the observation points on the ground in order to measure microclimate such as temperature and humidity, keeping sensors installed at many points is unfeasible due to its large cost. Therefore, some studies proposed methods to predict microclimate.

Ueyama proposed a method that uses machine learning technique to learn the effects of topography, and predict microclimate based on topographical map [2]. However, it is difficult to obtain sufficient amount of data to predict microclimate measurement because of a wide variety of factors such as shape, materials and colors of buildings that effect on microclimate as well as anthropogenic heat and plants effect on microclimate.

To solve the problem, Kumagai et al. proposed a method that predicts microclimate measurement at an observed point based on the difference between the observation values at the observed point and those of the nearby meteorological observatory [3]. However, this method has a problem. This method simply adds the average of the differences between the observed values at the same time of prediction time in past days to the observed values of the meteorological observatory, and thus the prediction accuracy is low.

In this study, we focus on the observation that the difference depends on weather. For example, difference between temperature observed on a road paved with asphalt and the temperature observed at meteorological observatory surrounded by lawn will depend on the amount of sunlight. In this paper, we focus on weather as an important factor that increase the prediction accuracy. Therefore, we propose weather-aware microclimate prediction methods that use weather classification and meteorological observation.

The remainder of the paper is organized as follows. In section 2, we describe related work. Section 3 describes our method and materials. Section 4 presents the evaluation methods and results with our method. In section 5, we provide conclusions.

2 RELATED WORK

2.1 Prediction Based on Machine Learning

Suzui at el. proposed a method that predicts microclimate with RNN (Recurrent Neural Network) based on the observed values at the prediction places and nearby meteorological observatory [4]. The method reflects the effect of surronding environment on the predicted values because it is based on the relation between the measurements at the prediction places and the nearby meteorological observatory. However, we need a large volume of learning data to use machine learning techniques, and observing micloclimate measurements at the prediction places for a long time is practically unfeasible.

2.2 Prediction Based on Topography Effects

Ueyama at el. proposed a method that predicts microclimate by learning the effects of topography [2]. The method is mainly for farm fields including mountain areas, and predicts the highest and the lowest, and the average temperature of each day based on topography for each square area of the 10 meter grid of a map. In this study, they assume to install environmental sensors at several square cells, and utilize the differences between the observation values of the cells and that from the nearby meteorological observatory. The study computes the difference in temperature according to several features in topography as the explanatory variables to estimate the effects of topography on temperature by means of stepwise regression. Then, they predict temperature for all cells based on the difference. In their evaluation, they installed sensors at 22 places, and predicted the highest and the lowest temperature, and the average of each day in each cell. As a result, the RMSE (Root Mean Square Error) values of the daily average temperature at the observed places was 0.8 °C, that of the highest temperature was 1.4 °C, that of the lowest temperature was 1.2 °C, respectively. Their RMSE might be considered a little large because the RMSE value of the pinpoint prediction provided by Japan Meteorological Agency is 1.5 °C.

2.3 Predicting Based on Meteorological Observatory

Kumagai at el. proposed a method that first observes microclimate at several prediction places, and predicts the microclimate measurements using the difference between the observed values at the prediction places and that of the nearby meteorological observatory at each predefined time segment [3]. In this study, their method predicts microclimate of each time segment by adding the differences of each time segment to the observed values of the meteorological observatory. Consequently, if we install a sensor at a prediction place for a certain period of time, we can predict the microclimate measurements at the place even after the sensor is removed. However, although the difference depends on weather and surrounding environment, this study do not take it into account. Hence, there is room for improvement.

3 PROPOSED METHOD

3.1 Overview

In this study, we propose a method that predicts microclimate with higher accuracy than the existing methods by extending the method of Kumagai et al. [3]. The conventional method predicts the differences between the observation values of the meteorological observatory and the microclimate measurements at each prediction time by calculating the average of the differences at the same time in the past days. Then, this method adds the differences to the observation values of the meteorological observatory for each prediction time to predict the microclimate measurements. We propose a method that predicts the differences with higher accuracy than the conventional method to improve the microclimate prediction accuracy. We observed the measured data and found that the difference between the observation values at the prediction places and that of the nearby meteorological observatory have different trends depending on weather. Accordingly, our method predicts the differences of the observation values between the prediction places and the nearest meteorological observatory at each prediction time by utilizing the trend of differences.

In Japan, meteorological measurements should generally be made in accordance with the Japan Meteorological Agency's guidebook [6] in order to avoid environmental influences. However, microclimate measurements are inevitably affected by the environment, equipment, installation methods, and other factors. In other words, the prediction of microclimate measurements must take all of these influences into account. Therefore, in this study, we predict the measurements when the same equipment is installed in the same location in the same manner as during the period of the learning data. This assumption, which is common to both our method and [3], is valid and appropriate because all those effects are included in the past observation data to be learned.

Figure 1 shows the common process of both our methods and the existing method [3] that predicts the microclimate measurement by using the predicted differences at each prediction time. Specifically, Fig. 1 shows an example of the microclimate prediction process for a day at a prediction place in a farm. We predict the microclimate measurements of the day at the prediction place from the observation data of the nearest meteorological observatory and the past measurements at the prediction place. The upper charts on the left of this figure show the time series microclimate observations at the prediction place in the past k days, and the lower charts show the time series observations of the meteorological observatory in the past k days. Our method predicts the differences using those observation values of the k-day learning period. The lower chart on the right of this figure shows the predicted differences of the prediction day, and the middle chart shows the observation values of the meteorological observatory of the prediction day. Our method and [3] add them together at each time of day to obtain the microclimate prediction, resulting in the graph on the upper right.

As written above, our method predicts microclimate measurements at a prediction place from the observation data of the nearby meteorological observatory and the past observation data at the prediction place. This is useful because we can predict microclimate measurements at the prediction place even after we remove the sensor at the prediction place. Note that it takes costs to maintain sensors long time. The proposed method enables us to obtain measurements for a long time by placing a sensor at the prediction place for a limited time period.

In this study, we propose two methods to predict the differences of the observation values to improve the prediction accuracy. The first method predicts the differences by using the weather classification obtained from the website. The second method predicts the differences by using the meteorological observation values such as sunlight strength, wind direction, and wind speed of the meteorological observatory. Note that the first method is applicable only if some weather report is available, whereas the second method requires specific measurement values. Users can make a good prediction using only weather reports, and can make a better prediction if specific measurement values related to sunlight and wind are available.

Note that the prediction steps shown in Fig. 1 is common among [3] and our proposed methods. The difference is the algorithm to compute the "difference" between measurements of the meteorological observatory and of the sensor located at the prediction place, which is shown in Fig. 1 as "predicts the differences of the prediction day." The specific difference is shown in the following.

Kumagai et al. [3]: The differences are computed by averaging the past differences between the meteorological observatory and the sensor at the prediction place.

Proposed method 1: The differences are computed by taking into account weather reports to make better prediction accuracy.

Proposed method 2: The differences are computed by using specific data measured by the meteorological observatory, etc., to further improve the accuracy.

The procedures of our first method are as follows. In the first step of our first method, we install the environmental sensors and observe microclimate for a certain period of time as learning data for the prediction places. In addition, we calculate the differences of the observed values between prediction places and the nearby meteorological observatory. The second step of this method classifies weather of each observed time and extracts the differences if the weather is the same as the prediction time. Finally, the method calculates the average of the extracted differences to predict the differences at the prediction time.

The procedures of our second method are as follows. The first step of our second method is the same as the first method. This step observes microclimate for a certain period of time as learning data for the prediction places and calculates the differences of the observed values between prediction places and the nearby meteorological observatory. The second step of this method obtains 5 kinds of meteorological observation values, i.e., sunlight strength, wind direction, wind speed, relative humidity and air pressure measured at the prediction place or the nearest meteorological observatory. Here, we define the distance between observation time point and prediction time point as the euclid distance in the 5-dimensional space. Finally, the method predicts the difference for the prediction time by using the differences computed from the time points whose distance in the 5-dimentional space is relatively low.

3.2 Microclimate Prediction Method

We assume to install the environmental sensors and observe microclimate for a certain period of time at the place for prediction. Then, we remove the sensors to observe the microclimate and predicts the microclimate measurements from this time on. We represent an observation time by t and a prediction place by p. Additionally, we represent the microclimate measurement at the prediction place p at the time t by v^t and the observation value at the nearest meteorological observatory at time t by V^t. Let the time to predict the measurement be t'. Our method predicts the difference $S_p^{t'}$ between $v^{t'}$ and $V^{t'}$ by using the differences S_p^t, where t is included in the period of learning data, to predict the unknown microclimate measurement at time t'. In this study, we propose two methods to predict $S_p^{t'}$ by utilizing the trend of S_p^t depending on the weather on the microclimate measuremenrs in the period of learning data.

The first method predicts the differences based on the weather classification obtained from such as weather forecast. The second method predicts the differences by using the meteorological observation values such as sunlight strength, wind direction, and wind speed. Note that the observation values are more detailed than the weather classification. According to the conventional study [3], it is effective to predict the microclimate from the differences of almost the same time in the past days. Hence, our methods uses the differences of almost the same time in the past days. In this study, we use the differences computed for the time in between before 30 minutes and after 30 minutes of prediction time t' in the each past observation day. We specifically describe the methods to predict the difference $S_p^{t'}$ at time t' in subsection 3.2 and 3.3.

We predict the microclimate measurements by using the differences S_p^t depending on the two methods. If we obtain the observation values of the meteorological observatory $V^{t'}$, we obtain the prediction value $e_p^{t'}$ for the prediction time t' and the prediction place p from the following equation (1).

$$e_p^{t'} = V^{t'} + S_p^{t'}. \tag{1}$$

3.3 Calculating Differences Based on Weather

In this subsection, we describe a method to predict $S_{t'}$ based on the weather classification, which we can obtain from such as weather forecast. This method predicts the differences based on the weather classification that we can commonly available. Since the difference has different trends depending on weather, our method extracts the differences at the time points when the weather is similar to the prediction time t' and also the time in a day is close to t', in the past observation days. Our method calculates the average values of the extracted differences and uses the average values as the predicted difference of prediction time t'.

