

Exploring Context-Aware User Interfaces for Smartphone-Smartwatch Cross-Device Interaction

YUKI KUBO, University of Tsukuba
RYOSUKE TAKADA, University of Tsukuba
BUNTAROU SHIZUKI, University of Tsukuba
SHIN TAKAHASHI, University of Tsukuba

In this study, we explore context-aware cross-device interactions between a smartphone and smartwatch. We present 24 contexts, and then examine and prioritize suitable user interfaces (UIs) for each. In addition, we present example applications, including a map, notification management system, multitasking application, music player, and video chat application, each of which has its own context-aware UIs. To support these context-aware UIs, we investigate the performance of our context recognizer in which recognition is based on machine-learning using the accelerometers in a smartphone and smartwatch. We conduct seven different evaluations using four machine-learning algorithms: J48 decision tree, sequential minimal optimization (SMO)-based support vector machine (SVM), random forest, and multilayer perceptron. With each algorithm, we conduct a long-interval experiment to examine the level of accuracy at which each context is recognized using data previously collected for training. The results show that SMO-based SVM is suitable for recognizing the 24 contexts considered in this study.

CCS Concepts: • **Human-centered computing** → **Contextual design**; *Activity centered design*; *Mobile devices*;

Additional Key Words and Phrases: Multi-devices, wearable, mobile, context-aware computing, interaction design

ACM Reference format:

Yuki Kubo, Ryosuke Takada, Buntarou Shizuki, and Shin Takahashi. 2017. Exploring Context-Aware User Interfaces for Smartphone-Smartwatch Cross-Device Interaction. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 1, 3, Article 69 (September 2017), 22 pages.
<https://doi.org/10.1145/3130934>

1 INTRODUCTION

Although many people tend to carry both a smartphone and smartwatch, as smartwatches become increasingly popular, many situations exist in which these devices are not readily usable in mobile computing environments. For example, a person often keeps a smartphone in a pocket. In addition, if a user is wearing a smartwatch on the same arm he uses to hold a child's hand while walking down a busy street, he or she cannot see the

This work was partially supported by Japan Science and Technology Agency (JST) ACT-I Grant Number JPMJPR16UA, Japan and a research granted from The Murata Science Foundation.

Author's address: Yuki Kubo, 1-1-1 Tennodai, Tsukuba, Ibaraki, Japan; email: kubo@iplab.cs.tsukuba.ac.jp; Ryosuke Takada, 1-1-1 Tennodai, Tsukuba, Ibaraki, Japan; email: rtakada@iplab.cs.tsukuba.ac.jp; Buntarou Shizuki, 1-1-1 Tennodai, Tsukuba, Ibaraki, Japan; email: shizuki@cs.tsukuba.ac.jp; Shin Takahashi, 1-1-1 Tennodai, Tsukuba, Ibaraki, Japan; email: shin@cs.tsukuba.ac.jp.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

© 2017 Association for Computing Machinery.

2474-9567/2017/9-ART69 \$15.00

<https://doi.org/10.1145/3130934>

smartwatch screen. These examples suggest that the usability and design of mobile computing environments depend considerably on the context in which these devices are being used.

Fortunately, carrying both devices implies that numerous sensors and considerable computational power are always available. Thus, contexts can be recognized by analyzing the data from the sensors. Such contexts include the manner of the user's gripping on a smartphone and arm posture, as well as activeness. In addition, as described in [8, 29], contexts can be used to enhance human-computer interaction. Furthermore, carrying both devices implies that rich information channels are available, including two touchscreens and two vibration motors. Therefore, providing an application user interface (UI) that is suitable for each context that is recognized by the devices is now possible by synthesizing the layouts, feedback, and input methods (i.e., cross-device interactions) of these devices.

In our study, we explore the possibilities of such context-aware cross-device interactions between a smartphone and smartwatch. We previously demonstrated that many contexts, all different but each including the manner of the user's gripping on a smartphone (*grip*), his or her arm posture (*arm*), and user's activeness (*activeness*), can be recognized by a context recognizer which is based on machine-learning that uses built-in accelerometers in each device [18]. Note that we use accelerometers only to explore the simplest form of context recognition. To demonstrate context-aware cross-device interactions, we also designed SynCro, a context-aware UI system that consists of a smartphone and smartwatch. SynCro provides a user with UIs suitable for different contexts. It synthesizes the layouts, feedback, and input methods of these devices according to the context recognized by these devices (e.g., the map application shown in Fig. 1).

In this study, we further explore context-aware cross-device interactions between a smartphone and smartwatch. To this end, we investigate UIs suitable for each context by examining available UIs and prioritizing them under each context. To demonstrate this investigation, we implement example applications, each of which has its own context-aware UIs. To support these context-aware UIs, we investigate the performance of the context recognizer, in which recognition is based on machine learning using the accelerometers of the smartphone and smartwatch. In our investigation, we use both the same and different user data for training. Furthermore, we determine whether a context recognizer can recognize contexts using data previously collected for training. In these investigations, we tested the following four machine-learning algorithms: J48 decision tree (J48), sequential minimal optimization (SMO)-based support vector machine (SVM), random forest (RF), and multilayer perceptron (MLP).



Fig. 1. Map application that demonstrates SynCro's usability. (a) User can use the wide screen of the smartphone to browse the map and select a destination using his or her thumb. (b) After the destination is selected, the smartwatch displays the distance and an arrow to indicate the direction to the destination, which allows for easy viewing while walking. (c) The smartwatch screen is mirrored on the smartphone when the user lowers his or her left arm. The user can then zoom in and out by using a wrist-tilt gesture. The user can use his or her right hand to browse the map again easily and confirm that he or she is walking the proper route.

2 RELATED WORK

We review previous studies on context-sensing and cross-device interaction techniques, as well as context-aware UIs.

2.1 Context Sensing

In our study, we consider contexts consisting of three factors: the manner of the user's gripping on a smartphone, his or her arm posture, and user's activeness. Several studies focused on developing methods that recognize these factors.

Various methods of grip sensing have been previously proposed. GripSense [9] recognizes the manner of gripping as well as the touch pressure using a position of the user's touch on the touchscreen, the built-in inertial sensors, and actuators. Park and Ogawa [27] recognized grip postures on a smartphone by its built-in accelerometer, gyroscope, and touchscreen. HandSense [41] detects six grip styles by utilizing capacitive sensors on opposite sides of a device. Touché [34] recognizes complex configurations of a hand touching an object, including the touch and grip style, by attaching a single capacitive electrode to the object. This method analyzed the signal in the frequency domain. Similar recognition was achieved by vibration in [26]. In a manner similar to GripSense, our recognizer recognizes the manner of the user's gripping on a smartphone using built-in accelerometers of a smartphone and smartwatch.

Previous research studied recognition of user posture using built-in or embedded sensors. For example, Lee et al. [20] proposed a posture monitoring system using the built-in front camera and accelerometer of a smartphone. Mutlu et al. [24] proposed a system that can recognize seated postures by installing pressure sensors in a chair. Nekoze [39] recognizes user's head postures to detect poor user head postures using built-in electrodes and inertial motion sensors in a smartglass. Liu et al. [23] proposed a system that recognizes driving postures by using built-in accelerometers, magnetometers, and gyroscopes of both a smartphone and smartwatch mounted inside a car. Shen et al. [38] proposed a system that tracks 3D arm postures of a user by using a smartwatch. Our recognizer also recognizes the manner of the user's gripping on a smartphone, user postures, and user's activeness by using a smartphone and smartwatch. Our recognizer shows that fusing the sensor data of each device has a potential to recognize many contexts. In addition, a long-interval experiment is performed to examine whether our recognizer can recognize contexts using data previously collected for training.

Several user activity and/or activeness recognition methods have been previously proposed, including those that utilize the following: a smartphone's accelerometer, gyroscope, magnetometer, microphone, and proximity sensors [7]; the built-in accelerometer and microphone in a smartwatch [21]; the built-in nine-axis inertial measurement unit in a smartwatch [30]; and the built-in sensors of both a smartphone and smartwatch [23]. Moreover, Bao et al. [2] developed and evaluated algorithms for physical activity recognition using five accelerometers worn on different parts of the body. puffMarker [33] recognizes the first lapse during the act of smoking by using the embedded accelerometers and gyroscopes in a smartwatch. Thomaz et al. [40] proposed a technique that recognizes a user's eating movements using the built-in accelerometer in a smartwatch. Goto et al. [10] proposed a technique that recognizes user activities, such as walking, stopping, and travelling on a train, by utilizing the built-in GPS and accelerometer in a smartphone. Kwapisz et al. [19] proposed a method that recognizes six user activities by using the built-in accelerometer of a smartphone when the device is stored in a pocket. These activities include walking, jogging, ascending and descending stairs, sitting, and standing. Park et al. [28] proposed a method that uses devices, such as smartphones and tablets to classify device poses (i.e., the device positions relative to the body) and to estimate a user's walking speed. In our study, the built-in accelerometers of a smartphone and smartwatch are used to recognize user postures.

