Electronic Preprint for Journal of Information Processing Vol.30

Regular Paper

Carrying-mode Free Indoor Positioning Using Smartphone and Smartwatch and Its Evaluations

Tomoya Wakaizumi[1,a)] Nozomu Togawa[1,b)]

Received: March 12, 2021, Accepted: October 8, 2021

Abstract: As smartphones are much used over a wide area, the pedestrian navigation systems are greatly utilized in our daily lives. In general, navigation systems use GPS (Global Positioning System) for the user's positioning but its accuracy tends to decrease in indoor environments. A pedestrian dead reckoning method, or PDR method in short, is one of the positioning methods in indoor environments, which estimates the user's positions by using sensors such as acceleration and angular velocity sensors. These PDR methods do not always use external infrastructures and hence they can be implemented with a lower cost. When we consider using a smartphone as a PDR sensor device, attention must be paid to the fact that there are various *carrying modes* such as holding it directly and carrying it inside a pocket. How to deal with these various carrying modes is of great concern in PDR when using a smartphone. In this paper, we propose a PDR method based on a combination of a smartphone and a smartwatch. By synchronizing the smartphone and smartwatch sensors effectively, the proposed method can successfully reduce drift errors and thus estimate accurately the user's positions, compared to just using a smartphone. Furthermore, even when the user carries his/her smartphone in various carrying modes, the proposed method still realizes accurate PDR. The experimental results demonstrate that the positioning errors are reduced by approximately 82.1% on average compared to the existing method.

Keywords: PDR, indoor positioning, smartphone, smartwatch, carrying mode

1. Introduction

1.1 Indoor Positioning by Pedestrian Dead Reckoning

In recent years, the pedestrian navigation systems are widely utilized in our daily lives as smartphones are much used over a wide area. In general, navigation systems use GPS (Global Positioning System) for the user's positioning but its accuracy tends to decrease in indoor environments, since radio waves from artificial satellites may be blocked there. In indoor environments, alternatives to GPS are required for positioning and the effective one is a pedestrian dead reckoning method, or PDR method in short. PDR methods estimate the user's positions based on walking distance and direction, which are estimated by acceleration and angular velocity sensors [1]. These PDR methods do not always use external infrastructures and hence they can be implemented at a lower cost compared with using Wi-Fi access points and Bluetooth beacons.

When using a smartphone as a PDR sensor device, attention must be paid to the fact that it has various *carrying modes*, such as carrying a smartphone in front of one's body, swinging a smartphone in the hand, keeping a smartphone in a trouser pocket and keeping a smartphone in a bag. The PDR method optimized for one particular mode may cause significant positioning errors when used in other modes.

In summary, the PDR method based on the smartphone sensors must satisfy that (Condition A) it uses no external infrastructures, (Condition B) it estimates the user's positions with any smartphone's carrying modes, and (Condition C) the positioning errors are minimized.

1.2 Previous Researches

Many PDR methods using smartphones have been proposed to date. In particular, those focusing on the smartphone's carrying modes have been proposed in Refs. [2], [3]. Lee et al. proposed the smartphone's carrying mode recognition method utilizing the threshold [2]. In this method, the smartphone's carrying mode is estimated by calculating the average of acceleration and angular velocity values per step. Tian et al. proposed the PDR method using smartphones based on a particular carrying mode [3], where the state machine is configured according to the smartphone modes and the state transition is performed based on the angular velocity values. These methods can improve estimation accuracy assuming a particular carrying mode but it is necessary to properly set the threshold for carrying mode estimation.

Recently, there have also been proposed smartphone's carrying mode recognition methods using machine learning [4], [5], [6]. In these machine learning based methods, the smartphone's carrying mode is estimated using the acceleration change period, its average values, and its variance values. These methods can successfully estimate the carrying mode but it is difficult to estimate the smartphone's carrying modes other than the pre-assumed ones. In addition, they require extra costs for learning. All the above methods estimate the user's positions without using external infrastructures. However, these methods are based on estimating a specific smartphone's carrying mode and hence these methods

[1] Waseda University, Shinjuku, Tokyo, 169–8555 Japan
[a)] tomoya.wakaizumi@togawa.cs.waseda.ac.jp
[b)] togawa@togawa.cs.waseda.ac.jp

satisfy (Condition A), but do not satisfy (Condition B) above.

Hoshi et al. proposed the PDR method without assuming a specific smartphone's carrying mode [7]. This method needs no external infrastructures. However, this method estimates the user's direction only using angular velocity and hence positioning errors due to drifts can occur very easily. This method satisfies (Condition A) and (Condition B), but does not satisfy (Condition C) above.

While a smartphone has many carrying modes, a smartwatch [8], [9] is usually worn on one's wrist and hence it is less susceptible to mode changes. A smartwatch can lead to providing robust PDR. Loh et al. proposed the PDR method with a smartwatch and a smartglass [10]. In this method, direction estimation is realized by using acceleration and geomagnetic sensors equipped with a smartwatch, and stride length estimation is realized by using a smartglass. This method satisfies (Condition B), but direction estimation is realized only by a smartwatch and hence the positioning errors due to drifts can occur too frequently.

Correra et al. proposed the PDR method with a smartphone and a smartwatch [11]. This method uses the received signal strength value from Wi-Fi as well as smartphone sensors, which results in a highly accurate estimation. This method satisfies (Condition A) and (Condition C) but it does not satisfy (Condition B) since it requires Wi-Fi.

Now we consider using smartphone sensors and smartwatch sensors. As mentioned above, a smartwatch is usually worn on the wrist and hence it is less susceptible to mode changes. Using both smartphone sensors and smartwatch sensors can also be less susceptible to mode changes, which can result in high accuracy position estimation. As a result, (Condition A) to (Condition C) can all be satisfied. As far as we know, no PDR methods are proposed using smartphones and smartwatches simultaneously satisfying (Condition A) to (Condition C).

1.3 Our Proposal

In this paper, we propose a PDR method using a smartphone and a smartwatch simultaneously [*1]. The proposed method can estimate the user's positions not depending on the smartphone's carrying modes, without using external infrastructures. In addition, the method reduces direction estimation errors due to drifts using smartphone and smartwatch sensors and hence realizes accurate PDR.

In the proposed method, step detection, axis correction, and direction estimation are firstly performed on a smartphone and a smartwatch, by using their acceleration and gyroscope sensors. These processes are performed on each device. After that, the data obtained by the smartphone is synchronized in the smartphone device with those obtained by the smartwatch. By comparing the synchronized data, we can estimate the user's direction very accurately. Finally, the user's position is estimated based on the estimated step counts and user's direction. The proposed method satisfies all the conditions of (Condition A) to (Condition C).