Specifically, this method predicts the difference $S_{t'}$ in equation (1) shown in Section 3.2 by using the weather classification. This method predicts the difference at time t' based on the differences computed for each time in between before 30 minutes and after 30 minutes of prediction time t' in each past observation day. More specifically, we denote the time t of day d by $t_{(d)}$, the number of past observation days by k, and the time we extracted from by $t \in [t_{(d)} - 30, t_{(d)} + 30], d = 1, 2, \ldots, k$, where the unit of time is a minute. Our method classifies the weather of each of the time t and prediction time t' into the 3 classes, i.e., sunny, cloudy, and rainy based on the weather forecast. We denote the weather of time t by $w(t)$ and the set of time t included in the same weather class as the prediction time t' by X. Let the difference of the measurements between the prediction place p and the nearest meteorological observatory at time t by $D(p, t)$. We obtain the $S_p^{t'}$ from the following equation (2).

$$S_p^{t'} = \frac{1}{|X|} \sum_{t \in X} D(p, t) \tag{2}$$

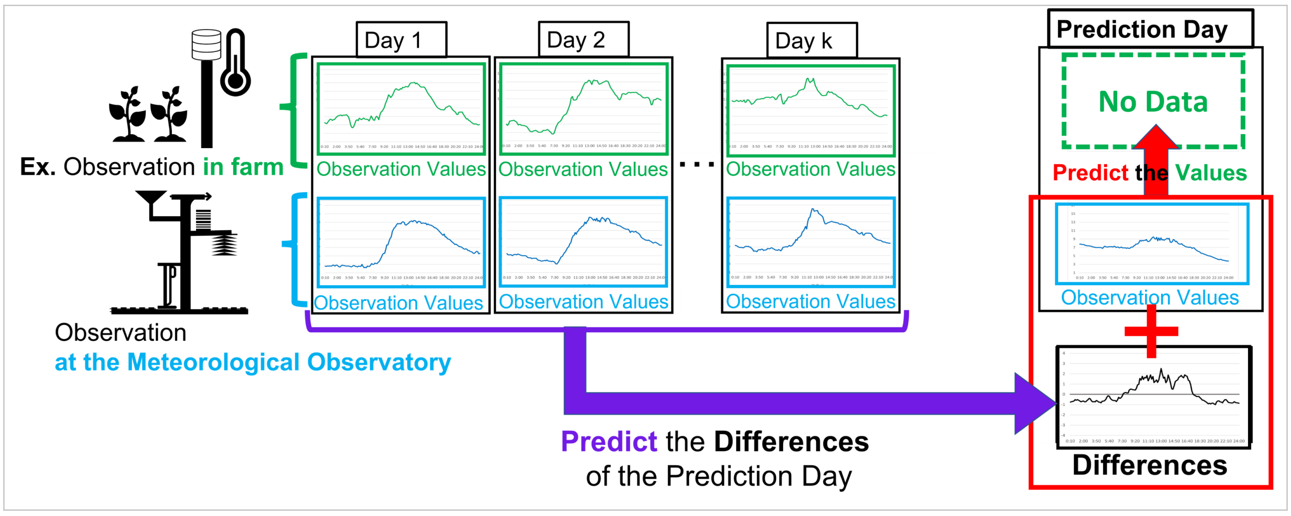


Figure 1: Overview of Predicting Microclimate

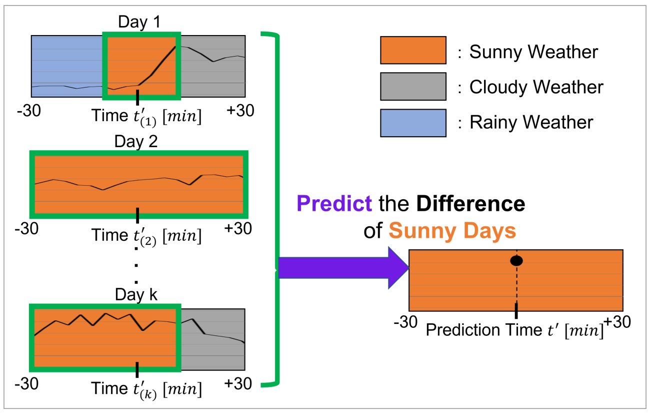


Figure 2: Predicting the Difference of Sunny Days

We show an example of predicting the difference in Fig. 2. Figure 2 shows the process to predict the difference at time t' in a sunny day based on the measured differences between the predicting place p and the nearest meteorological observatory in the past k days. The back ground colors of the charts in this figure show the weather classes of each time, i.e., sunny, cloudy, and rainy weather obtained by the weather forecast. Each of the orange, gray, and blue parts shows the sunny, cloudy, or rainy weather. Three charts in the left-hand side show the time series of the measured differences in the time in between before 30 minutes and after 30 minutes of time t' in each of the past k observation days. This method extracts the differences of the time where weather is the same as that of prediction time t'. The time of which we extract the differences are shown by the green square in this figure. This method predicts the difference at the prediction time t' shown in the right-hand side based on the extracted differences.

3.4 Calculating Differences Based on Meteorological Measurements

In this subsection, we describe the method to predict the difference $S_p^{t'}$ by using the meteorological observation values such as sunlight strength, wind direction, and wind speed. In this study, we focus on the meteorological observation values in addition to the weather as important factors that effect on the differences. We observed the differences and the meteorological observation values, and found that the difference has different trends depending on the observation values. Therefore, we propose a method that predicts the differences by using the meteorological observation values. That is, if the observation values are available, more accurate prediction will be possible.

Specifically, the prediction method in this section predicts $S_p^{t'}$ for prediction time t' based on the differences computed for the time in between before 30 minutes and after 30 minutes in prediction time t' in each of the k past observation days. More specifically, we denote the time t of day d by $t_{(d)}$, the number of past observation days by k, and the time we extracted from by $t \in [t_{(d)} - 30, t_{(d)} + 30], d = 1, 2, \ldots, k$, where the time unit is a minute. In addition to the differences, this method uses the meteorological observation values, i.e., sunlight strength, wind direction, wind speed, relative humidity, and air pressure, observed in the area that contains both the prediction place and the meteorological observatory at the same time t. Since the differences between two time points tend to be small if those five meteorological observation values take similar values, this method extracts the differences of the n-nearest neighbor time points in the 5-dimensional space in which all time points in the k days are plotted. From those n-nearest time points, we predict the difference of time t'.

In this paper, the meteorological observation values that we used are sunlight strength $[\mathrm{lx}]$, wind direction $[16 - \mathrm{Directions}]$, wind speed $[\mathrm{m/s}]$, relative humidity $[\%]$, and air pressure $[\mathrm{hPa}]$, observed by the meteorological observatory. We plot the observation values at time t and prediction time t' in the 5-dimensional space. Here, we define the distance between time t and prediction time t' as the Euclid distance in the 5-dimensional space.

Assuming that those five meteorological observation values follows the normal distributions, we normalize the observation values to so that they are comparable among different dimensions. Specifically, we transform the observation values into the values whose standard deviation α is 0.5 and mean value is 0. This transformation standardizes the observation values, and multiplies the standardized values by 0.5 in order to adjust the interval $[-2\alpha, 2\alpha]$, namely, 95% confidence interval, to the interval $[-1, 1]$.

However, the wind direction values do not follow the normal distribution because the values are angles. Hence, we transform the wind direction values into the values between -1 and 1 by min-max normalization. Since d wind direction is a plane angle, the direction is represented in range $[r-\pi \leq \theta < r+\pi]$ where r is the given reference angle. In order to extract the neighborhood points of the prediction time point in the 5-dimensional space, we set r is the wind angle of the prediction point. Thus, we transform the wind direction value at prediction time into 0, and normalize the values of the other time points between -1 and 1 by the min-max normalization. Using those normalized meteorological observation values, this method calculates the distance between each time point t and prediction time point t' as the Euclid distance of their normalized values in the 5-dimensional space.

Figure 3 shows an example to extract n-neighborhoods where we consider 3-dimensional space for conciseness. We obtain the learning data for time t' by extracting the n time points t in the ascending order of the distance from the prediction time t'. Plotted points in this figure shows meteorological measurement, and the colors of those points shows the variety of the measurements. The red point shows the measurement at prediction time t', the purple points show the neighbor measurements of the red one, and the black points show the measurements in the learning data. The light yellow circle shows the neighborhood area of the prediction time t'.

Specifically, the obtained n neighborhoods in the aforementioned 5-dimensional space are n time points of the meteorological observatory measurements. Note that we assume those five dimensions, i.e., sunlight strength, wind direction, wind speed, relative humidity, and air pressure, are all observed at the meteorological observatory. (Note that, in our evaluation in Sec. 5, sunlight strength occasionally is not available at the nearest meteorological observatory, and we used the values observed at the prediction area.) We denote this set of time points by $Y = \{t_1, t_2, \ldots, t_n\}$. This method calculates the difference for the prediction time t' based on Y from the following equation,

$$S_p^{t'} = \frac{1}{|Y|} \sum_{t \in Y} D(p, t), \tag{3}$$

where, we denote the differences between the meteorological observatory and the prediction place p at time t by $D(p, t)$.

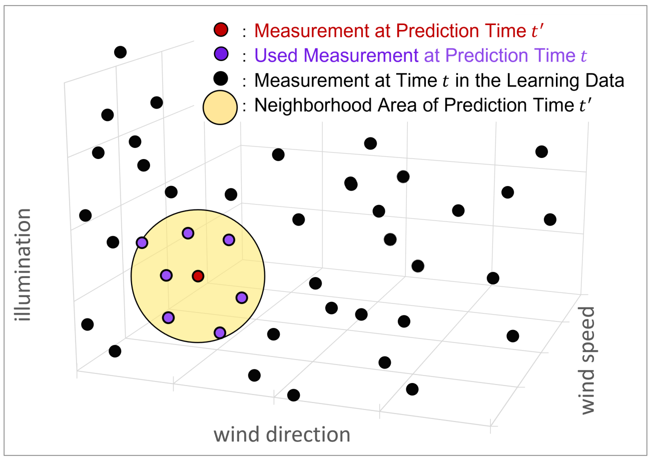

● : Measurement at Prediction Time t'
● : Used Measurement at Prediction Time t
● : Measurement at Time t in the Learning Data
● : Neighborhood Area of Prediction Time t'
illumination
wind direction
wind speed

Figure 3: Data Extraction in 3D Space

4 EVALUATION

4.1 Evaluation Method

We installed sensors 2JCIE-BL [5] made by OMRON and observed microclimate at the seven prediction places around a building in Wakayama University for 234 days from February 4 to August 25, 2021. Afterwards, we predicted the temperature as a microclimate measurement to compare RMSE and MAE(Mean Absolute Error) of errors among our two methods and the conventional method [3]. Hereafter, we refer to our methods described in the section 3.3 and 3.4 as "Proposed1" and "Proposed2." Proposed1 estimates the differences of the observation values between prediction places and the nearest meteorological observatory by using the weather classification. Proposed2 estimates the differences by using five kinds of meteorological observation values, i.e., sunlight strength, wind direction, wind speed, relative humidity, and air pressure. Figure 4 shows the seven prediction places in the map around the building of Wakayama University. The nearby meteorological observatory is Wakayama Local Meteorological Observatory. Accordingly, we use the five kinds of meteorological observation values of the local meteorological observatory. We put the sensors in the solar radiation shields to shelter the sensors from the direct effect of sunlight, wind, and rain as shown in Fig. 5. According to the Japan Meteorological Agency's guidebook [6], in meteorological observation, we installed the sensors 1.5 m above the ground to avoid the direct effect from the ground to the sensors. The guidebook is not for microclimate; however, we installed the sensors 1.5 m above the ground with a tripod because the effect of the ground is too strong in temperature measurement. We attached weights to the tripod to prevent it from tipping over or moving. The condition of the tripod was checked frequently, particularly during strong winds. If it tips over or moves, we exclude the anomalous measurement data and return it to its original location. In this way, we maintained the correctness of the data.