2.2 Cross-Device Interaction

Several studies explored cross-device interaction between two or more devices. For example, in Pick-and-Drop [32], a user uses a pen to transfer data between multiple displays. Yoon et al. [43] proposed a cross-device interaction that combines grip and micromobility when using a tablet. Schmidt et al. [37] proposed a cross-device interaction style that utilizes a smartphone as a tangible input tool to a large display. Hinckley et al. [15] explored a technique that combines grip and motion sensing on a pen and tablet. By contrast, our cross-device interaction uses a smartphone and smartwatch.

Some studies explored cross-device interaction between a smartphone and smartwatch. Duet [5], which was the study that most inspired ours, also uses a smartphone and smartwatch. In [5], the smartwatch is used as a tool palette and serves as a sub-display showing clipboard content on a smartphone. From a map on the smartphone, a location can be zoomed-out by bumping the smartphone twice on the smartwatch. TakeOut [25] is a drawing application that uses a smartphone and smartwatch as a canvas and palette, respectively. WhichHand [22] detects the hand that holds the smartphone (i.e., recognizing two contexts) and provides layouts for one-handed applications suitable for the holding hand. In our study, we explore context-aware cross-device interactions between a smartphone and smartwatch.

2.3 Context-Aware UI

Some studies have shown the potential of context sensing to improve interactions. Schilit et al. [35] showed that providing an optimal UI based on a given situation is possible. For example, proximate selection is a UI technique that makes selection easier based on a user's location information. Schmidt et al. [36] developed a context-aware system using a layered architecture based on sensors, and showed that this system enhances applications by sensing a mobile phone's context. Hinckley et al. [16] proposed a context-aware mobile interaction using several sensors. Specially, their system rotates the UI to conform to the device's orientation (i.e., portrait or landscape). Yang et al. [42] changed the function set on a smartwatch based on hand posture that was recognized by electromyographic (EMG) sensors attached to the arm. iGrasp [6] changes the keyboard layout based on grip recognition using a device case with embedded capacitive touch sensors. Mo-Bi [17] uses bimanual hand postures recognized using accelerometers of a smartphone and two wrist-worn devices (one on each wrist), to make interface layouts and functions of applications suitable to different postures. By contrast, our work provides UIs for both the smartphone and the smartwatch suitable for contexts on a smartphone-smartwatch cross-device interaction.

3 INTERACTION DESIGN

SynCro recognizes contexts and provides a user with UIs suitable to those contexts. In this section, we describe the different contexts that SynCro recognizes and the UIs for cross-device interactions based on the recognized contexts.

3.1 Contexts

The contexts considered in this study are shown in Fig. 2. The assumption is that the user wears the smartwatch on the left wrist. Among the many contexts, $a - j$ can occur while walking. Therefore, we define these as different contexts, $a' - j'$. By contrast, $k - n$ can occur only while resting in a seated position.

These contexts consist of the following three factors: *grip*, *arm*, and *activeness*. The *grip* factor has five levels as shown in Fig. 3a, including left-hand, right-hand, and both-hands. In addition, it has the following two levels relating to the smartphone: in-a-pocket and on-a-desk. We include these two levels as part of the *grip* factor for the sake of convenience, even though these mean the smartphone is not being hand-held by the user. The *arm* factor has four levels as shown in Fig. 3b. The *activeness* factor has two levels: resting and walking.

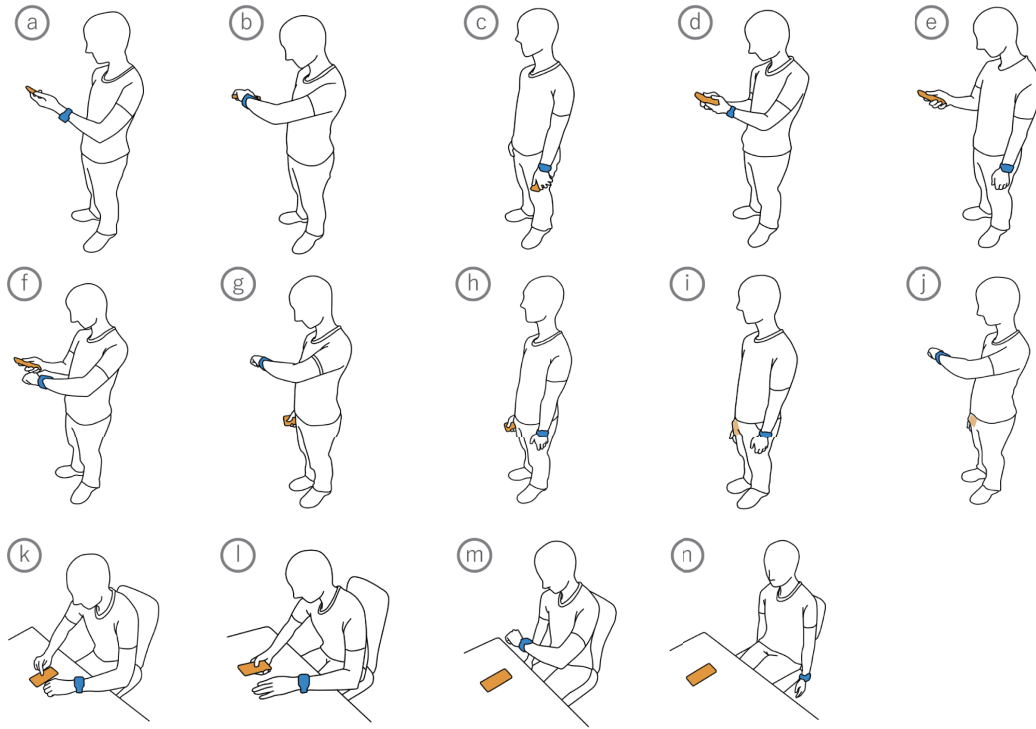
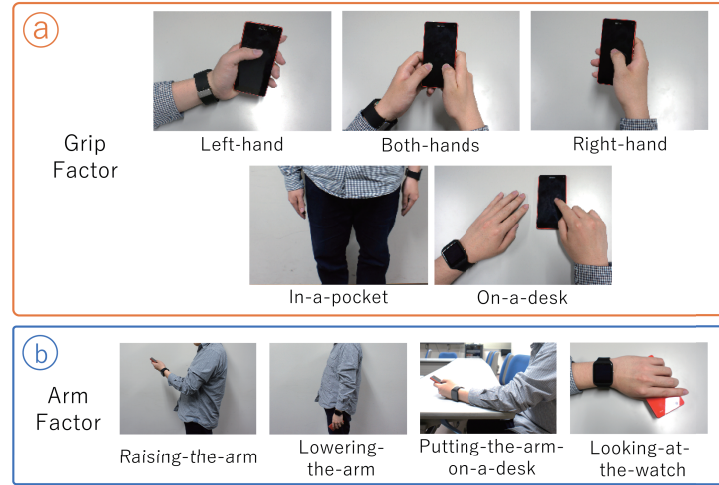


Fig. 2. Contexts considered in this study. As an example scenario, each of the contexts can use a) a smartphone for notification/input because the user is using it; b) a smartwatch mainly for notification/input because the user is looking at it, and a smartphone for simple actions such as stopping an alarm by tapping; c) a notification that can be reviewed on the smartphone’s larger display; d) a smartphone mainly for notification/input because the user is using it; e) a smartphone for notification/input because the user is using it, and a smartwatch for simple actions such as zooming in and out on a map using a wrist-tilt gesture; f) both smartphone and smartwatch for notification/input because the user is using both devices simultaneously; g) a smartwatch for notification/input because the user is looking at it, and a smartphone for simple actions such as stopping an alarm by tapping the display with the right hand; h) a smartphone for notification, that can be reviewed on the smartphone’s larger display; i) smartwatch for notification that is more instantly accessible than the smartphone in the pocket; j) a smartwatch for notification/input because the user is using it; k) both smartphone and smartwatch for notification/input because the user is using both devices; l) a smartphone mainly for input because the user is touching it, and a smartwatch for notification so as not to disturb the user operating the smartphone; m) smartwatch for notification/input because the user is looking at it; n) smartwatch for notification/input.

Relations between *grip* and *arm* factors and the recognized contexts utilizing the two devices are shown in Tables 1 and 2 for resting and walking, respectively. Note that our recognizer can recognize the *arm* factor of the right hand only when the smartphone is gripped in the right-hand. In these tables, N/A denotes contexts that do not apply. For example, under both-hands, lowering-the-hand, putting-the-arm-on-the-desk, and looking-at-the-watch contexts do not apply. In addition, we combined similar contexts; thus, one ID appears two or more times in these tables. Specifically, we combined raising-the-arm and looking-at-the-watch of the left-hand for the case in which the smartphone is not gripped in the left hand. This is because raising the left arm is a preliminary movement of looking-at-the-watch. Furthermore, “–” in Table 1 denotes contexts that our recognizer does not

Fig. 3. Two of the three factors of contexts: (a) *grip* and (b) *arm*.

attempt to recognize (Note that we erroneously thought that putting a smartphone in-a-pocket could occur only under a-j and a'-j').

3.2 UIs for Cross-device Interaction between Smartphone and Smartwatch

3.2.1 Mirroring. A mirroring function (Fig. 4a, b) can be used under a, d, and e where a user cannot look at the smartwatch screen. This function shows the contents of the smartwatch screen on an area in the smartphone screen (represented as a blue square in Fig. 4b). The position of the mirrored area can be adjusted by the user by dragging it with a finger. This smartphone – smartwatch cross-device interaction allows the smartwatch to be used even when the user cannot raise his or her left arm (e.g., when in a crowded place or when a user carries a bag in his or her left hand).