1.4 Contributions of This Paper

The contributions of this paper are summarized as follows:

(1) We realize indoor positioning not depending on the smartphone's carrying modes by effectively utilizing the properties of walking motions (See Section 3.4).
(2) Without using external infrastructures, low cost positioning is realized.
(3) Using a smartphone and a smartwatch simultaneously, positioning errors due to drifts are much reduced and hence highly accurate PDR can be realized.
(4) We have conducted experiments with various of the smartphone's carrying modes. As a result, the proposed method reduced the position estimation errors by approximately 82.1% on average, compared with the existing method [7].

2. Indoor Positioning Problem

In this section, we define an indoor positioning problem using smartphone and smartwatch sensors. In the indoor positioning defined in this section, the user's position is estimated by a tri-axis acceleration sensor and a tri-axis angular velocity sensor installed in smartphones and smartwatches.

As in **Fig. 1** (a), a tri-axis acceleration sensor installed in smartphones [13], [14] obtains horizontal lateral-direction acceleration values (X-axis acceleration values), horizontal longitudinal acceleration values (Y-axis acceleration values), and vertical acceleration values (Z-axis acceleration values). Let $\vec{a_p}(t) = (a_p^x(t), a_p^y(t), a_p^z(t))$ represent a tri-axis acceleration vector at time t, where $a_p^x(t)$, $a_p^y(t)$, and $a_p^z(t)$ represent the X-axis acceleration value, Y-axis acceleration value, and Z-axis acceleration value at time t, respectively. As in Fig. 1 (a), a tri-axis angular velocity sensor installed in smartphones obtains the angular velocities around horizontal lateral direction (X-axis angular velocities), and the angular velocities around horizontal longitudinal direction (Y-axis angular velocities), the angular velocities around vertical direction (Z-axis angular velocity). Let $\vec{\omega_p}(t) = (\omega_p^x(t), \omega_p^y(t), \omega_p^z(t))$ represent a tri-axis angular velocity vector at time t, where $\omega_p^x(t)$, $\omega_p^y(t)$, and $\omega_p^z(t)$ represent the X-axis angular velocity, Y-axis angular velocity, and Z-axis angular velocity at time t, respectively.

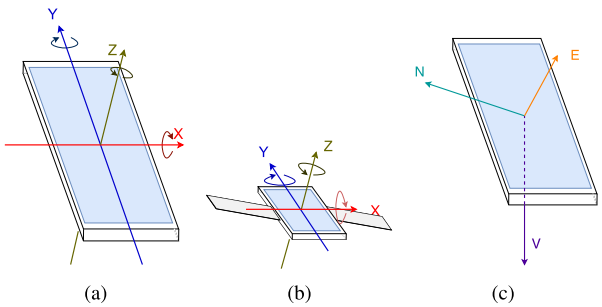


Fig. 1 Axes of smartphone and smartwatch and axes on geospatial space. (a) Three axes on smartphone. (b) Three axes on smartwatch. (c) Three axes (north direction (N), east direction (E) and gravity direction (V)) on geospatial space.

[*1] The preliminary version of this paper appeared in Ref. [12]. The main differences between Ref. [12] and this paper are that we describe the proposed method in detail in Section 3 and add more variety of experiments to clearly demonstrate the effectiveness of our proposed method in Section 4.

In the same way, we can define a tri-axis acceleration vector at time t, $\vec{a_w}(t) = (a_w^x(t), a_w^y(t), a_w^z(t))$, and a tri-axis angular velocity vector at time t, $\vec{\omega_w}(t) = (\omega_w^x(t), \omega_w^y(t), \omega_w^z(t))$, for smartwatches [9], [15] as in Fig. 1 (b).

It is assumed that the user's initial position $\vec{p} = (x_0, y_0)$, the user's initial direction θ_0, and the user's stride length l are given. We further assume that (1) the user wears a smartwatch on the wrist and (2) he/she slightly *swings* it at the side of the body. The tri-axis acceleration values and tri-axis angular velocity values are obtained every t_i^p seconds from the smartphone sensors [*2]. The tri-axis acceleration values and tri-axis angular velocity values are obtained every t_i^w seconds from the smartwatch sensors. Note that the smartphone sensors and smartwatch sensors are not synchronized and hence sensor data at exactly the same time cannot be obtained from them.

Based on the above discussion, the indoor positioning problem is defined as follows:

Definition 1 Given the user's initial position $\vec{p} = (x_0, y_0)$, the user's initial direction θ_0, and the user's stride length l, the indoor positioning problem here is to estimate the user's position every t_i seconds, by using the tri-axis acceleration vectors and the tri-axis angular velocity vectors obtained every t_i^p seconds from the smartphone sensors as well as the tri-axis acceleration vectors and the tri-axis angular velocity vectors obtained every t_i^w seconds from the smartwatch sensors [*3]. Note that the smartphone sensors and smartwatch sensors are not synchronized.

3. The Proposed PDR Method Using a Smartphone and a Smartwatch

As in the discussion in Section 1, we use smartphone and smartwatch sensors simultaneously to tackle the indoor positioning problem.

Positioning errors can be accumulated in the PDR method when only using a smartphone. In order to reduce the cumulative errors, external infrastructures such as Wi-Fi and Bluetooth are often used [16], [17] but these external infrastructures definitely increase installation costs. In our approach, this problem is solved by using a smartphone and a smartwatch simultaneously and correcting the positioning errors.

In this section, we firstly point out the problems when using a smartphone and a smartwatch simultaneously in Section 3.1. After that, we propose our PDR method in Section 3.2. The detailed steps in the proposed method are described in Section 3.3 to Section 3.8.

3.1 Problems of PDR with a Smartphone and a Smartwatch

Indoor positioning using a smartphone and a smartwatch simultaneously estimates the user's steps and direction using their acceleration and gyroscope sensors. In order to use the smartphone and smartwatch sensors simultaneously, it is necessary to collect sensor data into either one of the devices. However, according to our preliminary experiments, sending data from one device to the other device requires several hundreds of micro seconds and furthermore the delay time varies a great deal. It is quite difficult to accurately associate the data obtained by one device with the other one. To solve this problem, the proposed method synchronizes the data obtained by a smartphone with those obtained by a smartwatch based on the user's step behaviors (see Section 3.6 in detail).