The measurement time interval of the sensors was 1 minute. The measured values fluctuate with time as random measurement errors. Hence, we took the moving average with the 10-minutes window, as the 10-minute duration is the mea-

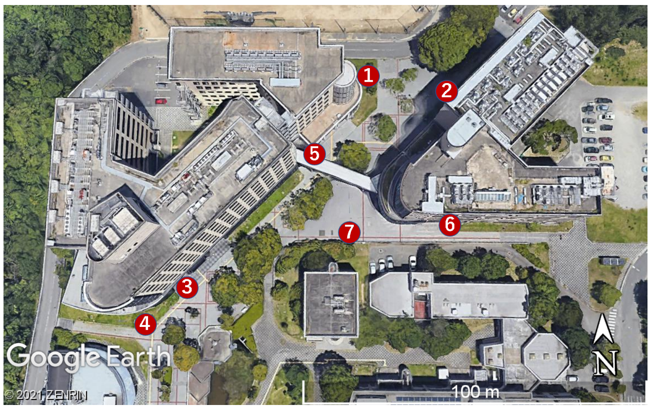


Figure 4: Observed Points (Source: Google Earth)

surement time interval of the meteorological observatory to remove the fluctuations.

In this paper, we used the sunlight measured by the sensors at the prediction places around Wakayama University because, unfortunately, Wakayama Local Meteorological Observatory does not measure the sunlight. Since the relative sunlight strength is similar if the locations are close to each other, the sunlight strength measured at Wakayama University will be applicable. Although the sensors were in the solar radiation shields, the solar radiation shields cannot completely shelter the sensors from sunlight, and our sensors measure indirectly the strength of sunlight. Hence, we can obtain the relative sunlight strength by measuring the sunlight strength at Place 6, which always is not in the shade even when the sun is at its lowest elevation in a day. Note that cloud has a large effect on the sunlight strength, and the effect is fluctuated if the cloud is devided into small pieces. Additionally, sunlight belatedly affects the microclimate because the effect is indirect through the ground warmed by the sunlight energy. Therefore, we use the average of the last 1-hour sunlight strength as the sunlight strength at that time, instead of using the instant value at the time.

We used leave-one-out cross-validation in which we predict microclimate measurements in each of the prediction days based on the dataset excluding the measurements in the prediction day. We used the data of 1 month before and after each prediction day because our methods and the conventional method do not take seasonal variations into account. Specifically, we predicted the temperature measured at each prediction place and each time in each day of our observation period from the observation data of 1 month before and after each prediction day.

4.2 Preliminary Analysis

Prior to the evaluation of our method, we observed the relationship between the differences and the five kinds of meteorological measurement to evaluate the predictability. Specifically, we observed scatter plots whose vertical axis shows the differences, and horizontal axis shows each meteorological measurement, for each prediction places. As a result, we found that the differences have different trends depending on the meteorological measurements. Figures 7 (a)-(c) show the

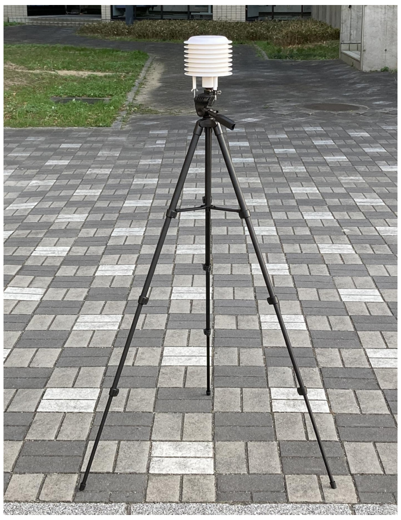


Figure 5: Observation Device

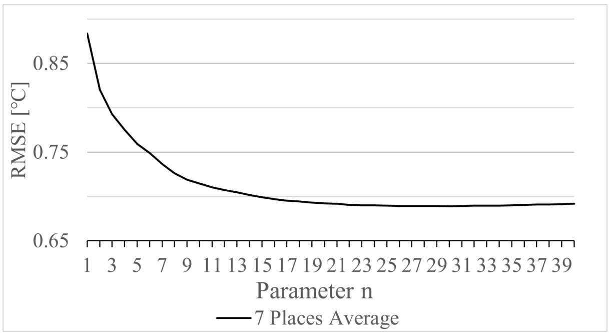


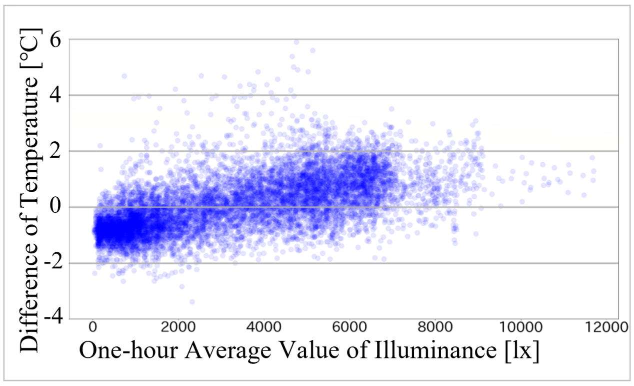
Figure 6: 7 Places Average of RMSE

scatter plots for Place 4 which notably show the result. Their vertical axes show the differences. The horizontal axes of the chart (a), (b), and (c) in the Fig. 7 show the sunlight strength, the wind direction, and the wind speed. Figure 8 shows a boxplot of the observation values for reference.

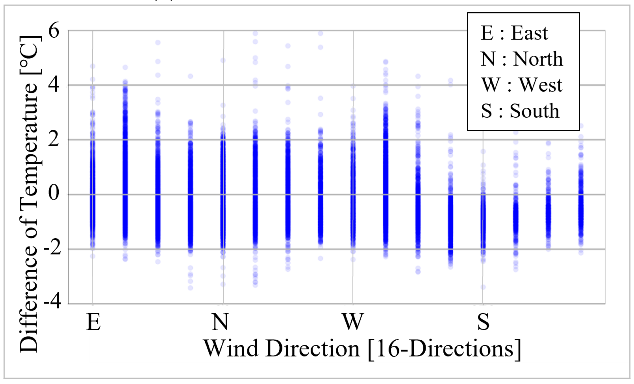
Figure 7 (a) shows the differences with the sunlight strength. We see that the differences correlate with the levels of the sunlight strength. Figure 7 (b) shows that the differences are in a large range between -2 and 4 °C when the wind direction is east-northeast or west-southwest and in a small range between -2 and 0 °C when the wind direction is approximately south. Figure 7 (c) shows the differences are approximately -1 °C only when the wind speed is 7.5 m/s or over. The differences of all places also have different trend depending on the meteorological measurements, which are similar to the case as the Place 4.

As the result, we concluded that the trends of the differences are similar when the meteorological measurements such as the sunlight strength, the wind direction, and the wind speed takes similar values. Therefore, we expect to improve the prediction accuracy by using the differences of the time points that have similar meteorological measurements to the prediction time points.

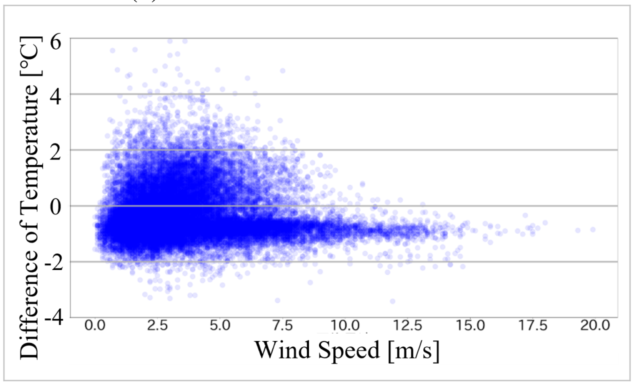
The parameter n in Proposed2 is the number of differences used to predict each microclimate measurement. Although the parameter n is an important factor to determine the ac-



(a) Difference with Illuminance



(b) Difference with Wind Direction



(c) Difference with Wind Speed

Figure 7: Scatter Charts of Difference and Observations

curacy, to determine n logically is difficult. Therefore, in this study, we predicted the microclimate measurements using each natural number of n and used the value that minimizes the average of RMSE. Figure 6 shows the average of RMSE values when n is in between 1 to 40, which shows that the average of RMSE values decreases when $30 \leq n$ and increases when $n > 30$. Hence, we use the value $n = 30$ in our evaluation.

4.3 Evaluation Results

Figure 9 shows the prediction errors of the conventional method and our two methods. Figure 9 (a) shows the errors in RMSE at each prediction places and their average. Figure 9 (b) shows those errors in MAE. Although the errors have fluctuations depending on places, the errors of the Proposed1 are totally smaller than that of the existing method at all places, and the errors of the Proposed2 are smaller than the Proposed1. Remember that the conventional method predicts the microclimate only based on the differences, but Proposed1 predicts it by using the weather classification, and Proposed2 predicts by using the meteorological observation values. As the results, if the weather classification is available, Proposed1 is the better prediction than the conventional method, and, if the meteorological observation values are available, Proposed2 is the better prediction than Proposed1.

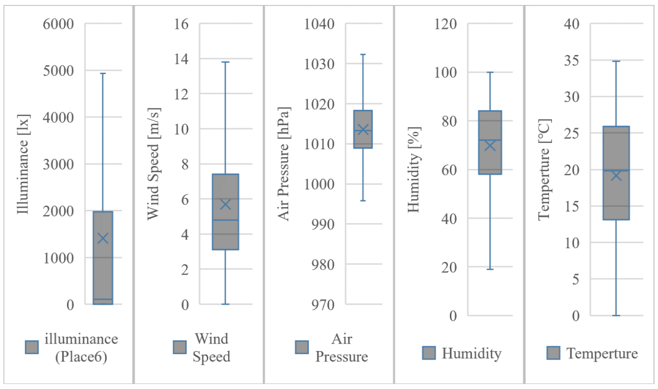
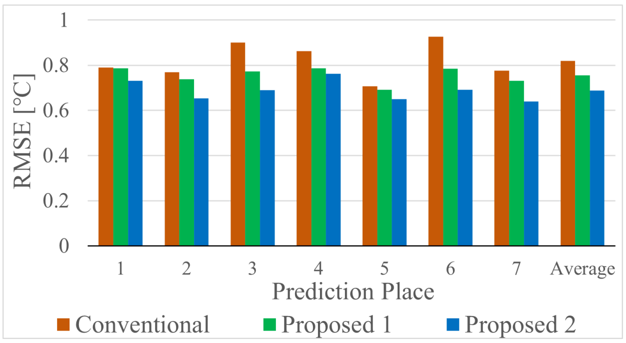


Figure 8: Boxplot of Observation Values

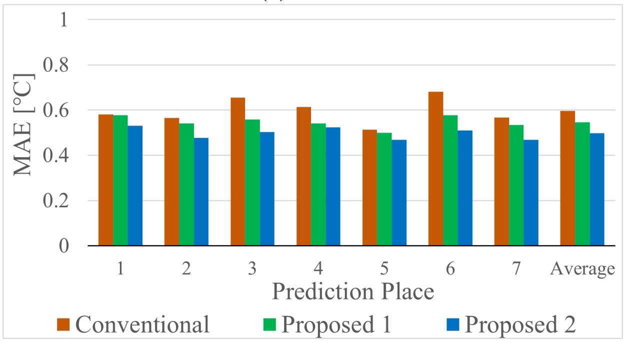
Figure 10 shows those RMSE values at each prediction time segment from 0:10 to 24:00. Figure 10 (a), (b) and (c) show the RMSE values at each prediction place with the conventional method, Proposed1 and Proposed2 respectively. Figure 11 shows the relative RMSE of our methods based on the RMSE of the conventional method, namely, if the RMSE of the proposed method is the same as the conventional one, the value is 1.0. We found that the prediction errors at night time varies little among methods and places, but the errors during the daytime vary greatly regardless of methods and places. Additionally, at Places 1 and 5, the prediction error of Proposed1 is similar to that of the conventional method, meaning that weather classification is not effective for those places. Both of those places are under tree branches or buildings and are concluded to be less affected by sunlight. Therefore, Proposed1 predicts microclimate with higher accuracy than the conventional method mainly for the places where the sunlight has large effect.