Table 1. Relations between the grip and arm factor and contexts while resting.

	Grip Factor					
	Left Arm	Left-hand	Both-hands	Right-hand		In-a-Pocket
				Raising-the-arm	Lowering-the-arm	
Raising-the-arm		(a)	(d)	(f)	(g)	(j)
Lowering-the-arm		(c)	N / A	(e)	(h)	(i)
Putting-the-arm-on-the-desk		(a)	N / A	(l)	(g)	—
Looking-at-the-watch		(b)	N / A	(f)	(g)	(j)

Table 2. Relations between the grip and arm factor and contexts while walking.

	Grip Factor				
	Left Arm	Left-hand	Both-hands	Right-hand	
				Raising-the-arm	Lowering-the-arm
Raising-the-arm		(a')	(d')	(f')	(g')
Lowering-the-arm		(c')	N / A	(e')	(h')
Looking-at-the-watch		(b')	N / A	(f')	(g')

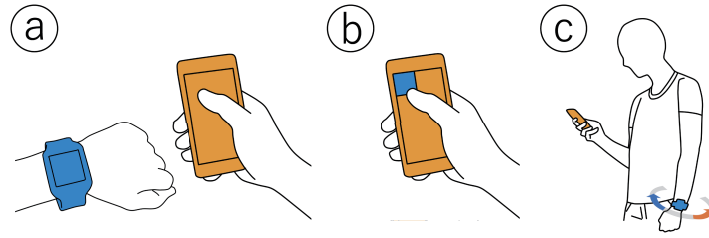


Fig. 4. Examples of context-aware UIs. (a), (b): Mirroring. (c) Wrist-tilt gesture.

3.2.2 Wrist-tilt Gesture. A wrist-tilt gesture [31] can be used under c, e, h, i, n when the user lowers his or her left arm (Fig. 4c). The main input method for the smartwatch is touching the screen. However, this requires that the user raise the left arm. Moreover, lowering the left arm forces the user to perform one-handed thumb interactions [3, 13, 14]. Therefore, we designed a smartphone – smartwatch cross-device interaction in which the smartwatch serves as a controller to operate the smartphone. This offers the user another input channel to adjust a parameter continually on the smartphone in a comfortable posture without having to look at it. Similarly, the wrist-tilt gesture can be adopted under b, f, g, j, and m when the user raises the left arm, although this slightly lowers the usability of the smartwatch.

3.3 UIs suitable for Different Contexts

We designed seven UIs and examined whether a user can use the UI in each context (Table 3). The seven UIs are right-handed input with a smartphone (RH), left-handed input with a smartphone (LH), both-handed input with a smartphone (BH), left wrist-tilt gesture (TG), looking at a smartphone (LP), looking at a smartwatch (LW), and input with a smartwatch touchscreen (IW). In this table, the symbol “✓” denotes UIs that the user can use in the different contexts, and the symbol “–” denotes unusable UIs. In this table, we also use L^1 (i.e., limited) to denote when the system cannot recognize the movement of the right hand. In addition, L^2 indicates that the user can only perform simple actions (e.g., tap the smartphone with the right or left hand).

3.4 Priority of UIs in Each Context






















For each context, we used Table 4 to determine the priority of UIs with respect to the following three categories.

Screen We determine the screen to be used in a context based on LP and LW in Table 3. If both LP and LW are ✓ in a context, we assigned the smartphone as the main screen and the smartwatch as the sub screen. If either LP or LW is ✓, we use ✓ to establish the device as the main screen. If both LP and LW are “–”, we do not assign any device as the main or sub screen.

Input Method We determine the input methods to be used in a context based on the screen and RH, LH, BH, and IW in Table 3. If the main screen is the smartphone, we assign the input method (✓) in Table 3 as the main input. If only the smartwatch can be used as the screen, we assign TG as the main input because the user can perform TG instantly.

Feedback We determine the feedback to be used in a context based on the screen. We assign the main screen as the main feedback. If no main screen exists in a specific context, we assign the smartwatch as the main feedback.

Table 3. UIs suitable for each context. Here, “✓” denotes UIs that the user can use in that context, “–” refers to unusable UIs, L^1 (i.e., limited) means that the system cannot recognize the movement of the right hand, and L^2 means that the user can perform only simple actions (e.g., tap the smartphone with the right or left hand). For example, under d, the user can operate a smartphone with his or her both hands, and look at only the smartphone. In case of j, the user cannot operate a smartphone. However, the user can perform a wrist-tilt gesture and look at the smartwatch.

Context	RH	LH	BH	TG	LP	LW	IW	Context	RH	LH	BH	TG	LP	LW	IW
															
a 	L^1	✓	–	–	✓	–	–	h 	L^2	–	–	✓	–	–	–
b 	–	L^2	–	✓	–	✓	✓	i 	–	–	–	✓	–	–	–
c 	–	L^2	–	✓	–	–	–	j 	–	–	–	✓	–	✓	✓
d 	✓	✓	✓	–	✓	–	–	k 	✓	✓	✓	–	✓	✓	✓
e 	✓	–	–	✓	✓	–	–	l 	✓	–	–	–	✓	✓	–
f 	✓	–	–	✓	✓	✓	–	m 	–	–	–	✓	✓	✓	✓
g 	L^2	–	–	✓	–	✓	–	n 	–	–	–	✓	✓	–	–

3.5 Example Applications

The following applications were used to demonstrate context-aware UI systems with a smartphone and smartwatch.

3.5.1 Map. In the smartphone-smartwatch cross-device interaction described in [5], a smartwatch serves as a sub-display showing a map application. We extended this application so that it changes the UIs automatically for different contexts (Fig. 1). Fig. 1a shows the map UI when a user raises both hands (f, k). In this UI, the user can use the wide screen of the smartphone to browse the map and select a destination with his or her thumb. After selecting a destination, the smartwatch displays the distance and an arrow to indicate the direction to the destination (Fig. 1b). This allows the user to view this information easily while walking (b, g, j, b', g', j'). The smartwatch screen is mirrored on the smartphone when the user lowers his or her left arm (Fig. 1c). In this UI, zooming in and out by using one hand (e) is difficult. Therefore, we designed the UI so user can zoom in and out using a wrist-tilt gesture with the left arm while pushing a button on the smartphone. This enables the user to browse the map easily with his or her right hand to confirm that he or she is walking the proper route to the destination.

3.5.2 Notification Management System. We implemented a smartphone-smartwatch notification management system that changes notified devices and notification methods (display or vibration) depending on the context (Table 5). In Table 5, a non-vibration indicates that the system displays only the notification and does not vibrate the device. By contrast, a vibration indicates that the system both displays the notification and vibrates the device.

The system displays notifications on both the smartphone (a, d, e) and smartwatch (b, g, j, m) without vibrations when a user operates either device. By contrast, the system displays notifications on the smartphone and vibrates it when a user does not operate, but merely holds the device(c, h). Otherwise, the system displays notifications on the smartwatch (f, i, k, l, m). Furthermore, the system displays all notifications on the smartwatch and vibrates it to prevent manipulation while walking (a'–j').

3.5.3 Multitasking. We implemented a multitasking application. We provide a scenario in which a user uses a music player and web browser simultaneously (Fig. 5). This application includes a button that, when pushed, causes the current application (e.g., the music player) to be mirrored on the smartwatch and operated there. Thus, a user can use another application on the smartphone (e.g., a web browser), as shown in Fig. 5b. Moreover, the application mirrors the smartwatch screen on the smartphone when a user lowers his or her left arm (Fig. 5c).

Table 4. Priority of UIs in each context.

Context	Main screen	Sub screen	Main input	Sub input	Main feedback	Sub feedback
a	Smartphone	–	LH	RH	Smartphone	Smartwatch
b	Smartwatch	–	TG	IW & LH (Tap)	Smartwatch	Smartphone
c	–	–	TG	LH (Tap)	Smartwatch	–
d	Smartphone	–	BH	–	Smartphone	Smartwatch
e	Smartphone	–	RH	TG	Smartphone	Smartwatch
f	Smartphone	Smartwatch	RH	TG	Smartphone	Smartwatch
g	Smartwatch	–	TG	RH (Tap)	Smartwatch	Smartphone
h	–	–	TG	RH (Tap)	Smartwatch	–
i	–	–	TG	–	Smartwatch	–
j	Smartwatch	–	TG	IW	Smartwatch	–
k	Smartphone	Smartwatch	RH	LH & BH	Smartphone	Smartwatch
l	Smartphone	Smartwatch	RH	–	Smartphone	Smartwatch
m	Smartwatch	Smartphone	TG	–	Smartwatch	Smartphone
n	Smartphone	–	TG	–	Smartphone	Smartwatch

Table 5. Notifications.

Contexts	Notified device and vibration
a, d, e	smartphone, non-vibration
b, g, j, m	smartwatch, non-vibration
c, h	smartphone, vibration
f, i, k, l, n, a' – j'	smartwatch, vibration

In the smartwatch screen shown on the smartphone, the user can operate the music player with his or her finger. Thus, the user can continue to multitask even when lowering his or her left arm.