When using smartphone and smartwatch sensors simultaneously, how to combine them becomes another problem. For example, consider estimating a user's direction by using smartphone and smartwatch sensors. The easiest approach is that we first estimate the user's direction by using each of the devices and then average them. However, this approach cannot expect to reduce positioning errors if both smartphone and smartwatch sensors occur drift errors to the same direction. The proposed method first calculates the *covariance* of the user's direction estimated by smartphone and smartwatch sensors. By filtering angular velocity values using the covariance, we can realize the accurate user's direction estimation (see Section 3.7 in detail). As a result, the proposed method can distinguish between the user's going straight and turning right/left and hence the cumulative errors can be reduced.

In order to estimate the user's positions no matter what the smartphone's carrying mode is, we have to recognize in which direction the smartphone turns. As described earlier, the existing methods distinguish between carrying modes by comparing the predefined ones and observed ones [2], [6]. These methods cannot estimate the user's positions when the smartphone's carrying modes are those other than prepared ones. The proposed method recognizes how much the smartphone is tilted by averaging acceleration values per step and comparing it to the gravity vector and hence we can recognize the user's direction independent of the smartphone's carrying mode (see Section 3.4 in detail).

From the above, the proposed method can satisfy (Condition A) to (Condition C) in Section 1.

3.2 Flow of Proposed Method

Based on the above discussions, **Fig. 2** shows the flow of the proposed method. First, tri-axis acceleration and tri-axis angular velocity values are obtained from the smartphone and smartwatch sensors and the user's steps and how much a smartphone and a smartwatch are tilted are estimated based on them. After

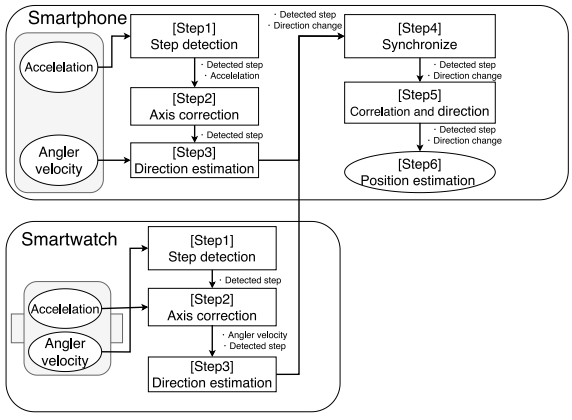


Fig. 2 Flow of the proposed method.

[*2] In Section 4, we define $t_i^p = 25\,\text{ms}$ and $t_i^w = 50\,\text{ms}$ according to Refs. [9], [14].

[*3] In Section 4, we set $t_i = 50\,\text{ms}$.

that, the proposed method synchronizes a smartphone data with a smartwatch data and estimate the user's direction and positions.

The flow of the proposed method is summarized as follows:

(Step 1) User's step detection

In Step 1, the user's steps are detected by acceleration periodicity in a smartphone. Also, the user's steps are detected independently by angular velocity periodicity in a smartwatch.

(Step 2) Axis correction

In Step 2, the three axes of the smartphone and smartwatch are converted into the three axes in the geospatial space.

(Step 3) User's direction estimation

By integrating angular velocity values from the user's initial direction, the proposed method estimates the user's direction as a difference from the initial direction.

(Step 4) Synchronization

It is difficult to accurately associate the data obtained from the smartphone with those from the smartwatch due to the varying sending/receiving delays between them. To solve the problem, the proposed method synchronizes the smartphone data with the smartwatch data by using the user's steps.

(Step 5) Covariance and integrated user's direction estimation

The proposed method calculates the covariance between the angular velocity values of the smartphone and those of the smartwatch, and changes the user's direction only if the positive covariance is large enough. As a result, the errors due to drifts are much reduced when the user is moving in a straight path, and a highly accurate estimation is realized.

(Step 6) Positioning

The proposed method estimates the user's current position by the user's step counts and estimated direction.

Note that Step 1 to Step 3 are processed in the smartphone and smartwatch independently. After that the data processed in the smartwatch is sent to the smartphone and Step 4 to Step 6 are processed in the smartphone.

In the following, we propose each step of Step 1 to Step 6 above.

3.3 Step Detection (Step 1)

3.3.1 In the Case of the Smartphone

Figure 3 shows the acceleration values when walking with a smartphone. Figure 3 shows the data during 10 seconds while walking in a straight path holding Google's smartphone Pixel 3 [14] in front of the body. In Fig. 3, the horizontal axis shows the measurement time and the vertical axis shows acceleration values. The acceleration value here is the total acceleration value, which is given by Eq. (1) below:

$$a_p^{sum}(t) = \sqrt{\bar{a}_p^x(t)^2 + \bar{a}_p^y(t)^2 + \bar{a}_p^z(t)^2} \tag{1}$$

where $\bar{a}_p^x(t)$, $\bar{a}_p^y(t)$, and $\bar{a}_p^z(t)$ are the X-axis, Y-axis, and Z-axis smoothed acceleration values at time t by the smartphone, respectively. Note that every acceleration value is smoothed [4] to reduce the noises as given below:

$$\begin{cases} \bar{a}_p^x(t) = \dfrac{1}{3} \displaystyle\sum_{k\in\{t^-,t,t^+\}} a_p^x(k) \\ \bar{a}_p^y(t) = \dfrac{1}{3} \displaystyle\sum_{k\in\{t^-,t,t^+\}} a_p^y(k) \\ \bar{a}_p^z(t) = \dfrac{1}{3} \displaystyle\sum_{k\in\{t^-,t,t^+\}} a_p^z(k) \end{cases} \tag{2}$$

where t^- and t^+ represent the times when the smartphone sensor obtains the acceleration values one time before and one time after the time t, respectively.

Figure 3 clearly demonstrates that the acceleration values of the smartphone while walking change periodically. According to our preliminary experiments, the acceleration values change periodically in other smartphone's carrying modes as well.

From the above, in Step 1, the X-axis, Y-axis, and Z-axis acceleration values of the smartphone are smoothed by Eq. (2) and we obtain the user's step counts on the smartphone by detecting every peak of total acceleration values.

3.3.2 In the Case of the Smartwatch

Figure 4 shows the acceleration values when walking with a smartwatch and **Fig. 5** shows the angular velocity values when walking with a smartwatch. Figures 4 and 5 show data for 10 seconds while walking on a straight path using Mobvoi's smartwatch TicWatch Pro [9]. In Fig. 4, the horizontal axis shows the measurement time and the vertical axis shows the acceleration values. As described above, the acceleration values here represent the total acceleration values as in Eqs. (1) and (2). In Fig. 5, the horizontal axis shows the measurement time and the vertical axis shows the angular velocity values, which are obtained in the same way as the total acceleration values.