Contrary to those places, at Places 3 and 6, Proposed1has much smaller errors than the conventional method. Although most of the prediction places are located between two buildings and thus tend to be windy, Place 3 and 6 is less windy because those locations are at the front porch or not in between buildings. Therefore, we conclude that the errors are relatively small because the places are less affected by the wind and have small temperature fluctuations.

Comparing Figures 11 (a) and (b), we found that the Proposed1 improves the accuracy only during the daytime, while the Proposed2 improves it even during the nighttime. Especially, we improve the prediction accuracy by about 40% in the daytime at the prediction place. At any time point, including nighttime, Proposed2 improves the prediction accuracy by about 10-30%. From these results, we concluded that Proposed2 can improve the prediction accuracy including nighttime by using the meteorological observations, i.e., sunlight

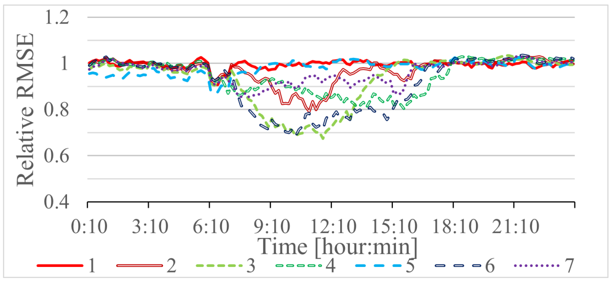


(a) RMSE

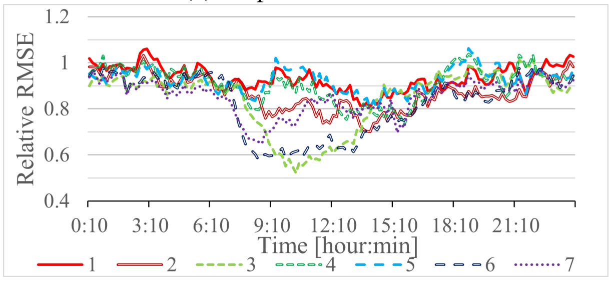


(b) MAE

Figure 9: Prediction Accuracy of each Place

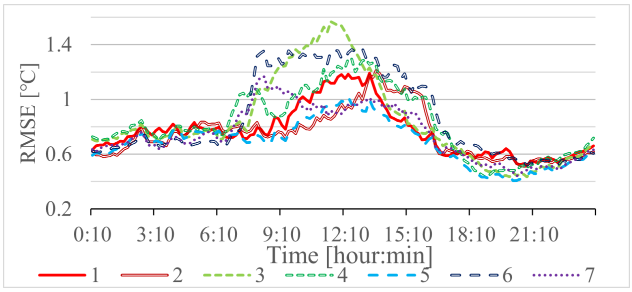


(a) Proposed Method 1

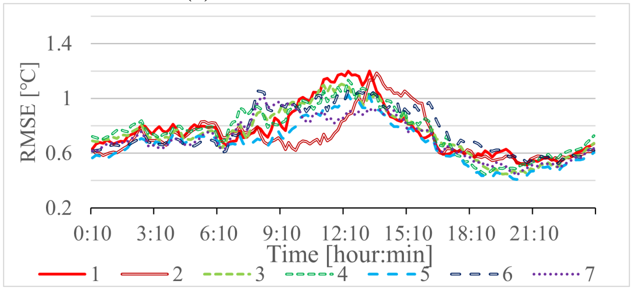


(b) Proposed Method 2

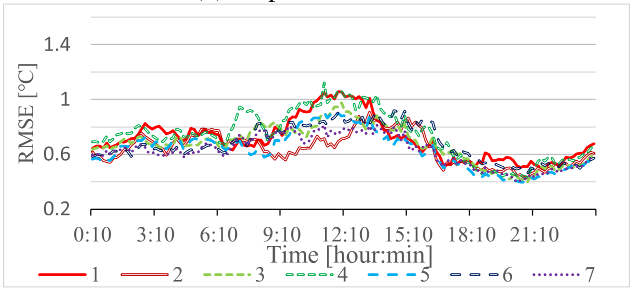
Figure 11: Relative RMSE in time Series



(a) Conventional Method

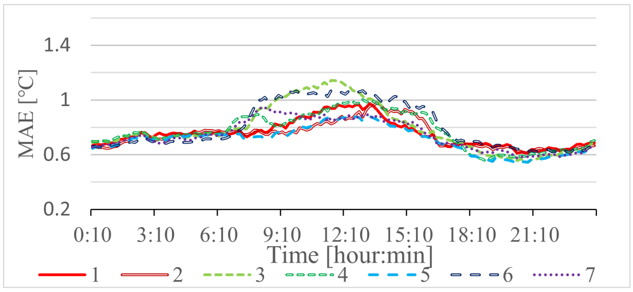


(b) Proposed Method 1

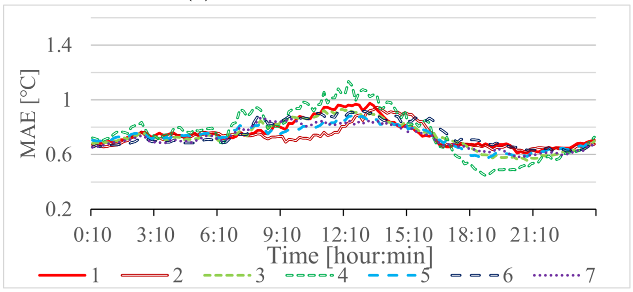


(c) Proposed Method 2

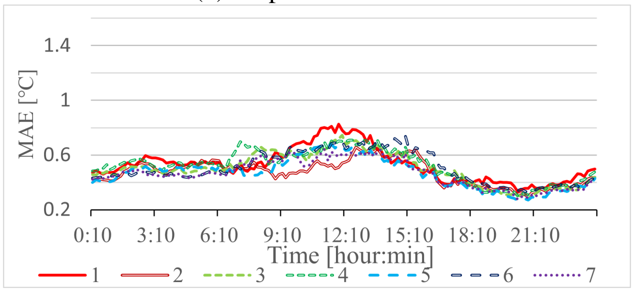
Figure 10: RMSE in time Series



(a) Conventional Method

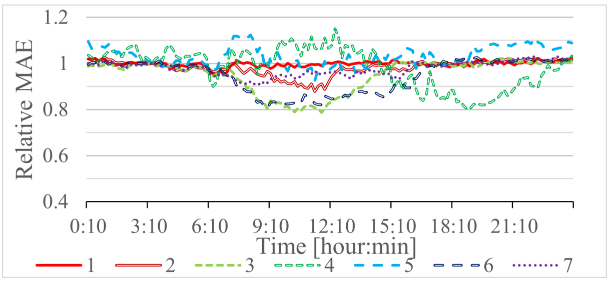


(b) Proposed Method 1

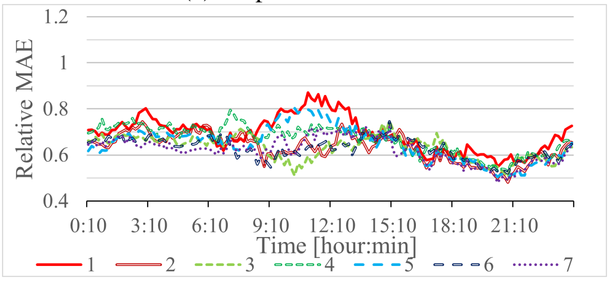


(c) Proposed Method 2

Figure 12: MAE in Time Series



(a) Proposed Method 1



(b) Proposed Method 2

Figure 13: Relative MAE in Time Series

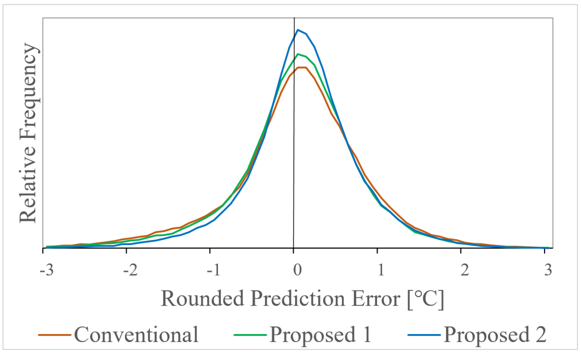


Figure 14: Relative Frequency Distribution

strength, wind direction, wind speed, relative humidity, and air pressure.

Figure 12 and 13 show the errors of Fig. 10 and Fig. 11 using MAE instead of RMSE. Comparing the errors of MAE in Fig. 13 (a), we find that the Proposed1 sometimes has a larger error than the conventional method. Note that the smaller RMSE with larger MAE means that the improvement in average is small, but extremely large error values are decreased. Therefore, we see that Proposed1 reduced the extremely large error values compared to the conventional method. In contrast, we find that the Proposed2 greatly improves the MAE including nighttime in Fig. 13 (b). In other words, Proposed1 improves prediction accuracy by reducing large errors, and Proposed2 improves the accuracy by improving the average errors.

Figure 14 shows the relative frequency distribution of the prediction errors. The horizontal axis shows the prediction errors rounded to the second decimal place, and the vertical axis shows the relative frequency of each rounded value. We found that Proposed1 has the prediction errors slightly smaller than that of the conventional method, and Proposed2 has the prediction errors that is slightly smaller than that of Proposed1.