3.5.4 Music Player. We implemented a music player application that selects UIs suitable for different contexts. Fig. 6a shows the UI when a user raises both hands (f, k). On this UI, tapping a track from the music list on the smartphone displays corresponding music information on the smartwatch. Double tapping a track on the smartphone plays the track. After a track is played, the music player displays a different UI (Fig. 6b). On this UI, the smartwatch displays a music player that can be easily controlled, even if the user places his or her smartphone in a pocket (b, g, j, m). When the user lowers his or her left arm, the smartwatch screen is mirrored on the smartphone (Fig. 6c). This application displays yet another UI, as depicted in Fig. 6d. In this UI, even when the user holds luggage in the right hand (b, j), thus preventing him or her from operating the player with the right hand, the user can easily access other tracks using a wrist-tilt gesture with the left arm.

3.5.5 Video Chat. For video chat, the user selects a contact address with the smartwatch, as shown in Fig. 7a. This can be done easily, even if the smartphone is in a pocket (j). After a connection is established, the contact's face is displayed on the smartwatch. Although the user can communicate through video chat on the smartwatch, he or she may want to communicate on the smartphone when its larger display is available (a, d, e, f, k, l, n). At this time, the main screen changes automatically between a smartphone and smartwatch based on the user context, which is recognized through our method. When the user wants to take a note when communicating through the smartphone, he or she can use the smartwatch as a voice recording device (f, k, l, m), as shown in Fig. 7b. When a user is unable to use either device (c, h, i), the application notifies the contact that the user is not looking at the display (Fig. 7c). If the user is walking (a', b', d', e', f', g', j'), the main screen notifies the user to stop (Fig. 7d). In addition, if the user is not looking at the display for a period (c, h, i), the application disconnects from video chat.

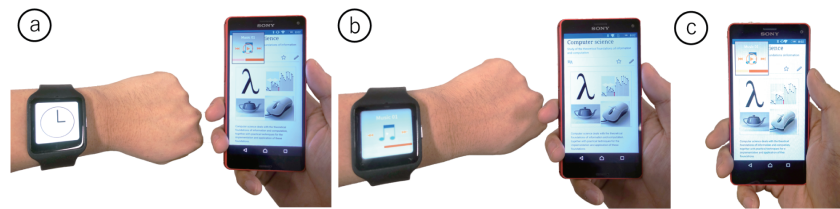


Fig. 5. Multitasking application. (a) The user can transmit the current application to the smartwatch by pushing a button. After transmission, the application can be operated from the smartwatch. (b) The user can use two applications using both devices simultaneously. (c) When the user lowers his or her left arm, the smartwatch screen is mirrored on the smartphone. This allows the user to continue multitasking (i.e., even when lowering the left arm).

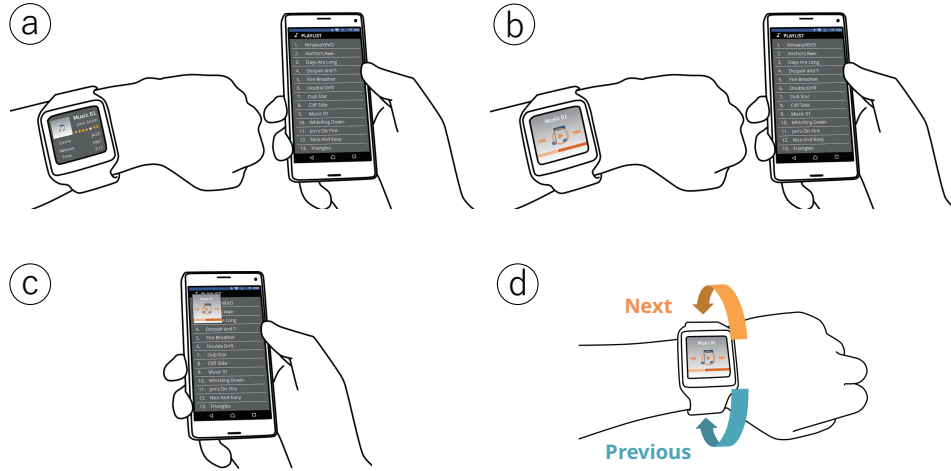


Fig. 6. Music Player. (a) The user can select music by double tapping with a thumb. (b) After a track is selected, the smartwatch displays the player. (c) The smartwatch screen is mirrored on the smartphone when the user lowers his or her left arm. (d) The user can access other tracks easily by using a wrist-tilt gesture with the left arm.

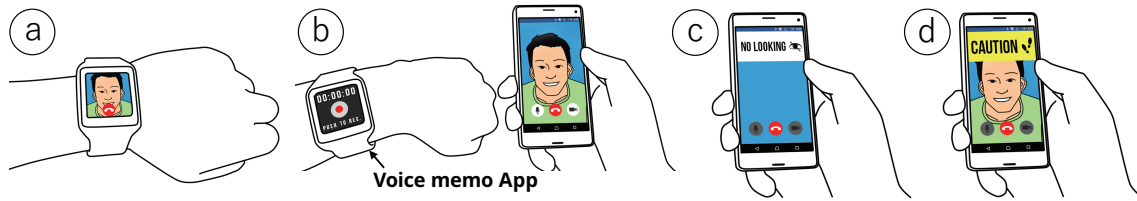


Fig. 7. Video Chat. (a) The user can easily select a contact address using a smartwatch. (b) The user can use the smartwatch as a voice recording device to record a note when using both devices. (c) When the contact is unable to use either device, the application notifies the user that the contact is not looking at the display. (d) If the user is walking, a caution is displayed on the screens of both devices.

4 IMPLEMENTATION

We implemented a context recognizer based on machine learning that uses the accelerometers in both a smartphone and smartwatch.

4.1 Devices and Configuration

We used a SONY Xperia Z3 Compact SO-02G and a SONY SmartWatch 3 SWR50. The smartphone has a quad-core 2.5 GHz processor with 2 GB RAM. The smartwatch has a quad-core 1.2 GHz processor and 512 MB RAM. After the smartwatch was connected to the smartphone through Bluetooth, the smartwatch transmitted frames to the smartphone continuously. Each frame contained a time stamp and 3-axis accelerometer data ($a_{wx}(t)$, $a_{wy}(t)$, $a_{wz}(t)$). The frame rate is 20 Hz. We determined this frame rate empirically so that the smartwatch achieved a stable transmission to the smartphone. The smartphone also stored frames at 20 Hz. Each frame contained a time stamp and 3-axis accelerometer data from the smartphone ($a_{px}(t)$, $a_{py}(t)$, $a_{pz}(t)$). The time stamps

were used to synchronize the frames of both devices because those from the smartwatch arrived irregularly to the smartphone as a result of latency of transmission.

4.2 Context Recognizer

Our context recognizer ran on the smartphone and used J48, SMO-based SVM, RF, and MLP in the WEKA data mining software [11] with a default setting. It collected a series of 32 frames from both the smartphone and smartwatch as a sliding window covering a time span of 1.6 s. Then, it calculated a feature vector for each sliding window. With this feature vector, the recognition result using machine learning was also obtained at 20 Hz.

To recognize the grip and arm factors, the recognizer first calculated $a_p(t)^2$ and $a_w(t)^2$, which are the sum of the squares of acceleration from the smartphone and smartwatch, respectively. Furthermore, the recognizer calculated $a_d(t)^2$, that is, by subtracting $a_p(t)^2$ from $a_w(t)^2$. These values are given as:

$$\begin{aligned} a_p(t)^2 &= a_{px}(t)^2 + a_{py}(t)^2 + a_{pz}(t)^2 \\ a_w(t)^2 &= a_{wx}(t)^2 + a_{wy}(t)^2 + a_{wz}(t)^2 \\ a_d(t)^2 &= a_w(t)^2 - a_p(t)^2 \end{aligned}$$

To recognize user's activeness (i.e., resting or walking), we used a fast Fourier transform (FFT) to calculate the following features:

Frequency Power (FP) [4] We used FFT for each $a_p(t)^2$, $a_w(t)^2$, and $a_d(t)^2$ of a sliding window. Therefore, each sliding window produced 16 powers of frequency; the frequency range was 0–5 Hz with a resolution of 0.31 Hz, because the sliding window consisted of 32 frames and the frame rate was 20 Hz.

Maximum Frequency Power (MFP) MFP in FFT for each $a_p(t)^2$, $a_w(t)^2$, and $a_d(t)^2$.

Frequency of Maximum Frequency Power (FMFP) Frequency that shows the MFP for each $a_p(t)^2$, $a_w(t)^2$, and $a_d(t)^2$.

In addition, the recognizer calculates the following features:

Average Acceleration (AA) [2] AA in a sliding window for each axis of both the smartphone and smartwatch.

Average Difference Acceleration (ADA) Average of $a_d(t)^2$ in a sliding window.

Average Resultant Acceleration (ARA) [1] Average of each $a_p(t)^2$ and $a_w(t)^2$ in a sliding window.

In summary, our feature vector consisted of the following 63 features: 48 FPs, three MPFs, three FMFPs, six AAs, one ADA, and two ARAs.

5 EVALUATION OF ACCURACY FOR CONTEXT RECOGNITION

We conducted a long-interval experiment to examine the level of accuracy of our recognizer in recognizing 24 contexts. Moreover, because we were interested in whether the recognizer could recognize these contexts when using data previously collected for training data, we designed this as a long-interval experiment. Specifically, we collected the data twice (first and second rounds), where the second round followed the first by approximately five months later¹. We designed this five month period so that we could collect data that had little learning effect.