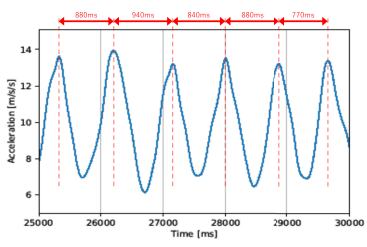


Fig. 3 Total acceleration values by smartphone.

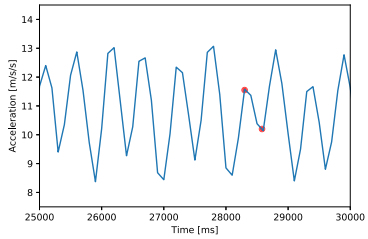


Fig. 4 Total acceleration values by smartwatch.

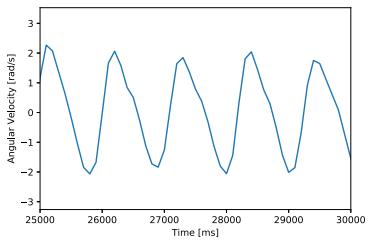


Fig. 5 Total angular velocity values by smartwatch.

Figure 4 demonstrates that the acceleration values of the smartwatch change periodically, but there is a part where the difference between the maximum and minimum values of the acceleration values are much too small as in the red dots in Fig. 4. For this reason, we may not detect either local minima or local maxima when using these acceleration values. However, Fig. 5 demonstrates that the angular velocity values of the smartwatch change periodically and hence we can detect every peak more clearly in the angular velocity when using a smartwatch.

From the above, as in the smartphone, we obtain the user's step counts on the smartwatch by detecting every peak of angular velocity values.

3.4 Axis Correction (Step 2)

As discussed earlier, the axes of a smartphone and a smartwatch are different from those of geospatial spaces as in Fig. 1. We have to convert the devices' axes to geospatial spaces for every data obtained.

3.4.1 In the Case of the Smartphone

Firstly, **Fig. 6** depicts the typical user's motion where the blue arrow shows the gravity and the red arrow shows the acceleration. If we walk at a constant speed, the acceleration occurs up/down and front/back as depicted and the acceleration values over a certain period of time without the gravity will cancel out.

Due to the walking characteristics, the vertical and forward-to-backward-directed acceleration values are cyclic for each user's step. This indicates that, if the acceleration values of the user's one step are averaged, the acceleration values horizontal to the ground will be zero. Since the gravity always acts on the user, it remains in the vertical down direction to the ground in the averaged acceleration values.

From the above direction, in the case of walking at a constant speed, the average value over all the acceleration values per step in the horizontal direction will become zero. In this case, if we calculate the average value over total acceleration values per step, it definitely shows the gravity. Once we can obtain the gravity vector, we can also obtain the angles between the gravity vector and each axis in the smartphone no matter in what mode the smartphone is located.

Let $\vec{a}^{x}_{p,ave}(n)$, $\vec{a}^{y}_{p,ave}(n)$, and $\vec{a}^{z}_{p,ave}(n)$ be the average acceleration vectors over X-axis, Y-axis, and Z-axis acceleration values, respectively, in the n-th of the user's step by the smartphone sensor as given below:

$$\begin{cases} \vec{a}^{x}_{p,ave}(n) = \dfrac{1}{|\mathbb{T}_p(n)|} \displaystyle\sum_{k \in \mathbb{T}_p(n)} \vec{a}^{x}_{p}(k) \\ \vec{a}^{y}_{p,ave}(n) = \dfrac{1}{|\mathbb{T}_p(n)|} \displaystyle\sum_{k \in \mathbb{T}_p(n)} \vec{a}^{y}_{p}(k) \\ \vec{a}^{z}_{p,ave}(n) = \dfrac{1}{|\mathbb{T}_p(n)|} \displaystyle\sum_{k \in \mathbb{T}_p(n)} \vec{a}^{z}_{p}(k) \end{cases} \tag{3}$$

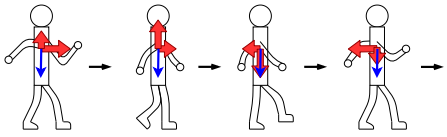


Fig. 6 Walking motion. The blue arrow shows the gravity and the red arrow shows the acceleration.

where $\vec{a}_p(t) = (a^x_p(t), a^y_p(t), a^z_p(t))$ shows the tri-axis acceleration vector at time t obtained by the smartphone sensor and $\mathbb{T}_p(n)$ shows a set of measurement times included in the n-th of the user's step by the smartphone sensor.

At that time, $\vec{g}_p(n)$ which is the estimated gravity vector at the n-th of the user's step by the smartphone can be written by:

$$\vec{g}_p(n) = \vec{a}^{x}_{p,ave}(n) + \vec{a}^{y}_{p,ave}(n) + \vec{a}^{z}_{p,ave}(n) \tag{4}$$

Let $\vec{\Omega}_p(n) = (\Omega^x_p(n), \Omega^y_p(n), \Omega^z_p(n))$ be the angle vector, in which each of the components shows the angle between the gravity vector and X-axis, Y-axis, or Z-axis of the smartphone at the n-th of the user's step. Then we have:

$$\begin{cases} \vec{a}^{x}_{p,ave}(n) \cdot \vec{g}_p(n) = |\vec{a}^{x}_{p,ave}(n)| \cdot |\vec{g}_p(n)| \cdot \cos\Omega^x_p(n) \\ \vec{a}^{y}_{p,ave}(n) \cdot \vec{g}_p(n) = |\vec{a}^{y}_{p,ave}(n)| \cdot |\vec{g}_p(n)| \cdot \cos\Omega^y_p(n) \\ \vec{a}^{z}_{p,ave}(n) \cdot \vec{g}_p(n) = |\vec{a}^{z}_{p,ave}(n)| \cdot |\vec{g}_p(n)| \cdot \cos\Omega^z_p(n) \end{cases} \tag{5}$$

Since $\vec{a}^{x}_{p,ave}(n)$, $\vec{a}^{y}_{p,ave}(n)$, and $\vec{a}^{z}_{p,ave}(n)$ are orthogonal to each other, we can lead to Eq. (6) below by substituting Eq. (4) for Eq. (5):

$$\begin{cases} \Omega^x_p(n) = \arccos \dfrac{|\vec{a}^{x}_{p,ave}(n)|}{|\vec{g}_p(n)|} \\ \Omega^y_p(n) = \arccos \dfrac{|\vec{a}^{y}_{p,ave}(n)|}{|\vec{g}_p(n)|} \\ \Omega^z_p(n) = \arccos \dfrac{|\vec{a}^{z}_{p,ave}(n)|}{|\vec{g}_p(n)|} \end{cases} \tag{6}$$

The above discussion clearly demonstrates that, no matter what the smartphone's carrying mode is, we can convert three smartphone's axes to the three axes of the geospatial spaces. Hence the proposed method can satisfy (Condition B) in Section 1.