Table 1: Characteristics of 7 places

Place	Description
1	Located under a small tree, with poor sunlight throughout the day because of two high buildings. The ground is grass and not paved. Temperatures are similar to those of the weather station at dawn, and lower than those of the weather station at other times of the day. The temperature difference from the weather station increases from morning to evening, and decreases toward dawn.
2	Direct sunlight only when the sun's altitude is high or the sun's direction is southwest. The ground is grass and not paved. The lowest temperature among the 7 places is observed during the daytime. Temperatures tend to be similar to those of Place 1, with a few hours of higher temperatures in the late afternoon.
3	No sunlight in the west. Wind is weak among the seven sites because placed at the enterance of the building. The ground is paved with stones. The second highest temperature among the seven stations is observed during the daytime. Temperatures are lower than those of the weather station from evening to midnight because of surrounded stones.
4	No sunlight from the southwest when the sun's altitude is low. The ground is grass and not paved. Higher temperatures than the weather station are observed during the daytime.
5	Under a footbridge as a roof, without daylight all day. The ground is paved with stones. Highest temperatures among the seven stations are observed from late at night to early in the morning. At other times of the day, temperatures are lower than those of the weather station.
6	No shade all day because of low buildings on the south side. Wind is weak among the seven sites. The highest temperatures are observed during the daytime among the seven sites due to the influence of sunlight. The temperature rises the fastest and drops the latest among the seven sites.
7	The site is further away from high buildings than other sites. The site receives only southeast exposure when the sun's altitude is low. The ground is paved with stones. The temperatures are always around the average of all the 7 sites. Temperatures are higher than those of the weather station in the morning and lower at other times of the day.

Table 1 shows the characteristics of these seven sites. Places 4 and 6 have high buildings to the north (about 30 m and 23 m, respectively), which provide shade in many places. On the other hand, the buildings on the south side of places 4, 7, and 6 are all low (under 5 m), allowing sunlight to reach the site from the south. As a result, as shown in Fig. 9, 10, etc., places 3, 4, and 6 have relatively long sunlight hours, resulting in large prediction errors. The proposed method is particularly suited to reduce the error due to such sunlight effects.

5 CONCLUSIONS

We proposed two weather-aware microclimate prediction methods that predict microclimate measurements based on the differences between the past measurements at the prediction place and that of the nearby meteorological observatory. One of our methods utilizes weather classification to predict microclimate measurements, and the other utilizes multiple meteorological measurement values.

We evaluated our methods using a data set measured nearby Wakayama University. Specifically, we measured microclimates at seven prediction places around a building of Wakayama

University for 234 days. As a result, we showed that both of our methods outperform the conventional method, and among the two proposals, the method based on meteorological measurements has higher performance than that based on simple weather classification. It means that, if detailed data is available, better prediction of microclimate is possible.

In future work, it would be important to analyze the seasonal effect in order to reduce the amount of data for microclimate prediction. Knowing the relationship among different microclimate measurements also will contribute to data reduction and performance improvements.

REFERENCES

[1] Ministry of the Environment, Heat Illness Prevention Information, ⟨https://www.wbgt.env.go.jp/wbgt.php⟩ Referred in October 2022 (in Japanese).

[2] H. Ueyama, ”Developing Applications to Create 50m-mesh Data of Temperature, Sunlight Strength, Relative Humidity, Reference Evaporation”, Bulletin of the NARO Agricultural Research for Western Region, Vol. 19, pp. 13-43 (2019) (in Japanese).

[3] K. Kumagai, T. Uchibayashi, T. Abe, T. Suganuma, ”Predicting Microclimate from Sensor Data for Town Management”, IPSJ SIG Technical Report, , Vol. 2016-IS-138 No.10, pp. 1-7 (2016) (in Japanese).

[4] S. Suzuki, N. Fujiki, ”Prediction and Supplement of Meteorological Data Using Recurrent Neural Networks”, The 82nd National Convention of IPSJ, pp. 285-286 (2020), (in Japanese).

[5] Omron, 2JCIE-BL01 Environment Sensor, ⟨https://www.omron.co.jp/ecb/product-info/sensor/iot-sensor/environmental-sensor⟩ Referred in October 2022 (in Japanese).

[6] Japan Meteorological Aency, ”A Guidbook for Meteorological Observation”, ⟨https://www.jma.go.jp/jma/kishou/know/kansoku_guide/guidebook.pdf⟩ Referred in October, 2022 (in Japanese).

(Received October 11, 2022)
(Accepted May 22, 2023)

Genki Nishikawa is a graduate student of Graduate School of Systems Engineering, Wakayama University, Japan. He received the B.E. degree from Wakayama University, in 2021. His research interests include IoT and meteorology. He is currently with Core Corporation.

Takuya Yoshihiro is an associate profesor of Faculty of Systems Engineering, Wakayama University, Japan. He received the B.E., M.I., and Ph.D. degrees from Kyoto University, in 1998, 2000, and 2003, respectively. He was an Assistant Professor at Wakayama University, from 2003 to 2009, where he has been an Associate Professor, since 2009. He is currently interested in graph theory, distributed algorithms, computer networks, wireless networks, medical applications, bioinformatics, IoT, and data ecosystems. He is a member of ACM, IEICE, and IPSJ.

Regular Paper

Pedestrian Cooperative Autonomous Mobility -Path Planning Adapted to Pedestrian Face Direction-

Yuto Yada[*], Shunsuke Michita[*], Seiji Komiya[*] and Toshihiro Wakita[*]

[*]Graduate School of Engineering, Kanagawa Institute of Technology, Japan
s2284001@cco.kanagawa-it.ac.jp

Abstract - Autonomous mobility in mixed traffic environments with pedestrians need functions to avoid contact with pedestrians. In this study, path planning method adapted to pedestrian face direction was developed. For pedestrians who are aware of mobility (forward facing walking), small avoidance path is generated. For pedestrians who are unaware of mobility (downward facing walking), such as those who are walking on their smartphones, large avoidance path is generated. Subjective evaluation experiments were conducted on four items: distance, speed, smoothness of avoidance, and reliability. The subjective evaluation results showed that the evaluations improved for all items except speed, both for forward and downward walking. In particular, for downward facing pedestrians the evaluation of the distance was considerably improved. In addition, objective evaluation indicators corresponding to the subjective evaluations were examined.

Keywords: Autonomous Mobility, Pedestrian behavior prediction, Yolo

1 INTRODUCTION

In recent years, the practical application of autonomous mobility has been progressing worldwide in areas such as office building security and package delivery. Autonomous mobility moves in mixed traffic environments with pedestrians need a path planning function that avoids contact with pedestrians. When humans pass each other, they unconsciously make eye contact with each other and anticipate the other's movements to ensure smooth movement. Therefore, we are working on the realization of autonomous mobility that enables this type of behavior.

DWA (Dynamic Window Approach) [1] and RRT (Rapidly exploring random tree) [2] have been widely used as static obstacle avoidance methods for mobile mobility. The problem with these previous studies was that dynamic obstacles such as pedestrians could not be avoided because they were not considered.

For dynamic obstacle avoidance, pedestrian prediction using the Kalman filter [3], pedestrian prediction and avoidance using the potential method [4], and the application of ORCA (Optimal Reciprocal Collision Avoidance) to the prediction and avoidance of multiple pedestrians [5] have been studied.

One of the problems for previous studies is avoidance for pedestrians walking on their smartphones. It is difficult for mobility to avoid pedestrians walking while gazing at their smartphones. This is because their walking path is unstable and behavior prediction is difficult. Also, pedestrians walking on their smartphones may be surprised when mobility suddenly appears in their field of vision when they pass by at close range while they are gazing at their smartphones. To solve the problem, a method of warning by sound can be considered. Although pedestrians may notice mobility with sound warnings, this method causes mobility to impede pedestrians' walking, and frequent warning sounds can make pedestrians uncomfortable. Especially when autonomous mobility increases in the future, it is unlikely that autonomous mobility will always be prioritized over pedestrians. As another means, a method of large avoidance can be considered. Although the method could avoid the pedestrian safely, such large avoidance would be excessive for pedestrians walking forward. If excessive avoidance is always performed, there is a high possibility that it will take a long time to arrive at the destination or the route cannot be generated and the mobility cannot move. Pedestrians appear to avoid other pedestrians according to their characteristics.

Therefore, in this study, a method to adjust the amount of avoidance according to the face direction was attempted. After predicting the pedestrian's behavior, the risk of collision is reduced by avoiding a small amount when the pedestrian's face is in front of the vehicle and a large amount when the face is facing downwards. Despite avoiding pedestrians using this method, if a collision is unavoidable, the mobility stops. This avoidance strategy is similar to that used by humans every day.

2 PEDESTRIAN BEHAVIOUR EXPERIMENTS

Pedestrian trajectory measurement experiments were conducted to see how pedestrians avoid each other.

The experiment was conducted on 12 subjects. Each subjects were asked to walk at a natural speed and pass opposite pedestrians. The starting position was completely face each other and the walking distance was 10 m.

An example of pedestrian trajectory against a forward-facing pedestrian is shown in Fig. 1. And an example of pedestrian trajectory against a downward-facing pedestrian is shown in Fig. 2.

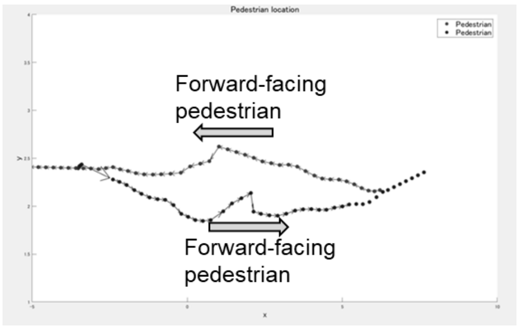


Figure 1: An example of pedestrian trajectory against a forward-facing pedestrian.

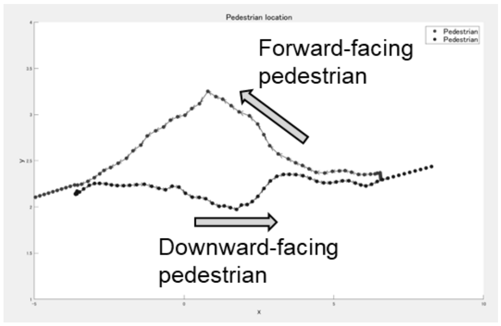


Figure 2: An example of pedestrian trajectory against a downward-facing pedestrian.

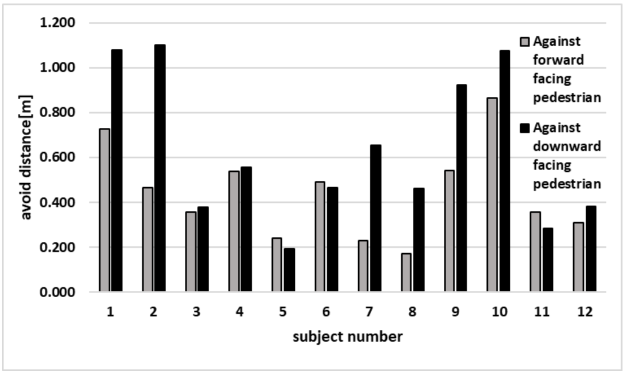


Figure 3: The avoidance distance result.