5.1 Participants

Twelve Japanese male volunteers that included nine laboratory students, participated in the experiment. Their ages were in the range of 21 – 24 years ($M = 22.3$) when the first round was conducted. Their experience with smartphones ranged from 0 to 96 months ($M = 50.25$). All participants except one were right-handed and wore

¹The first round was conducted from 2016-04-09 to 2016-04-10; the second round was conducted from 2016-09-01 to 2016-09-11.

a smartwatch on their left wrist. Four participants had previous experience using a smartwatch in the range of 1 – 12 months ($M = 6.0$). Five participants had previously used a wristwatch. One round for each participant required approximately 50 minutes to complete.

5.2 Procedure

We asked participants to perform the actions in the given contexts shown in Fig. 2, and then collected acceleration data from the smartphone and smartwatch. We conducted this experiment in a quiet and closed room for safety.

First, we asked a participant to wear a smartwatch on his left wrist. Subsequently, we displayed the ID (i.e., a-n and a'-j') of one context on the smartphone. These contexts are described in a document containing instructions and pictures, as shown in Fig. 2. When participants performed the actions in the given contexts, we allowed them sufficient time between contexts to avoid one context affecting the next. If the participant failed to perform instructed actions in a context, we asked him to perform it again. When performing a' - j', we asked the participant to walk between two tape markings on the floor: the distance between the marks was 10 m (10.94 yards). Furthermore, if the context allowed the participant to operate the smartphone (e.g., a and d), we asked the participant to operate an image viewer application by swiping or tapping at an interval of approximately 1 s. The participant performed the actions in the given contexts until he finished (i.e., when the data for 100 frames were collected). In both rounds, we used the same chairs and desks to control the experimental conditions. In each round, the participant performed four sessions. In each session, the participant performed the actions in the 24 given contexts once in a randomized order. The participant took a break of five minutes between the two sessions.

In total, 230,400 pieces of data ($2 \text{ rounds} \times 12 \text{ participants} \times 24 \text{ contexts} \times 4 \text{ sessions} \times 100 \text{ frames}$) were collected.

5.3 Results

To understand the performance of our recognizer, particularly to examine whether it could recognize contexts by using data previously collected for training, we performed the following seven types of analyses. These recognition tests were performed using J48, SMO-based SVM, RF, and MLP in WEKA data mining software [11].

5.3.1 Per-User Classifiers. To understand the feasibility of our recognizer for individual users, we assessed the accuracy of per-user classifiers. First, we trained a model using the three sessions of data collected from a single participant in a single round and tested it with the remaining session of data from that same participant. All training and test combinations were tested and the results for each participant were averaged (i.e., four-fold cross validation). We performed this four-fold cross validation using both rounds of data of each participant. These results showed that, for the first-round data, the accuracies of the per-user classifiers for J48, SMO-based SVM, RF, and MLP were 83.9% (SD = 6.9), 94.0% (SD = 4.6), 94.0% (SD = 4.3), and 93.7% (SD = 4.2), respectively. For the second-round data, they were 86.2% (SD = 6.9), 93.2% (SD = 12.0), 93.3% (SD = 9.4), and 93.0% (SD = 13.1), respectively, as shown in Table 6. However, we examined our data and found that the general classifier trained with the second-round data exhibited considerably lower recognition accuracy for one participant. Therefore, we removed that participant's data from the average as outliers. The resulting accuracies were then 87.2% (SD = 5.9), 95.8% (SD = 3.9), 95.1% (SD = 3.3), and 95.5% (SD = 3.3), respectively.

5.3.2 General Classifier. We assessed the accuracy of the general classifier to evaluate the feasibility of the recognizer for "walk up" users [12]. This involved training a model using eleven participants' data and testing that data with those of another participant (i.e., 12-fold participant-independent cross-validation). As shown in Table 8, the results of the first-round data for J48, SMO-based SVM, RF, and MLP show accuracies of 84.9% (SD = 6.1), 88.8% (SD = 2.9), 89.7% (SD = 2.8), and 85.0% (SD = 5.7), respectively. The confusion matrix for the

results when applying J48 is shown in Table 7. Moreover, the accuracies of the second round data were 77.8% (SD = 14.7), 84.0% (SD = 13.6), 84.0% (SD = 12.1), and 78.8% (SD = 14.3), respectively. However, we examined our data and found that the general classifier trained with the second-round data exhibited considerably lower recognition accuracies for one participant. Therefore, we removed that participant's data from the average as outliers. The resulting accuracies were then 81.5% (SD = 7.8), 87.6% (SD = 5.9), 87.3% (SD = 4.3), and 82.4% (SD = 7.2), respectively, as shown in Table 8.

5.3.3 Long-Interval Per-User Classifiers. To examine whether the recognizer could recognize contexts by using data previously collected for training, we first trained a model with the first-round data collected from a single participant and then tested it with the second-round data of that same participant. All training and test combinations were tested and the results for each participant were averaged. The accuracies of the long-interval per-user classifiers for J48, SMO-based SVM, RF, and MLP were 69.4% (SD = 17.5), 84.7% (SD = 13.8), 77.8% (SD = 9.1), and 83.8% (SD = 17.1), respectively (Table 9). However, we found that the recognition accuracy of one participant was 24.9%, which was considerably lower than that of the other participants. After we removed that participant's data from the average, the resulting accuracies were 73.4% (SD = 11.0), 88.5% (SD = 5.1), 79.5% (SD = 7.3), and 88.5% (SD = 5.0), respectively.

5.3.4 Long-Interval General Classifier. We assessed the accuracy of the long-interval general classifier to evaluate the feasibility of the recognizer for “walk up” users over a long-interval. We trained a model using all participants' data from the first-round and tested that data with those of the second single participant's data (all combinations). The average accuracies of the long-interval general classifier for J48, SMO-based SVM, RF, and MLP were 71.5%, 85.1%, 82.2%, and 82.9%, respectively (Table 9).

5.3.5 Volume of Training Data. We examined the amount of training data required to stabilize the recognition accuracy. To this end, we trained models incrementally with the data of a greater number of participants and tested them with the remaining data. For each $n = \{1, \dots, 11\}$, we first selected n sets of participants' data randomly from 12 participants' data and tested the remaining participants' data. We conducted this training/test set 12 times for each n by using the first-round data, and the results were averaged. We show the average recognition accuracy for each number of participants in the training data in Fig. 8. The figure shows that the accuracy stabilized with eight participants (i.e., 76,800 sets of data) in J48, SMO-based SVM, and RF and four sets of training data (i.e., 38,400 sets of data) in MLP. Therefore, these results suggest that the data of eight participants were sufficient to train the recognizer.

5.3.6 Recognition Accuracy when using Only One Device. We discussed the method of detecting contexts by using a combination of smartphone and smartwatch. To test the hypothesis that using both devices simultaneously improves recognition accuracy, we analyzed the accuracy when using the data from either a smartphone or smartwatch data. We selected the first-round data for testing because its data recognition accuracy was better than that of the second round. In addition, we analyzed the accuracy of the per-user classifiers.

Table 6. Accuracy of per-user classifiers with the first and second round data. Values in parentheses are SD.

Classifier	first round	second round	second round (except P2)
J48	83.9 (6.9)	86.2 (6.9)	87.2 (5.9)
SMO-based SVM	94.0 (4.6)	93.2 (12.0)	95.8 (3.9)
RF	94.0 (4.3)	93.3 (9.4)	95.1 (3.3)
MLP	93.7 (4.2)	93.0 (13.1)	95.5 (3.3)

Table 7. Confusion matrix of the general classifier with the first-round data.

	Predicted Contexts																					
	a	b	c	d	e	f	g	h	i	j	k	l	m	n	a'	b'	c'	d'	e'	f'	g'	h'
Actual Contexts	a	74.4		25.0																		
	b		97.8																			
	c			84.5					15.2													
	d	43.7			48.0	0.2					8.1											
	e				98.5				0.2													
	f					84.3	2.1				10.1											
	g			6.3			81.7			5.7			6.3									
	h							93.9	0.6					5.5								
	i		9.6		8.3			2.1	77.1	2.1												
	j			0.4			10.6		6.2	75.6												7.1
	k					2.1					89.4	0.1	8.4									
	l					9.0						91.0										
	m										2.1		97.9									
	n													100.0								
	a'	0.4			0.1										49.4	1.6		48.6				
	b'	0.2	0.9												1.9	95.8		1.2				
	c'			2.8					0.2								93.7					3.3
	d'	1.0			0.3										35.3	0.1		55.1		8.3		
	e'					0.1													99.9			
	f'						0.3					0.6							99.1			
	g'										0.9		4.1					4.6		88.7		
	h'							0.5						6.2							90.9	2.4
	i'								0.6								8.0		4.2		0.1	83.7
	j'						0.1		0.2	0.7								3.3		1.5	1.9	92.4

Table 8. Accuracy of the general classifiers with the first- and second-round data. Values in parentheses are SD.