3.4.2 In the Case of the Smartwatch

Since the smartwatch's carrying modes are very limited, it is possible to assume that the smartwatch is worn on the wrist. However, every user has different ways of swinging his/her arm and he/she does not always swings his/her arm on the plane vertical to the ground. It is still necessary to obtain how much the smartwatch is tilted.

Let $\vec{a}^{x}_{w,ave}(n)$, $\vec{a}^{y}_{w,ave}(n)$, and $\vec{a}^{z}_{w,ave}(n)$ be the average acceleration vectors over X-axis, Y-axis, and Z-axis acceleration values, respectively, in the n-th of the user's step by the smartwatch sensor as given below:

$$\begin{cases} \vec{a}^{x}_{w,ave}(n) = \dfrac{1}{|\mathbb{T}_w(n)|} \displaystyle\sum_{k \in \mathbb{T}_w(n)} \vec{a}^{x}_{w}(k) \\ \vec{a}^{y}_{w,ave}(n) = \dfrac{1}{|\mathbb{T}_w(n)|} \displaystyle\sum_{k \in \mathbb{T}_w(n)} \vec{a}^{y}_{w}(k) \\ \vec{a}^{z}_{w,ave}(n) = \dfrac{1}{|\mathbb{T}_w(n)|} \displaystyle\sum_{k \in \mathbb{T}_w(n)} \vec{a}^{z}_{w}(k) \end{cases} \tag{7}$$

At that time, $\vec{g}_w(n)$ which is the estimated gravity vector at the n-th of the user's step by the smartphone can be written by:

$$\vec{g}_w(n) = \vec{a}^{x}_{w,ave}(n) + \vec{a}^{y}_{w,ave}(n) + \vec{a}^{z}_{w,ave}(n) \tag{8}$$

Let $\vec{\Omega}_w(n) = (\Omega^x_w(n), \Omega^y_w(n), \Omega^z_w(n))$ be the angle vector, in

which each of the components shows the angle between the gravity vector and X-axis, Y-axis, or Z-axis of the smartwatch at the n-th of the user's step. We can lead to Eq. (9) below as in the case of smartphone:

$$\begin{cases} \Omega_w^x(n) = \arccos \dfrac{|\vec{a}_{w,ave}^x(n)|}{|\vec{g}_w(n)|} \\ \Omega_w^y(n) = \arccos \dfrac{|\vec{a}_{w,ave}^y(n)|}{|\vec{g}_w(n)|} \\ \Omega_w^z(n) = \arccos \dfrac{|\vec{a}_{w,ave}^z(n)|}{|\vec{g}_w(n)|} \end{cases} \tag{9}$$

The above discussion clearly demonstrates that, no matter what the smartwatch's carrying mode is, we can convert three smartwatch's axes to the three axes of the geospatial spaces. Hence the proposed method can still satisfy (Condition B) in Section 1.

Note that, as the smartwatch's carrying modes are very limited, we can expect that the errors in Eq. (9) must be very small and combining the smartphone and the smartwatch leads to very robust indoor positioning.

3.5 User's Direction Estimation (Step 3)

The proposed method estimates the user's direction by using angular velocity values in the smartphone and the smartwatch.

3.5.1 In the Case of the Smartphone

In the user's direction estimation, horizontal angular velocity values are used. Let $\omega_p^{hol}(t)$ be the horizontal angular velocity of the smartphone at time t. Let $\Omega_p^x(i)$, $\Omega_p^y(i)$, and $\Omega_p^z(i)$ be the angles between the gravity vector and X-axis, Y-axis, and Z-axis of the smartphone, respectively, at i-th of the user's step. Then $\omega_p^{hol}(t)$ is obtained by Eq. (11) below:

$$\begin{aligned} \omega_p^{hol}(t) = {} & \omega_p^x(t) \times \cos \Omega_p^x(i) \\ & + \omega_p^y(t) \times \cos \Omega_p^y(i) + \omega_p^z(t) \times \cos \Omega_p^z(i) \end{aligned} \tag{10}$$

Now let $\Delta\theta_p(i)$ be the user's direction change at i-th of the user's step from $(i-1)$-th of the user's step by the smartphone. $\Delta\theta_p(i)$ can be obtained by integrating the horizontal angular velocity values over the times corresponding to i-th of the user's step. Let $t_p^{start}(i)$ and $t_p^{end}(i)$ be the start time and end time at i-th of the user's step, respectively, and Δt_p be the data retrieval interval by the smartphone. Then $\Delta\theta_p(i)$ can be written by:

$$\Delta\theta_p(i) = \sum_{k=t_p^{start}(i)}^{t_p^{end}(i)} (\omega_p^{hol}(k) \times \Delta t_p) \tag{11}$$

3.5.2 In the Case of the Smartwatch

The user's direction change by the smartwatch is obtained similarly. Let $\omega_w^{hol}(t)$ be the horizontal angular velocity of the smartwatch at time t. $\Omega_w^x(i)$, $\Omega_w^y(i)$, and $\Omega_w^z(i)$ be the angles between the gravity vector and X-axis, Y-axis, and Z-axis of the smartwatch, respectively, at i-th of the user's step. Then $\omega_w^{hol}(t)$ is obtained by Eq. (13) below:

$$\begin{aligned} \omega_w^{hol}(t) = {} & \omega_w^x(t) \times \cos \Omega_w^x(i) \\ & + \omega_w^y(t) \times \cos \Omega_w^y(i) + \omega_w^z(t) \times \cos \Omega_w^z(i) \end{aligned} \tag{12}$$

Now let $\Delta\theta_w(i)$ be the user's direction change at i-th of the user's step from $(i-1)$-th of the user's step by the smartwatch. Let $t_p^{start}(i)$ and $t_p^{end}(i)$ be the start time and end time at i-th of the user's step, respectively, and Δt_p be the data retrieval interval by the smartwatch. Then $\Delta\theta_p(i)$ can be written by:

$$\Delta\theta_w(i) = \sum_{k=t_w^{start}(i)}^{t_w^{end}(i)} (\omega_w^{hol}(k) \times \Delta t_w) \tag{13}$$

3.6 Synchronization (Step 4)

To use sensor values obtained by both the smartphone and smartwatch, we have to send the sensor data of one device to the other, which requires the transmission delay between them. We have conducted the preliminary experiments on the transmission delays. In this experiment, we use Google's smartphone Pixel 3 and Mobvoi's smartwatch TicWatch Pro as follows:

(1) We connect the smartphone to the smartwatch via Bluetooth and send a 16-byte character string A from the smartphone to the smartwatch (the send start time is t_A).
(2) After the smartwatch receives the character string A, we send back a 16-byte character string B from the smartwatch to the smartphone.
(3) Then the smartphone receives the character string B (the receiving time is t_B).
(4) The delay time between the smartphone and the smartwatch is defined by $(t_B - t_A)/2$ at time t_A.
(5) **Figure 7** is the plots of the delay time obtained by repeating (1) to (4) above every 500 ms. The horizontal axis represents the time t and the vertical axis represents the delay time.