The amount of avoidance against forward-facing pedestrians and the amount of avoidance against downward-facing pedestrians were compared. The amount of avoidance is shown in Fig. 3.

Significant difference tests were conducted paired t-test with a significance level of 0.01 and a null hypothesis of ‘no difference in mean values between the two groups. The two groups are the amount of avoidance against forward-facing pedestrians and the amount of avoidance against downward-facing pedestrians. The results of the significance difference test showed a p-value of 0.0077, with a significance level of 1%. This shows that pedestrians, on average, largely avoid other pedestrians who are downward-facing pedestrian. Therefore, in this research, an attempt was made to realize such human behavior in autonomous mobility.

3 METHOD

To safely avoid a downward-facing pedestrian, the face direction of the pedestrian is recognized by face direction recognition. Next, if the result of the face direction is a downward-facing pedestrian, a large avoidance path is generated. We considered these two requirements for pedestrian avoidance.

The method is based on the following procedure.

Step1 Measurement of pedestrian position using 3D LiDAR, pedestrian detection, tracking, and pedestrian behavior prediction.

Step2 Pedestrian face direction detection by image recognition .

Step3 Collision risk area calculation based on the pedestrian behavior prediction and pedestrian face direction detection results.

Step4 Pedestrian avoidance route generation.

A schematic diagram of the proposed algorithm is shown below (Fig. 4).

As in previous studies [3], 3D LiDAR information and a Kalman filter were used for pedestrian recognition and pedestrian behavior prediction.

3.1 System Configuration

The autonomous mobility used in this experiment is shown in Fig. 5. It was equipped with a camera for face direction detection of pedestrians and an omnidirectional laser sensor for self-localization and obstacle detection. Data obtained from these onboard devices is processed and controlled by a compact computer DH310 (Shuttle) and a Jetson Xavier NX (NVIDIA) (Table. 1).

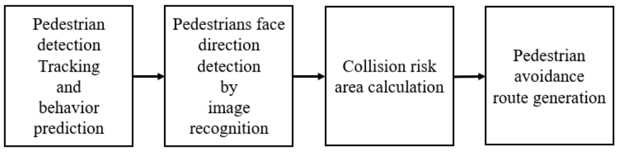

Pedestrian detection Tracking and behavior prediction
Pedestrians face direction detection by image recognition
Collision risk area calculation
Pedestrian avoidance route generation

Figure 4: Method of pedestrian cooperative path planning.

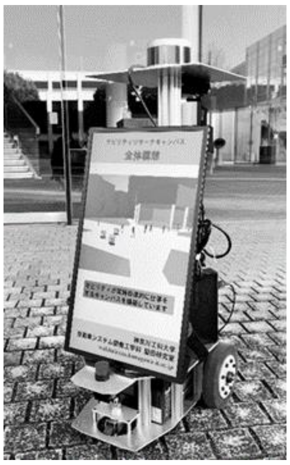


Figure 5: Mobility.

Table 1: System Configuration.

Camera	C920n web camera (Logicool)
LiDAR	VLP-16 (Velodyne)
Computers	DH310 (Shutlle)
	Jetson Xavier NX (NVIDIA)

3.2 Face Recognition

To estimate whether pedestrians are aware of autonomous mobility or not, this study assumes that pedestrians whose faces are forward-facing are aware of autonomous mobility and those whose faces are downward-facing are not aware. Pedestrian face direction recognition was performed using deep learning with Yolo [6]. An example of recognition is shown in Fig. 6. First, 300 images were taken of each pedestrian with a forward face and a downward face. Next, 4000 epochs of deep learning by Yolo were performed using the collected images. The recognition rate of forwarding-facing was about 99% and that of downward-facing was about 77% (Table. 2).

3.3 Path Planning

Route generation was performed by RRT* [7] based on the occupancy grid map.

Based on the results of pedestrian face direction recognition and pedestrian behavior prediction, collision risk areas were defined on the occupancy grid map used in route generation (Fig. 7). For pedestrian behavior prediction, the walking speed of the tracked pedestrian was calculated using a Kalman filter, and the predicted position was calculated based on the calculated walking speed.

The width of the collision risk area is the same as the width of the body when avoiding a forward-facing pedestrian (hereinafter referred to as 'small avoidance'), and is wider when avoiding a downward-facing pedestrian (hereinafter referred to as 'large avoidance') so that a safe distance is maintained between the pedestrian and the collision risk area. The collision risk area was treated like an obstacle in the route generation.

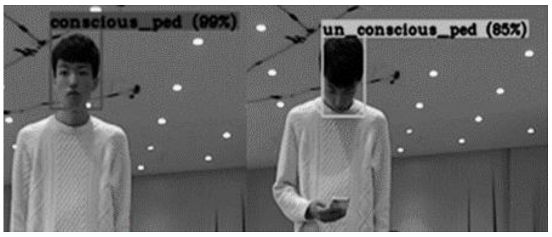


Figure 6: Recognized example of face direction.

Table 2: Recognition result of face direction.

		Recognition result	
		Forward	Down-ward
Human behav-ior	Forward	99%	1%
	Downward	23%	77%

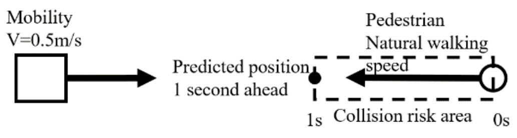


Figure 7: Collision risk area.

4 EXPERIMENS AND RESULTS

Using the proposed path planning algorithm, a subjective evaluation experiment on pedestrian avoidance was conducted in a laboratory. A course was created as shown in Fig. 8, and the autonomous mobility was moved at a translational velocity of 0.5 m/s. Pedestrians were instructed to walk at their natural walking speed and evaluate whether the autonomous mobility could avoid pedestrians.

4.1 Path Planning Results

The path planning results are shown below. For comparison, similar experiments were conducted under path planning without behavior prediction. Three types of path planning were used: without behavior prediction (Fig. 9), small avoidance (Fig. 10), and large avoidance (Fig. 11). The bold lines are the selected paths and the branches are the candidate paths. It was confirmed that the system generated a largely avoidable path for downward-facing pedestrians compared to forward-facing pedestrians.

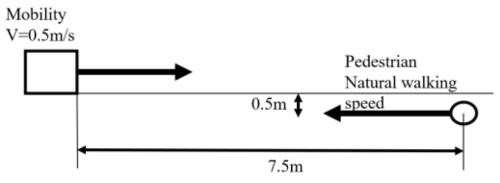


Figure 8: Layout of pedestrian avoidance experiment.

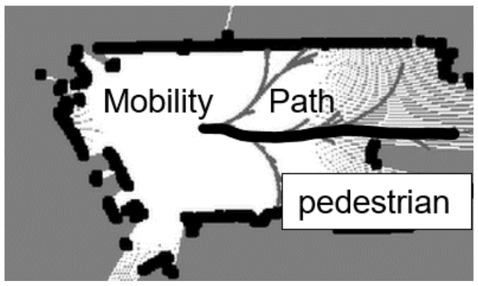


Figure 9: Path planning example of without behavior prediction path planning. The bold line represents the selected path and the branch line represent candidate path.

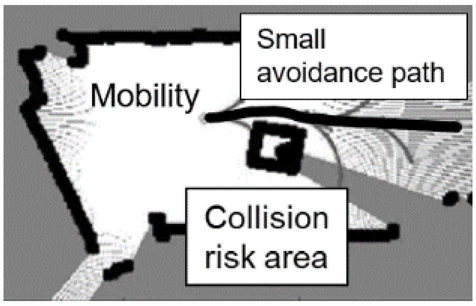


Figure 10: Path planning example of small avoidance cooperative path planning. The bold line represents the selected path and the branch lines represent candidate path. The square represents the collision risk area.

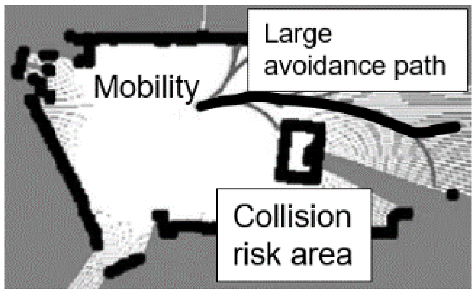


Figure 11: Path planning example of large avoidance cooperative path planning. The bold line represents the selected path and the branch lines represent candidate path. The square represents the collision risk area.

Figs 12, 13, and 14 show the trajectory examples for without behavior prediction, small avoidance, and large avoidance, respectively.

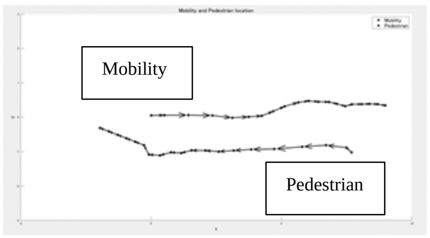


Figure 12: Trajectory example of without behavior prediction path planning.

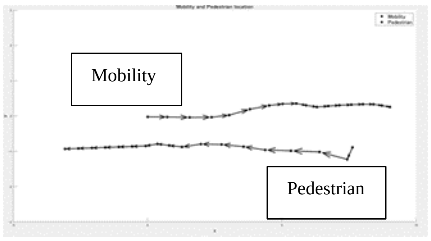


Figure 13: Trajectory example of small avoidance cooperative path planning.

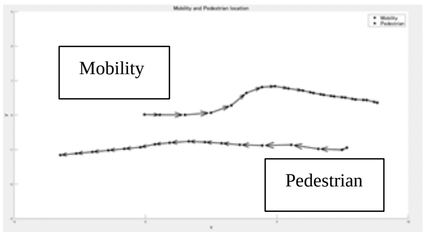


Figure 14: Trajectory example of large avoidance cooperative path planning.

The paths also showed that large avoidance was avoided to a greater extent than small avoidance and that the timing of avoidance was delayed without the collision risk area.

4.2 Subjective Evaluation Results

Subjective evaluation of pedestrian avoidance performance was carried out on nine subjects. In the experiment with 12 subjects in Chapter 2, it was shown that there was a significant difference between the trajectories of avoidance against forward-facing pedestrians and those against downward-facing pedestrians. Therefore, a more detailed experiment on avoidance with mobility was conducted with nine subjects. Two trials of each condition were made to each subject. Experiments were carried out based on the approval of the Human Ethics Review Committee of Kanagawa institute of technology.

Subjective evaluation was performed with 4 evaluation items. They are "distance from the autonomous mobility when passing by", "speed of the mobility when passing by", "smoothness of passing by (avoidance performance)", and "reliability when passing by". These items were evaluated in five levels, with the following ratings: 'good', 'a little good', 'undecided', 'a little bad', and 'bad'.

Subjective evaluation results are as follows (Figs. 15 and 16). In the case of forward-facing pedestrians, the evaluation of all items improved in comparison without behavior prediction and small avoidance. In the case of downward-facing pedestrians, the evaluation values of all items improved in comparison without behavior prediction and large avoidance. Especially, in the case of downward-facing pedestrians, the evaluation of distance was considerably improved. No change was observed in the evaluation of speed, partly because the vehicles were driven at the same speed in all three conditions.