Classifier	first round	second round	second round (except P2)
J48	84.9 (6.4)	77.8 (14.7)	81.5 (7.8)
SMO-based SVM	88.8 (2.9)	84.0 (13.6)	87.6 (5.9)
RF	89.7 (2.8)	84.0 (12.1)	87.3 (4.3)
MLP	85.0 (5.7)	78.8 (14.3)	82.4 (7.2)

Table 9. Accuracy of the general classifiers with the first- and second-round data. Values in parentheses are SD.

Classifier	long-interval per-user	long-interval per-user (except P2)	long-interval general
J48	69.4 (17.5)	73.4 (11.0)	71.5
SMO-based SVM	84.7 (13.8)	88.5 (5.1)	85.1
RF	77.8 (9.1)	79.5 (7.3)	82.2
MLP	83.8 (17.1)	88.5 (5.0)	82.9

Using only Smartphone Data We used the following 22 features from the smartphone data: 16 FPs, one MFP, one FMFP, three AAs, and one ARA. The accuracies of the first per-user classifiers for J48, SMO-based SVM, RF, and MLP were 50.7% (SD = 8.8), 54.2% (SD = 7.7), 53.2% (SD = 9.1), and 52.2% (SD = 8.4), respectively, as shown in Table 10.

Using only Smartwatch Data We used the following 22 features from the smartwatch data: 16 FPs, one MFP, one FMFP, three AAs, and one ARA. The accuracies of the first per-user classifiers for J48, SMO-based SVM, RF, and MLP were 44.6% (SD = 8.7), 48.6% (SD = 7.9), 49.7% (SD = 8.3), and 49.7% (SD = 8.6), respectively, as shown in Table 10.

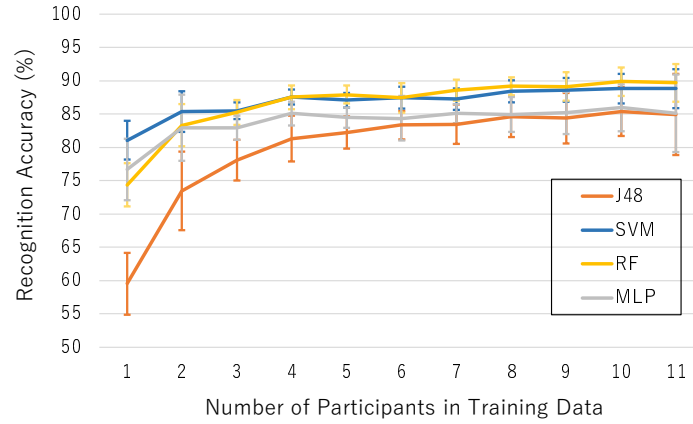


Fig. 8. Volume of training data and accuracy.

Table 10. Accuracy of per-user classifiers with the smartphone and smartwatch data.

Classifier	Only Smartphone	Only Smartwatch	Both Devices
J48	50.7 (8.8)	44.6 (8.7)	83.9 (6.9)
SMO-based SVM	54.2 (7.7)	48.6 (7.9)	94.0 (4.6)
RF	53.2 (9.1)	<u>49.7</u> (8.3)	94.0 (4.3)
MLP	52.2 (8.4)	<u>49.7</u> (8.6)	93.0 (13.1)

5.3.7 Important Features. We analyzed important features in RF for recognition by using WEKA data mining software [11]. We selected the first-round data for testing because its data recognition accuracy was better than that of the second-round. In addition, we analyzed the accuracy of the general classifier. These results are shown in Table 11. In this table, {Phone/Watch/diff}FFT n is the FP at $n \times 0.31$ Hz for each $a_p(t)^2$, $a_w(t)^2$, and $a_d(t)^2$. {Phone/Watch}Accel{X/Y/Z} is the AA of each axis of the smartphone/smartwatch; DiffAccel is the ADA; max{Phone/Watch/Diff}FFT n is the MFP for each $a_p(t)^2$, $a_w(t)^2$, and $a_d(t)^2$; max{Phone/Watch/Diff}FFT is the FMFP for each $a_p(t)^2$, $a_w(t)^2$, and $a_d(t)^2$. As Table 11 shows, the most important feature was PhoneFFT2, which is the FP of the smartphone at 0.62 Hz (i.e., 2×0.31 Hz = 0.62 Hz).

5.4 Discussion of Results

The per-user classifiers achieved over 90% recognition accuracy in both rounds, with the exception of J48. The general classifiers achieved 80% recognition accuracy in both rounds with SMO-based SVM and RF. In addition, the long-interval per-user and general classifiers achieved recognition accuracies of over 80% with SMO-based SVM and MLP. However, the recognition accuracy with RF was less than 80%. These results indicate that SMO-based SVM is suitable for recognizing the 24 contexts considered in this study.

As shown in Tables 6, 8, and 9, the accuracy of the second-round improved by removing participant data with the lowest accuracy. We analyzed the outlier participant's data and showed that the recognition accuracy using data from a particular session is the lowest for per-user classifiers. This may have been because the smartwatch belt was loose on that participant (due to a slender wrist). However, the classifiers trained with the second-round data without those of the outlier participant still tended to show lower accuracies than data from the

Table 11. Averages of mean decrease impurity (MDI) in the first general classifier (higher is better in MDI).

#	Feature	MDI	#	Feature	MDI	#	Feature	MDI
1	PhoneFFT2	0.538	22	WatchAccelX	0.413	43	diffAccel	0.340
2	PhoneFFT3	0.520	23	WatchAccelY	0.413	44	diffFFT3	0.338
3	PhoneAccelX	0.515	24	WatchFFT1	0.410	45	diffFFT4	0.330
4	PhoneFFT1	0.501	25	WatchFFT3	0.408	46	phoneAccelSum	0.328
5	PhoneFFT4	0.499	26	WatchAccelZ	0.403	47	maxDiffFFTHz	0.327
6	PhoneFFT6	0.492	27	WatchFFT4	0.399	48	diffFFT8	0.324
7	PhoneFFT5	0.489	28	WatchFFT6	0.392	49	diffFFT6	0.323
8	PhoneFFT7	0.489	29	WatchFFT5	0.391	50	diffFFT5	0.320
9	PhoneFFT8	0.489	30	WatchFFT7	0.385	51	diffFFT10	0.320
10	PhoneFFT9	0.476	31	WatchFFT10	0.372	52	diffFFT7	0.319
11	PhoneAccelY	0.468	32	WatchFFT8	0.371	53	watchAccelSum	0.318
12	PhoneFFT10	0.467	33	WatchFFT11	0.369	54	diffFFT9	0.315
13	PhoneFFT11	0.458	34	WatchFFT9	0.367	55	diffFFT11	0.311
14	PhoneAccelZ	0.453	35	WatchFFT12	0.366	56	diffFFT12	0.308
15	PhoneFFT12	0.449	36	WatchFFT13	0.365	57	diffFFT13	0.306
16	PhoneFFT13	0.444	37	maxWatchFFTHz	0.363	58	diffFFT15	0.305
17	PhoneFFT14	0.433	38	WatchFFT14	0.358	59	diffFFT14	0.303
18	PhoneFFT15	0.425	39	diffFFT1	0.356	60	diffFFT16	0.299
19	maxPhoneFFTHz	0.422	40	WatchFFT15	0.349	61	maxDiffFFT	0.283
20	PhoneFFT16	0.419	41	WatchFFT16	0.347	62	maxPhoneFFT	0.000
21	WatchFFT2	0.415	42	diffFFT2	0.344	63	maxWatchFFT	0.000

first-round data. Therefore, the quality of the second-round data was found to be worse than that of the first-round. A possible reason was traffic congestion with wireless communication. We conducted the first-round on a Saturday and Sunday. The second round included weekdays. Therefore, the transmission rate from smartwatch to smartphone could be low. To address this problem, we plan to modify our implementation such that all packets from the smartwatch are time-stamped so that the system can synchronize the accelerometer data of both the smartphone and smartwatch.

The accuracy when using both devices was more than 30% higher than that when using only one device, as shown in 10. Therefore, numerous contexts could be recognized by using both devices.

As shown in Table 11, the features we used contributed to recognition, with the exception of max{Phone/Watch} FFT. In this experiment, the MDIs of the max{Phone/Watch}FFT were 0.000 because the frequency of the maximum power spectrum was always 0 Hz. Therefore, we will modify our feature vector to use the frequencies that show the second and third maximum power spectrum instead (max{Phone/Watch}FFT) in future studies.

6 DISCUSSION AND FUTURE WORK

The results of our experiment suggest that both per-user and general classifiers successfully recognize many contexts with high accuracy. However, these classifiers must be substantially improved for commercial use. First, there is a tendency to confuse similar contexts (e.g., a and d, a' and d', c and i), as shown in Table 7. Specifically, a and d (a' and d') are similar in terms of the grip and arm factors of the left hand. The difference between c and i is based on whether the smartphone is held with the left hand or placed in a pocket. An immediate future work will focus on resolving such confusion. Feasible approaches include using additional built-in sensors (e.g.,

inertial sensor in a smartphone) and combining our technique with other previously established techniques such as GripSense [9]. Second, the results described previously are per-frame results. Therefore, an immediate future work is to use temporal low-pass filtering to improve classification accuracy. These will be followed by evaluating the context recognizer in real-life applications.