As in Fig. 7, the transition delay time becomes 100 ms to 400 ms. It is not the constant value. It is difficult to uniquely associate the smartphone sensor data with the smartwatch sensor data. On the other hand, the walking cycle is approximately 700 ms to 1,000 ms as in Fig. 3 and the delay time in Fig. 7 is smaller than the walking cycle. Hence the proposed method synchronizes the smartphone sensor data with the smartwatch sensor data based on every user's step obtained by Step 1. Thus, the goal of the synchronization step is to find out the correspondence between the smartphone sensor data and the smartwatch sensor data in a step-by-step manner.

The synchronization step (Step 4) is composed of (1) the step-synchronization process and (2) the step-correction process. We propose these two processes below:

3.6.1 (1) Step-synchronization Process

Firstly, we describe the step-synchronization process. As above, it is difficult to uniquely associate the smartphone sen-

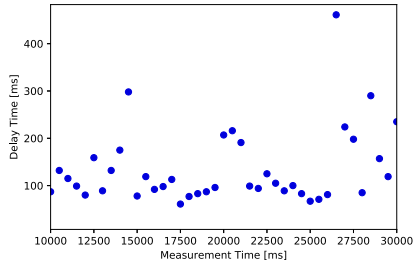


Fig. 7 Delay time between a smartphone and a smartwatch.

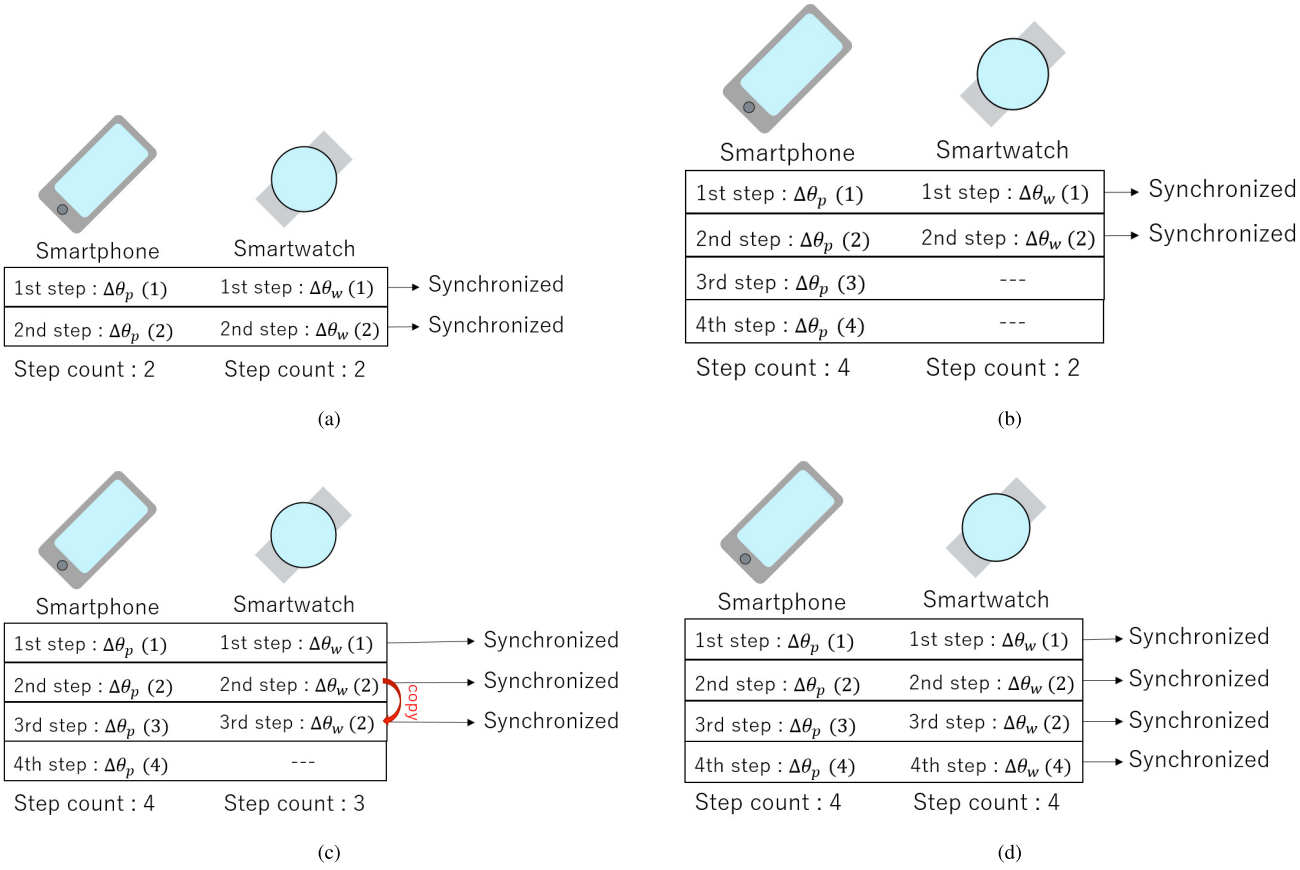


Fig. 8 Example of the step-correction process.

sor data with the smartwatch sensor data due to their transition delays. The proposed method synchronizes the user's direction estimated by the smartphone with that by the smartwatch having the same user's step count.

Let n and m be the current step counts measured by the smartphone and the smartwatch sensor data in Step 1, respectively. We send the user's direction estimated by the smartwatch to the smartphone together with the step count m by Bluetooth, every time the smartwatch estimates the user's direction. Then the smartphone synchronizes the user's direction estimated by the smartphone with that by the smartwatch having the same user's step count.