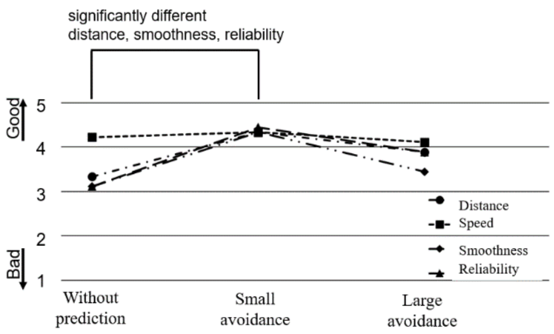


Figure 15: Subjective evaluation results for conscious pedestrians.

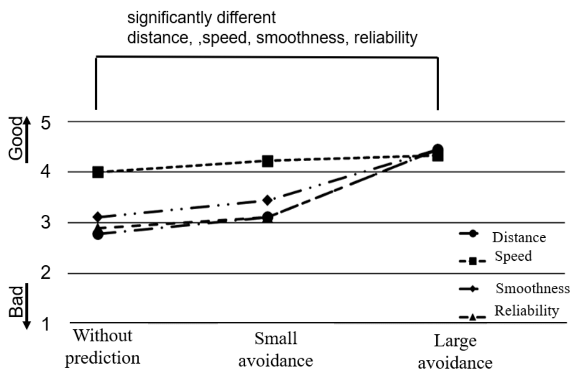


Figure 16: Subjective evaluation result.

Significant difference tests were conducted paired t-test with a significance level of 0.05 and a null hypothesis of 'no difference in mean values between the two groups. In this study, method of successive categories was used to convert subjective evaluation results from ordinal scale to interval scales. Without behavior prediction and small avoidance were compared for forward-facing pedestrians, and without behavior prediction and large avoidance were compared for downward-facing pedestrians. The p-values for each item are shown below (Tables 3 and 4).

5 PHYSICAL INDICATORS

5.1 Objective Evaluation Results

The physical quantities corresponding to the four evaluation items “distance”, “speed”, “smoothness”, and “reliability”, which were considered to be related to passing each other, were examined.

First, we considered the item of "distance" when passing each other. In this study, the nearest neighbor distance at the time of passing was used as the distance perceived by the subjects. As an example, the distance was 0.83 m in the case of without behavior prediction, 1.05 m in the case of small avoidance and 1.45 m in the case of large avoidance.

Table 3: Result of significance test for forward facing pedestrian (* p<0.05).

item	P-value
Distance	0.043* significantly different
Speed	0.173
Smoothness	0.057 significantly different
Reliability	0.025* significantly different

Table 4: Result of significance test for downward facing pedestrian (* p<0.05, ** p<0.01).

item	P-value
Distance	0.004** significantly different
Speed	0.041* significantly different
Smoothness	0.017* significantly different
Reliability	0.005** significantly different

Next, "speed" was examined. Relative speed and absolute speed can be considered as index candidates. In this study, the speed perceived by the pedestrian is considered as the relative speed of the mobility. The pedestrian was asked to walk at natural walking speed in all three conditions and the translational velocity of the mobility was constant for all three conditions. As an example, the relative velocity was 0.87 m/s in the case of without behavior prediction, 1.01 m/s in the case of small avoidance, and 0.93 m/s in the case of large avoidance.

“Smoothness” was further considered. The turning angular velocity ω during the avoidance is used as an indicator. The point where the angular velocity became 0.1 rad/s or bigger was defined as the avoidance start, and the angular velocity until it became less than 0.1 was averaged. In the case of without behavior prediction, no avoidance is performed, therefore the average value of the turning angular velocity reached to the nearest distance is used. As an example, the average value of the turning angular velocity during avoidance was 0.00 rad/s is for the without behavior prediction, 0.14 rad/s for small avoidance, and 0.29 rad/s for large avoidance.

5.2 Indicator for Reliability

A physical quantity that correspond to the "reliability" when passing each other was examined.

Time to collision (TTC) is a physical index used in the Autonomous Emergency Braking (AEB) installed in vehicles. This indicator represents the time until collision with the leading vehicle if the current relative speed of the leading vehicle to the ego vehicle is maintained. Let x_e, v_e, be the front end position and velocity of the ego vehicle and x_l, v_l be the rear end position and velocity of the leading vehicle in the world coordinate system. In this case, the relative distance d_x and the relative velocity v_x of the leading vehicle relative to the ego vehicle are $x_l - x_e$, $v_l - v_e$ (Fig. 17). Therefore, if the value of TTC is t_x, the following equation is obtained (1).

$$t_x = -\frac{d_x}{v_x} = -\frac{x_l - x_e}{v_l - v_e} \tag{1}$$

In a previous study [8], it was shown that most people drive with a TTC of 4 seconds or longer. Therefore, TTC may be used as an indicator of reliability. In this study, a physical quantity by expanding the TTC to two dimensions (2D TTC) was investigated. In addition, pedestrians are assumed to be in constant velocity linear motion and mobility is assumed to have constant translational and rotational velocity.

The method for calculating 2D TTC is as follows. It was assumed that the pedestrian would move linearly at a constant velocity and the mobility would move at a constant translational velocity and a constant angular velocity. The coordinate transformation from the world coordinate system to the mobility coordinate system is performed. The positions and velocities of the autonomous mobility and pedestrian in the world coordinate system are shown in the Fig. 18. The position and speed of the mobility are $(x_e(t), y_e(t))$ and

$\left(v_{ex}(t), v_{ey}(t)\right)$, and the predicted position and the predicted speed of the pedestrian are $\left(x_p(t), y_p(t)\right)$ *and* $\left(v_{px}, v_{py}\right)$. The distance from the center of gravity of the vehicle to the front center of the vehicle is h. The position and velocity of the pedestrian in the world coordinate system are transformed into the mobility coordinate system with reference to the mobility's center of gravity. In this case, the relative position $(x_r(t), y_r(t))$ and relative velocity $(v_{rx}(t), v_{ry}(t))$ of the pedestrian relative to the vehicle front of the mobility are $\left(x_p(t) - (x_e(t) + h), y_p(t) - y_e(t)\right)$ *and* $\left(v_{px} - v_{ex}(t), v_{py} - v_{ey}(t)\right)$ (Fig. 19).

The path distance d_p is defined as the Equation (2). The path distance represents the distance from the current location to the collision point. $t_{collision}$ represents the predicted time when the collision would occur.

$$d_p(t) = \int_t^{t_{collision}} \sqrt{v_{rx}(t)^2 + v_{ry}(t)^2 dt} \tag{2}$$

In addition, the absolute value of the relative velocity is given by equation(3).

$$v_r(t) = \sqrt{v_{rx}(t)^2 + v_{ry}(t)^2} \tag{3}$$

Therefore, if the value of the 2D TTC is t_c, it follows that equation(4).

$$t_c(t) = \frac{d_{p(t)}}{v_r(t)} \tag{4}$$

The value of the 2D TTC in the case of no collision is defined as infinite.

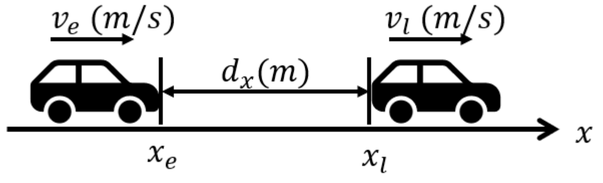


Figure 17:TTC outline figure.

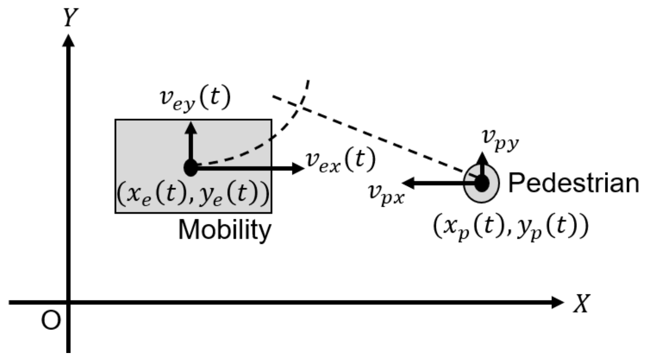


Figure 18:World coordinate system.

The 2D TTC was calculated for three conditions: without behavior prediction, small avoidance, and large avoidance. The mobility, pedestrian position and 2D TTC values during the calculation using MATLAB are shown (Figs 20, 21, and 22).

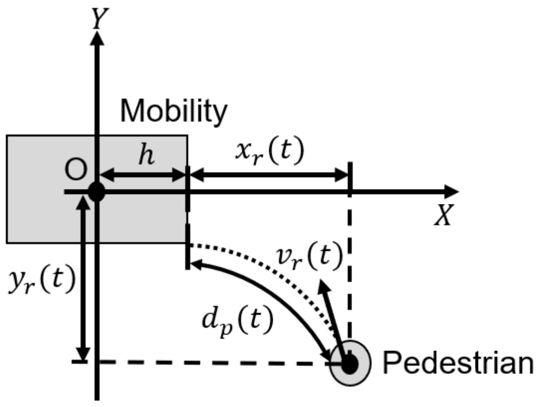


Figure 19: Mobility coordinate system.

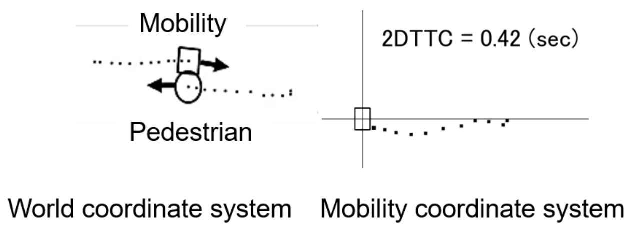


Figure 20: Without behavior prediction 2DTTC calculation diagram.

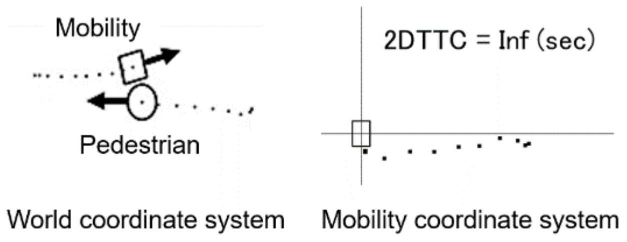


Figure 21: Small avoidance 2DTTC calculation diagram.

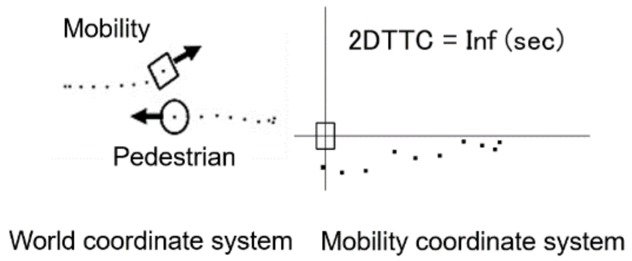


Figure 22: Large avoidance 2DTTC calculation diagram.