Our experiment was conducted under limited conditions, that is, in a controlled environment (a closed room). However, a daily user commonly uses a smartphone and smartwatch in various situations and settings (e.g., while riding a bus and outdoors). Therefore, it is necessary to conduct experiments to evaluate the influence of various environments. Moreover, all participants in the experiment were Japanese males. Therefore, considering similarities in term of race, culture, gender, and with respect to walking patterns and postures and how they influence similar lifestyles is all necessary. We will explore these aspects by widening the range of participants in the future.

The contexts shown in Fig. 2 are just a few that exist in daily life. Many others must be explored (e.g., using a smartphone while in bed, taking off a smartwatch and placing it on a desk, and using a smartphone with both hands under the table). Other simple contexts that involve daily activities such as carrying a bag with one hand were also not examined in our study. A future study will explore whether a recognizer can recognize a greater number of contexts and perform the same during daily activities.

In this study, we defined the priority of UIs suitable for contexts. However, these UIs were not evaluated as to whether they were really suitable for the contexts. To address this issue, a user study should be conducted to evaluate user experience of the proposed UIs. In addition, we plan to conduct an additional experiment to investigate the period effect in detail and to evaluate the usability of SynCro's applications.

Latency is another concern. In the recognition method, recognition latency occurs because of using FFT and calculating recognition results. In our current implementation, the latency is 1.65 s: a sliding window covers 1.6 s and the calculation requires 0.05 s. In future work, we plan to explore the effect of latency on the application's usability.

7 CONCLUSION

In this study, we explored context-aware cross-device interactions between a smartphone and smartwatch. First, we showed 24 contexts, and then examine and prioritized suitable UIs for each. In addition, we presented example applications that have their own context-aware UIs. These applications included a map, notification management system, multitasking application, music player, and video chat application.

Moreover, to support these context-aware UIs, we investigated the performance of our context recognizer in which recognition was based on machine-learning using the accelerometers in a smartphone and smartwatch. We conducted seven different evaluations using four machine-learning algorithms: J48 decision tree, SMO-based SVM, random forest, and multilayer perceptron. With each algorithm, we conducted a long-interval experiment to examine the level of accuracy at which each context was recognized using data previously collected for training. The results showed that SMO-based SVM is suitable for recognizing the 24 contexts considered in this study. The results also showed that a greater number of contexts can be recognized using both devices than when using only a single device.

REFERENCES

- [1] Ian Anderson, Julie Maitland, Scott Sherwood, Louise Barkhuus, Matthew Chalmers, Malcolm Hall, Barry Brown, and Henk Muller. 2007. Shakra: Tracking and Sharing Daily Activity Levels with Unaugmented Mobile Phones. *Mobile Networks and Applications* 12, 2 (2007), 185–199. DOI: <http://dx.doi.org/10.1007/s11036-007-0011-7>
- [2] Ling Bao and Stephen S. Intille. 2004. Activity Recognition from User-Annotated Acceleration Data. In *Proceedings of the 2nd International Conference on Pervasive Computing (Pervasive '04)*. Springer, 1–17. DOI: http://dx.doi.org/10.1007/978-3-540-24646-6_1
- [3] Sebastian Boring, David Ledo, Xiang 'Anthony' Chen, Nicolai Marquardt, Anthony Tang, and Saul Greenberg. 2012. The Fat

- Thumb: Using the Thumb's Contact Size for Single-handed Mobile Interaction. In *Proceedings of the 14th International Conference on Human-computer Interaction with Mobile Devices and Services (MobileHCI '12)*. ACM, New York, NY, USA, 39–48. DOI: <http://dx.doi.org/10.1145/2371574.2371582>
- [4] Agata Brajdic and Robert Harle. 2013. Walk Detection and Step Counting on Unconstrained Smartphones. In *Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '13)*. ACM, New York, NY, USA, 225–234. DOI: <http://dx.doi.org/10.1145/2493432.2493449>
 - [5] Xiang 'Anthony' Chen, Tovi Grossman, Daniel J. Wigdor, and George Fitzmaurice. 2014. Duet: Exploring Joint Interactions on a Smart Phone and a Smart Watch. In *Proceedings of the 32nd SIGCHI Conference on Human Factors in Computing Systems (CHI '14)*. ACM, New York, NY, USA, 159–168. DOI: <http://dx.doi.org/10.1145/2556288.2556955>
 - [6] Lung-Pan Cheng, Hsiang-Sheng Liang, Che-Yang Wu, and Mike Y. Chen. 2013. iGrasp: Grasp-based Adaptive Keyboard for Mobile Devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '13)*. ACM, New York, NY, USA, 3037–3046. DOI: <http://dx.doi.org/10.1145/2470654.2481422>
 - [7] Rajkumar Darbar and Debasis Samanta. 2015. SurfaceSense: Smartphone Can Recognize Where It Is Kept. In *Proceedings of the 7th International Conference on HCI, IndiaHCI 2015 (IndiaHCI'15)*. ACM, New York, NY, USA, 39–46. DOI: <http://dx.doi.org/10.1145/2835966.2835971>
 - [8] Anind K. Dey, Gregory D. Abowd, and Daniel Salber. 2001. A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-aware Applications. *Human-Computer Interaction* 16, 2 (Dec. 2001), 97–166. DOI: http://dx.doi.org/10.1207/S15327051HCI16234_02
 - [9] Mayank Goel, Jacob Wobbrock, and Shwetak Patel. 2012. GripSense: Using Built-in Sensors to Detect Hand Posture and Pressure on Commodity Mobile Phones. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology (UIST '12)*. ACM, New York, NY, USA, 545–554. DOI: <http://dx.doi.org/10.1145/2380116.2380184>
 - [10] Mitsuhiro Goto, Hideaki Kimata, Masahiko Toyoshi, Tatsuya Nishikiori, Kosuke Moriwaki, and Koji Nakamura. 2015. A Wearable Action Support System for Business Use by Context-aware Computing Based on Web Schedule. In *Adjunct Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2015 ACM International Symposium on Wearable Computers (UbiComp/ISWC'15 Adjunct)*. ACM, New York, NY, USA, 53–56. DOI: <http://dx.doi.org/10.1145/2800835.2800863>
 - [11] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The WEKA Data Mining Software: An Update. *SIGKDD Explorations Newsletter* 11, 1 (Nov. 2009), 10–18. DOI: <http://dx.doi.org/10.1145/1656274.1656278>
 - [12] Chris Harrison, Julia Schwarz, and Scott E. Hudson. 2011. TapSense: Enhancing Finger Interaction on Touch Surfaces. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology (UIST '11)*. ACM, New York, NY, USA, 627–636. DOI: <http://dx.doi.org/10.1145/2047196.2047279>
 - [13] Seongkook Heo, Jiseong Gu, and Geehyuk Lee. 2014. Expanding Touch Input Vocabulary by using Consecutive Distant Taps. In *Proceedings of the 32nd Annual ACM Conference on Human Factors in Computing Systems (CHI '14)*. ACM, New York, NY, USA, 2597–2606. DOI: <http://dx.doi.org/10.1145/2556288.2557234>
 - [14] Ken Hinckley, Seongkook Heo, Michel Pahud, Christian Holz, Hrvoje Benko, Abigail Sellen, Richard Banks, Kenton O'Hara, Gavin Smyth, and William Buxton. 2016. Pre-Touch Sensing for Mobile Interaction. In *Proceedings of the 34th SIGCHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 2869–2881. DOI: <http://dx.doi.org/10.1145/2858036.2858095>
 - [15] Ken Hinckley, Michel Pahud, Hrvoje Benko, Pourang Irani, François Guimbretière, Marcel Gavrilu, Xiang 'Anthony' Chen, Fabrice Matulic, William Buxton, and Andrew Wilson. 2014. Sensing Techniques for Tablet+Stylus Interaction. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology (UIST '14)*. ACM, New York, NY, USA, 605–614. DOI: <http://dx.doi.org/10.1145/2642918.2647379>
 - [16] Ken Hinckley, Jeff Pierce, Mike Sinclair, and Eric Horvitz. 2000. Sensing Techniques for Mobile Interaction. In *Proceedings of the 13th Annual ACM Symposium on User Interface Software and Technology (UIST '00)*. ACM, New York, NY, USA, 91–100. DOI: <http://dx.doi.org/10.1145/354401.354417>
 - [17] Han-Jong Kim, Seijin Cha, Richard C. Park, Tek-Jin Nam, Woohun Lee, and Geehyuk Lee. 2016. Mo-Bi: Contextual Mobile Interfaces Through Bimanual Posture Sensing with Wrist-Worn Devices. In *Proceedings of HCI Korea (HCIK '16)*. Hanbit Media, Inc., South Korea, 94–99. DOI: <http://dx.doi.org/10.17210/hcik.2016.01.94>
 - [18] Yuki Kubo, Ryosuke Takada, Buntarou Shizuki, and Shin Takahashi. 2017. SynCro: Context-Aware User Interface System for Smartphone-Smartwatch Cross-Device Interaction. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA '17)*. ACM, New York, NY, USA, 1794–1801. DOI: <http://dx.doi.org/10.1145/3027063.3053088>
 - [19] Jennifer R. Kwapisz, Gary M. Weiss, and Samuel A. Moore. 2011. Activity Recognition using Cell Phone Accelerometers. *SIGKDD Explorations Newsletter* 12, 2 (March 2011), 74–82. DOI: <http://dx.doi.org/10.1145/1964897.1964918>
 - [20] Hosub Lee, Young Sang Choi, Sunjae Lee, and Eunsoo Shim. 2013. Smart Pose: Mobile Posture-aware System for Lowering Physical Health Risk of Smartphone Users. In *CHI '13 Extended Abstracts on Human Factors in Computing Systems (CHI EA '13)*. ACM, New York, NY, USA, 2257–2266. DOI: <http://dx.doi.org/10.1145/2468356.2468747>
 - [21] Seungwoo Lee, Yungeun Kim, Daye Ahn, Rhan Ha, Kyoungwoo Lee, and Hojung Cha. 2015. Non-obstructive Room-level