3.6.2 (2) Step-correction Process

Next, we describe the step-correction process. The proposed method synchronizes the smartphone sensor data with the smartwatch sensor data based on every user's step as mentioned above. However, the smartphone and/or smartwatch may miss the user's step detection, and hence we correct the steps obtained by the smartphone and/or smartwatch in the step-correction process.

The delay time between the smartphone and the smartwatch is smaller than the walking cycle and thus the step count difference between the smartphone and the smartwatch cannot always be larger than one. If it is larger than one, we can consider that the smartphone or smartwatch misses the user's step count. Therefore, the proposed method corrects the smartphone or the smartwatch step count if the difference between the smartphone and the smartwatch step count is larger than one.

Consider the case of $|n - m| \geq 2$ and $n > m$, where n and m are the current step counts measured by the smartphone and the smartwatch sensor data in Step 1, respectively. In this case, we have no $(m + 1)$-th step smartwatch data. Then, the proposed method just copies m-th step smartwatch sensor data to $(m+1)$-th step smartwatch sensor data. The smartphone's $(m + 1)$-th step sensor data is associated with the smartwatch's $(m + 1)$-th step sensor data and we can continue the subsequent processes.

Figure 8 shows an example of the step-correction process. Figure 8 (a) shows the step-synchronization process from the 1st step to the 2nd step. Figure 8 (b) shows the case of missing the step detection. In Fig. 8 (b), "-" shows that the device cannot obtain the sensor data. In this figure, the smartphone obtains the 3rd step sensor data and the 4th step sensor data, but the smartwatch has sensor data only from the 1st step to the 2nd step. In this case, $|n - m| \geq 2$ holds. The smartwatch has no sensor data at the 3rd step and we copy the 2nd step sensor data to the 3rd step sensor data and increase its step count as in Fig. 8 (c). We perform the subsequent processes for the 3rd step sensor data.

Now assume that the smartwatch obtains the 4th step sensor data before the smartphone obtains the 5th step sensor data (Fig. 8 (d)). Then we perform the subsequent processes (Sections 3.7 and 3.8) for the 4th step sensor data. Note that, if the smartwatch does not obtain the 4th step sensor data before the smartphone obtains the 5th step sensor data, $|n-m| \geq 2$ holds and the above step-correction process is performed again. In this example, we assume $n-m \geq 2$ but the same step-correction process will be performed if $m - n \geq 2$ occurs.

Note that, the step count detected cannot usually be larger than

the actual step count as in preliminary experiments (See Table 1 and discussion in Section 4.2). For this reason, we do not drop any sensor data but adopt the step-correction process above.

3.7 Correlation and Integrated Direction Estimation (Step 5)

In Step 3, the cumulative errors in the user's direction estimation can occur by the drift errors [18], [19]. The proposed method corrects these errors by correlating the direction estimation by the smartphone and smartwatch.

3.7.1 Calculating Correlation

When a user turns to the right or left, the angular velocity sensor on both of the smartphone and smartwatch must obtain similar data. Angular velocity of the smartphone and smartwatch always changes due to noises such as thermal noises and the user's small behaviors. However, it is rare that the same noises are added to both angular velocity sensors.

The proposed method takes the covariance [20] of both the direction change values of the smartphone and smartwatch in Step 3. The covariance shows a large value when the two data have similar changes and it also shows a larger value when the original two data have a large value (see **Fig. 12**). By using the covariance, it is possible to recognize that the two direction change values are similar to each other and that the changes are large enough. We can detect whether the user actually turns or not by using the covariance.

Based on this discussion, the proposed method calculates the covariance of the direction change values of the smartphone and smartwatch obtained by Step 3. Then, we recognize that the user actually turns when the covariance is positive and large. This approach prevents unnecessary fluctuations in the user's direction change when the user is only moving in a straight path, and keeps the drift errors small enough.

Let s_i be the covariance of the direction change values of the smartphone and smartwatch at i-th of the user's step. s_i can be defined by

$$s_i = \frac{1}{3}\sum_{k=i-1}^{i+1}((\Delta\theta_p(k) - \Delta\theta_p^{ave}) \times (\Delta\theta_w(k) - \Delta\theta_w^{ave})) \tag{14}$$

where $\Delta\theta_p(i)$ and $\Delta\theta_w(i)$ show the direction change values of the smartphone and smartwatch at i-th of the user's step, respectively, obtained by Step 3 and $\Delta\theta_p^{ave}$ and $\Delta\theta_w^{ave}$ show the averaged direction change values of the smartphone and smartwatch, respectively, from $(i-1)$-th of the user's step to $(i+1)$-th of the user's step.

In Eq. (14), we average the direction change values from $(i-1)$-th of the user's step to $(i+1)$-th of the user's step. This is because it is difficult to calculate the covariance for long steps in real-time positioning. Generally, one walking cycle is composed of two steps since the right foot and the left foot go forward step by step. This means that the walking cycle is completely finished in three steps. Then we adopt Eq. (14) to obtain the covariance.

3.7.2 Integrated Direction Estimation by Correlation

Let $\Delta\theta(i)$ be the average of the direction change values at i-th of the user's step obtained from the smartphone and smartwatch. $\Delta\theta(i)$ is written by

$$\Delta\theta(i) = \frac{1}{2}(\Delta\theta_p(i) + \Delta\theta_w(i)) \tag{15}$$

Based on the covariance s_i, we correct the direction change value at i-th of the user's step as follows:

$$\Delta\theta_{cor}(i) = \begin{cases} \Delta\theta(i) \times a_{pass} & (\text{either } s_i > s_{th},\ s_{i\pm1} > s_{th}, \\ & \quad \text{or } s_{i\pm2} > s_{th} \text{ is satisfied}) \\ \Delta\theta(i) \times a_{cut} & (\text{otherwise}) \end{cases} \tag{16}$$

where a_{pass} and a_{cut} show the coefficients for correction and s_{th} shows the threshold. a_{pass} is set to large enough and a_{cut} is set to small enough. In our experiment, we set $a_{pass} = 1$ and $a_{cut} = 0$.

We take a few steps when we turn to the right or to the left. Then, in Eq. (16), we see the covariance values of the current user's step (i-th step) as well as $(i \pm 1)$-th steps and $(i \pm 2)$-th steps. If one of the covariance values during these steps is larger than the threshold s_{th}, then we recognize that the user turns and we set $\Delta\theta_{cor}(i)$ to $\Delta\theta(i) \times a_{pass}$. Otherwise, we recognize that the user still goes straight and we set $\Delta\theta_{cor}(i)$ to $\Delta\theta(i) \times a_{cut}$.