5.3 Example of Physical Indicators

An example of physical indicators value during forward-facing pedestrians is shown in the figure in chronological order (Fig. 23).

5.4 Relationship Between Subjective Evaluation and Physical Indicators.

The results for the subjective and objective evaluation values are shown in Figs 24, 25, 26, and 27.

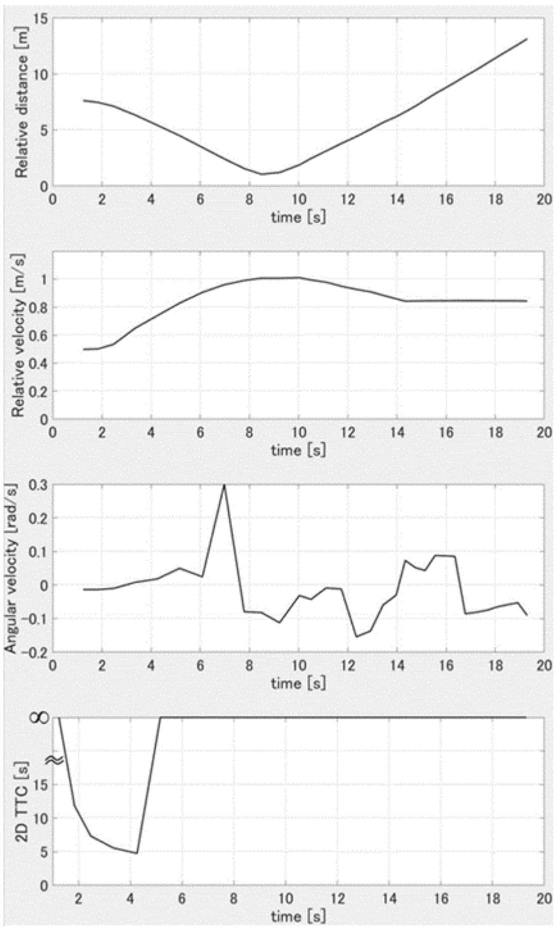


Figure 23: Example of physical indicators.

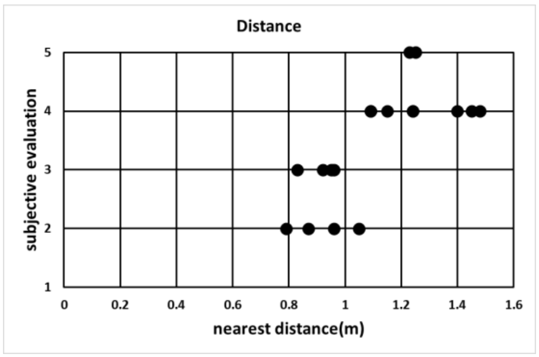


Figure 24: Relationships between distance and subjective evaluation.

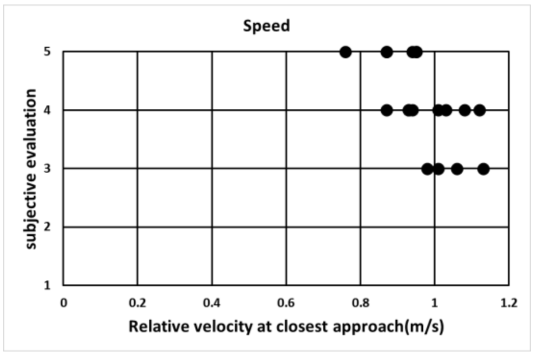


Figure 25: Relationships between speed and subjective evaluation.

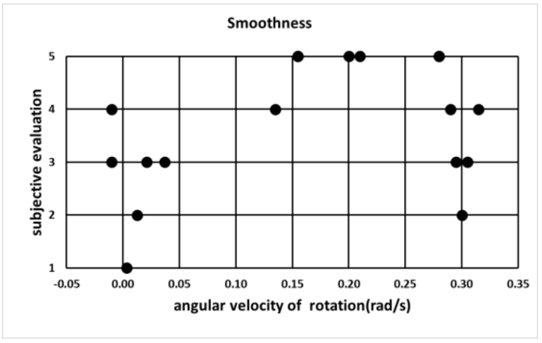


Figure 26: Relationships between smoothness and subjective evaluation.

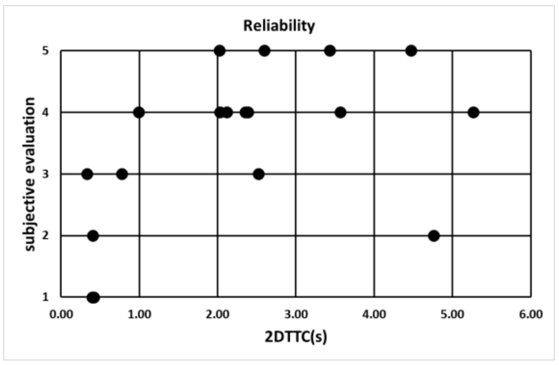


Figure 27: Relationships between reliability and subjective evaluation.

As for distance, the subjective evaluation value improves as the value of the closest neighbor distance increase.

For speed, the subjective evaluation value improves as the value of the relative speed at the closest approach decrease.

Regarding smoothness, the subjective evaluation value improves as the angular velocity of the turn increase.

The subjective evaluation value of the reliability worsens when the 2D TTC value is less than one second, while the subjective evaluation value improves when the 2D TTC value is two seconds or more.

6 DISCUSSION

6.1 The Results of the Subjective Evaluation.

Both forward and downward-facing pedestrians improved the subjective evaluation results for all items except speed. Speed was highly rated in all three conditions, with no significant differences observed. It is considered that this is because the mobility moved at a constant speed under all experimental conditions.

In the subjective evaluation of forward-facing pedestrians, the evaluation of large avoidance was slightly worse than that of small avoidance. Several subjects commented on the poor smoothness of large avoidance, such as "I felt poor smoothness" and "If I were a human, I would feel un-comfortable as if I were being large avoided". From this result, it seems that small avoidance is appropriate for for-ward-facing pedestrians.

Significant differences in distance and reliability were found for forward-facing walking. This result is thought to be due to the fact that the mobility without prediction (no collision risk area) pass pedestrians at a close distance, while those with a collision risk area maintain a certain distance while avoiding pedestrians.

In downward-facing walking, significant differences were observed in all items except speed. In forward-facing walking, the presence of mobility can be confirmed early on in the effective field of view, whereas in downward-facing walking, the effective field of view is narrower than in forward-facing walking because walking is done while gazing at the smartphone [9], and the pedestrian only confirms the presence of the mobility when it enters the peripheral field of view just before passing by. Therefore, the evaluation of distance, the reliability and smoothness decreased, whereas with the collision risk area, the reliability also improved because the robot maintains a maximum safe distance and makes a larger avoidance compared to forward-facing walking.

6.2 The Results of the Objective Evaluation.

With regard to the relationships graph for smoothness (Fig. 26), the subjective evaluation value largely changes at the small turning angle speed value. These conditions are path without behavior prediction and no avoidance is performed. Further studies are needed on these characteristics.

With regard to the relationships graph for the reliability (Fig. 27), there is an outlier where the subjective evaluation worsens despite the high 2DTTC value, but this is thought to be influenced by one subject's opinion that he felt nothing in particular about the reliability because he avoided the area a little himself. Overall, the subjective evaluation largely changes as the value of 2DTTC increased.

7 CONCLUSION

In this study, we proposed a path planning algorithm that adapts to the face direction of pedestrians and safely avoids pedestrians who are walking while on their smartphones.

Our subjective evaluation results suggest that there are relationships between the evaluation values and the physical indicators.

This method would enable the operation of advanced collaboration between pedestrians and autonomous mobility on campus.

REFERENCES

[1] D. Fox, W. Burgard, and S. Thrun, "The dynamic window aproach to collision avoidance," IEEE Robot. Autom. Mag., Vol. 4, No. 1, pp. 23-33,(1997).

[2] S. M. LaValle, "Rapidly-Exploring Random Trees: A New Tool for Path Planning," Tech. Rep, (1998).

[3] T. Goto, "Pedestrian behavior prediction and forecast circle generation using Kalman filter"(2019).

[4] H. Hisahara et al, "Human Avoidance Function for Robotic Vacuum Cleaner Through Use of Environmental Sensors -Roomba® Making Way for Humans" , IEEE, Fifth International Conference on Intelligent Systems, Modelling and Simula-tion, pp.64-67.(2014).

[5] D. Zhang, Z. Xie et al,"Real-Time Navigation in Dynamic Human Environments Using Optimal Reciprocal Collision Avoidance", IEEE International Conference on Mechatronics and Automation.(2015).

[6] J. Redmon et al., "You Only Look Once: Unified, Real-Time Object Detection", 2016 IEEE Conference on Compute Vision and Pattern Recognition (2016).

[7] "Plan Mobile Robot Paths Using RRT", https://jp.mathworks.com/help/nav/ug/plan-mobile-robot-paths-using-rrt.html.

[8] C. Shibata et al, "Driver characteristic estimation using vehicle behavior fata while preceding vehicle decelerating", Information Processing Society of Japan (2016).

[9] D. Saito " Change in effective visual field using smartphone with walking" , Biomedical Fuzzy System Association (2019).

(Received: October 31, 2022)
(Accepted: June 4, 2023)

Yuto Yada
He was graduated from school of Vehicle System Engineering, Kanagawa Institute of Technology as a bachelor. Also, he is a master student at Graduate School of Mechanical system engineering, Kanagawa Institute of Technology, Japan. His expertise is development of intelligent mobility.

Shunsuke Michita
He was graduated from school of Vehicle System Engineering, Kanagawa Institute of Technology as a bachelor. Also, he earned his master's degree in the Department of Mechanical System Engineering from the Kanagawa Institute of Technology. He is currently employed at Honda Motor Co., Ltd.

Seiji Komiya
He was graduated from the Graduate School of Engineering, Yokohama National University in March 1990. Joined Kanagawa Institute of Technology, where he currently works. His expertise is the development of intelligent mobility.

Toshihiro Wakita received B.E. from Kyoto University, M.S. from The University of Tokyo and Ph.D. degree from Nagoya University in 1983, 1985 and 2006, respectively. He is currently a professor at Kanagawa Institute of Technology. His research interest includes intelligent mobility and human machine interface. He is a member of IEEE, JSAE, IEICE, and IPSJ.

Submission Guidance

About IJIS

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:

Internet of Things (IoT)	Intelligent Transportation System
Smart Cities, Communities, and Spaces	Distributed Computing
Big Data, Artificial Intelligence, and Data Science	Multi-media communication
Network Systems and Protocols	Information systems
Computer Supported Cooperative Work and Groupware	Mobile computing
Security and Privacy in Information Systems	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^{TM}

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