- Locating System in Home Environments using Activity Fingerprints from Smartwatch. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '15)*. ACM, New York, NY, USA, 939–950. DOI: <http://dx.doi.org/10.1145/2750858.2804272>
- [22] Hyunchul Lim, Gwangseok An, Yoonkyong Cho, Kyogu Lee, and Bongwon Suh. 2016. WhichHand: Automatic Recognition of a Smartphone's Position in the Hand using a Smartwatch. In *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services Adjunct (MobileHCI '16)*. ACM, New York, NY, USA, 675–681. DOI: <http://dx.doi.org/10.1145/2957265.2961857>
 - [23] Luyang Liu, Cagdas Karatas, Hongyu Li, Sheng Tan, Marco Gruteser, Jie Yang, Yingying Chen, and Richard P. Martin. 2015. Toward Detection of Unsafe Driving with Wearables. In *Proceedings of the 1st Workshop on Wearable Systems and Applications (WearSys '15)*. ACM, New York, NY, USA, 27–32. DOI: <http://dx.doi.org/10.1145/2753509.2753518>
 - [24] Bilge Mutlu, Andreas Krause, Jodi Forlizzi, Carlos Guestrin, and Jessica Hodgins. 2007. Robust, Low-cost, Non-intrusive Sensing and Recognition of Seated Postures. In *Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology (UIST '07)*. ACM, New York, NY, USA, 149–158. DOI: <http://dx.doi.org/10.1145/1294211.1294237>
 - [25] Woori Noh, Mankyung Lee, Hyelim Cheon, Joohee Kim, Kwangjae Lee, and Jundong Cho. 2016. TakeOut: Drawing Application using Distributed User Interface for Being Close to Real Experience. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct (UbiComp '16)*. ACM, New York, NY, USA, 173–176. DOI: <http://dx.doi.org/10.1145/2968219.2971439>
 - [26] Makoto Ono, Buntarou Shizuki, and Jiro Tanaka. 2013. Touch & Activate: Adding Interactivity to Existing Objects using Active Acoustic Sensing. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology (UIST '13)*. ACM, New York, NY, USA, 31–40. DOI: <http://dx.doi.org/10.1145/2501988.2501989>
 - [27] Chanhho Park and Takefumi Ogawa. 2015. A Study on Grasp Recognition Independent of Users' Situations using Built-in Sensors of Smartphones. In *Adjunct Proceedings of the 28th Annual ACM Symposium on User Interface Software and Technology (UIST '15 Adjunct)*. ACM, New York, NY, USA, 69–70. DOI: <http://dx.doi.org/10.1145/2815585.2815722>
 - [28] Jun-geun Park, Ami Patel, Dorothy Curtis, Seth Teller, and Jonathan Ledlie. 2012. Online Pose Classification and Walking Speed Estimation using Handheld Devices. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing (UbiComp '12)*. ACM, New York, NY, USA, 113–122. DOI: <http://dx.doi.org/10.1145/2370216.2370235>
 - [29] Anand Ranganathan and Roy H. Campbell. 2003. An Infrastructure for Context-awareness Based on First Order Logic. *Personal Ubiquitous Comput.* 7, 6 (Dec. 2003), 353–364. DOI: <http://dx.doi.org/10.1007/s00779-003-0251-x>
 - [30] Juhi Ranjan and Kamin Whitehouse. 2015. Object Hallmarks: Identifying Object Users using Wearable Wrist Sensors. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '15)*. ACM, New York, NY, USA, 51–61. DOI: <http://dx.doi.org/10.1145/2750858.2804263>
 - [31] Jun Rekimoto. 1996. Tilting Operations for Small Screen Interfaces. In *Proceedings of the 9th Annual ACM Symposium on User Interface Software and Technology (UIST '96)*. ACM, New York, NY, USA, 167–168. DOI: <http://dx.doi.org/10.1145/237091.237115>
 - [32] Jun Rekimoto. 1997. Pick-and-Drop: A Direct Manipulation Technique for Multiple Computer Environments. In *Proceedings of the 10th Annual ACM Symposium on User Interface Software and Technology (UIST '97)*. ACM, New York, NY, USA, 31–39. DOI: <http://dx.doi.org/10.1145/263407.263505>
 - [33] Nazir Saleheen, Amin Ahsan Ali, Syed Monowar Hossain, Hillol Sarker, Soujanya Chatterjee, Benjamin Marlin, Emre Ertin, Mustafa al'Absi, and Santosh Kumar. 2015. puffMarker: A Multi-sensor Approach for Pinpointing the Timing of First Lapse in Smoking Cessation. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '15)*. ACM, New York, NY, USA, 999–1010. DOI: <http://dx.doi.org/10.1145/2750858.2806897>
 - [34] Munehiko Sato, Ivan Poupyrev, and Chris Harrison. 2012. Touché: Enhancing Touch Interaction on Humans, Screens, Liquids, and Everyday Objects. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '12)*. ACM, New York, NY, USA, 483–492. DOI: <http://dx.doi.org/10.1145/2207676.2207743>
 - [35] Bill N. Schilit, Norman Adams, and Roy Want. 1994. Context-Aware Computing Applications. In *Proceedings of the 1994 First Workshop on Mobile Computing Systems and Applications (WMCSA '94)*. IEEE Computer Society, Washington, DC, USA, 85–90. DOI: <http://dx.doi.org/10.1109/WMCSA.1994.16>
 - [36] Albrecht Schmidt, Kofi Asante Aidoo, Antti Takaluoma, Urpo Tuomela, Kristof Van Laerhoven, and Walter Van de Velde. 1999. Advanced Interaction in Context. In *Proceedings of the 1st International Symposium on Handheld and Ubiquitous Computing (HUC '99)*. Springer-Verlag, London, UK, 89–101. <http://dl.acm.org/citation.cfm?id=647985.743849>
 - [37] Dominik Schmidt, Julian Seifert, Enrico Rukzio, and Hans Gellersen. 2012. A Cross-device Interaction Style for Mobiles and Surfaces. In *Proceedings of the Designing Interactive Systems Conference (DIS '12)*. ACM, New York, NY, USA, 318–327. DOI: <http://dx.doi.org/10.1145/2317956.2318005>
 - [38] Sheng Shen, He Wang, and Romit Roy Choudhury. 2016. I Am a Smartwatch and I Can Track My User's Arm. In *Proceedings of the 14th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys '16)*. ACM, New York, NY, USA, 85–96. DOI: <http://dx.doi.org/10.1145/2906388.2906407>
 - [39] Katsuma Tanaka, Shoya Ishimaru, Koichi Kise, Kai Kunze, and Masahiko Inami. 2015. Nekoze!: Monitoring and Detecting Head Posture

- While Working with Laptop and Mobile Phone. In *Proceedings of the 9th International Conference on Pervasive Computing Technologies for Healthcare (PervasiveHealth '15)*. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), ICST, Brussels, Belgium, Belgium, 237–240. <http://dl.acm.org/citation.cfm?id=2826165.2826203>
- [40] Edison Thomaz, Irfan Essa, and Gregory D. Abowd. 2015. A Practical Approach for Recognizing Eating Moments with Wrist-mounted Inertial Sensing. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '15)*. ACM, New York, NY, USA, 1029–1040. DOI : <http://dx.doi.org/10.1145/2750858.2807545>
 - [41] Raphael Wimmer and Sebastian Boring. 2009. HandSense: Discriminating Different Ways of Grasping and Holding a Tangible User Interface. In *Proceedings of the 3rd International Conference on Tangible and Embedded Interaction (TEI '09)*. ACM, New York, NY, USA, 359–362. DOI : <http://dx.doi.org/10.1145/1517664.1517736>
 - [42] Yoonsik Yang, Seunggho Chae, Jinwook Shim, and Tack-Don Han. 2015. EMG Sensor-based Two-Hand Smart Watch Interaction. In *Adjunct Proceedings of the 28th Annual ACM Symposium on User Interface Software and Technology (UIST '15 Adjunct)*. ACM, New York, NY, USA, 73–74. DOI : <http://dx.doi.org/10.1145/2815585.2815724>
 - [43] Dongwook Yoon, Ken Hinckley, Hrvoje Benko, François Guimbretière, Pourang Irani, Michel Pahud, and Marcel Gavriliu. 2015. Sensing Tablet Grasp + Micro-mobility for Active Reading. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software and Technology (UIST '15)*. ACM, New York, NY, USA, 477–487. DOI : <http://dx.doi.org/10.1145/2807442.2807510>

Received February 2017; revised May 2017; accepted June 2017