By summing up $\Delta\theta_{cor}(i)$ from the 0-th of the user's step to n-th of the user's step, we obtain the relative direction difference up to n-th of the user's step:

$$\theta_n = \sum_{i=0}^{n}\Delta\theta_{cor}(i) \tag{17}$$

As above, the correlation is calculated between smartphone and smartwatch sensor data. In this calculation, the proposed method uses direction changes per each step of both the smartphone and the smartwatch. These direction changes are calculated with angular velocity values at i-th step. Therefore, we do not have to need synchronization of sampling timing, but we synchronize *step timing* because the proposed method uses the sensor data that depend on steps, as described above.

3.7.3 Threshold Setting

In Eq. (16), how to set the threshold s_{th} is important. **Figure 10** shows the covariance values when walking on a straight path of approximately 30 m as in the red line of **Fig. 9** using Google's smartphone Pixel 3 and Mobvoi's smartwatch TicWatchPro. **Figure 11** shows the covariance values when walking on a straight path of approximately 15 m, turning by 90 degrees to the left, and going straight for approximately 15 m as in the blue line of Fig. 9. In Fig. 10 and Fig. 11, the smartphone's carrying modes are a) holding it in the right hand in front of the body (Right-Hand), b) swinging it in the right hand at the side of the body

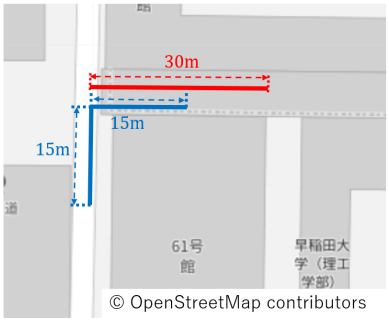
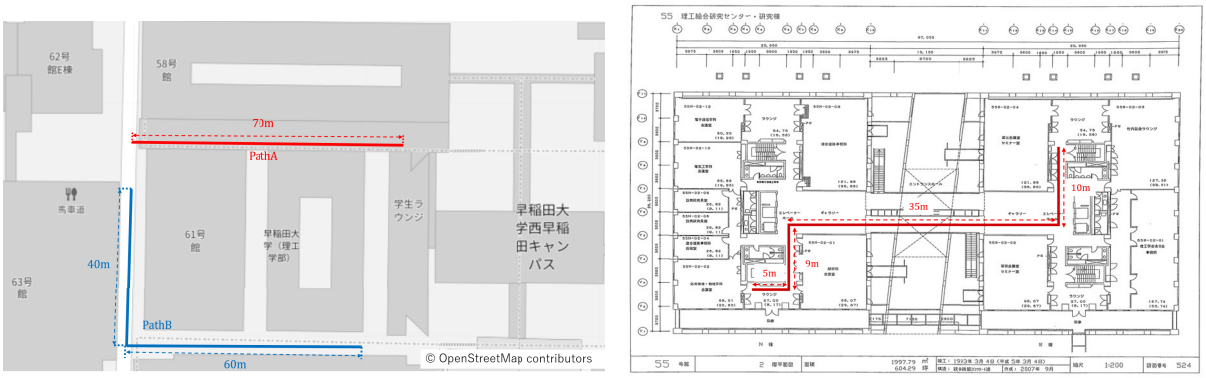


Fig. 9 Routes used in the experiment.



(a) Path A and Path B. (b) Path C.

Fig. 13 Routes used in the experiment.

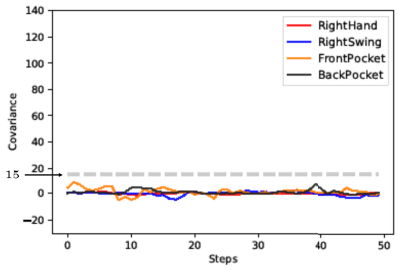


Fig. 10 Covariance when walking straight.

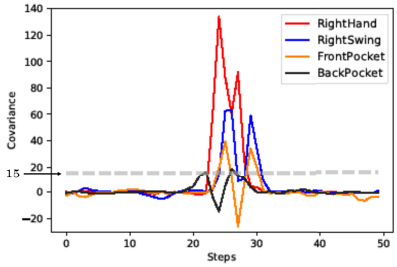


Fig. 11 Covariance when turning.

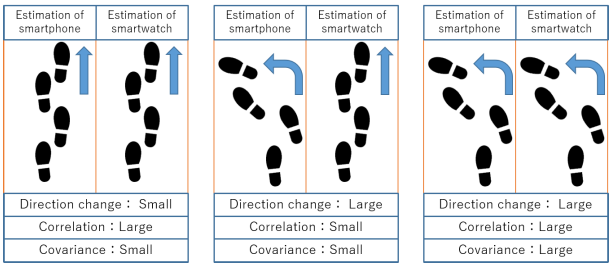


Fig. 12 Relationship between direction change and covariance.

(RightSwing), c) putting it in the front right trouser pocket (FrontPocket), and d) putting it in the back right trouser pocket (BackPocket). The smartwatch is worn on the left wrist. In Fig. 10 and Fig. 11, the horizontal axis shows the covariance and the vertical axis shows the user's steps.

As Fig. 10 shows, the covariance on going straight becomes approximately 15 or less. On the other hand, as Fig. 11 shows, the covariance on turning is over 15 in any carrying modes. Hence we set s_{th} to 15.

3.8 Position Estimation (Step 6)

Finally, the user's position $\vec{p_n} = (x_n, y_n)$ at n-th of the user's step is estimated by

$$\begin{cases} x_n = l \times \cos\theta_n + x_{n-1} \\ y_n = l \times \sin\theta_n + y_{n-1} \end{cases} \tag{18}$$

Note that it is assumed that the stride length l is given.

4. Evaluation

4.1 Condition

We have implemented the proposed method as an Android application in Java and several evaluation experiments have been conducted using Google's smartphone Pixel 3 [14] and Mobvoi's smartwatch TicWatchPro [9]. In the experiments, the following three types of paths, Path A, Path B, and Path C, are used.

Path A: A straight path of approximately 70 m as in the red line of **Fig. 13** (a).

Path B: A straight path of approximately 40 m, turning by 90 degrees to the left, and going straight for approximately 60 m as in the blue line of Fig. 13 (a).

Path C: A straight path of approximately 5 m, turning by 90 degrees to the left, going straight for approximately 9 m, turning by 90 degrees to the right, going straight for approximately 35 m, turning by 90 degrees to the left, and going straight for approximately 10 m as in the red line of Fig. 13 (b).

In all the experiments, we have carried the smartphone in various modes and worn the smartwatch on the left wrist. We set the stride length to be 70 cm based on the preliminary experiment [*4]. The types and purposes of the experiments are shown below.

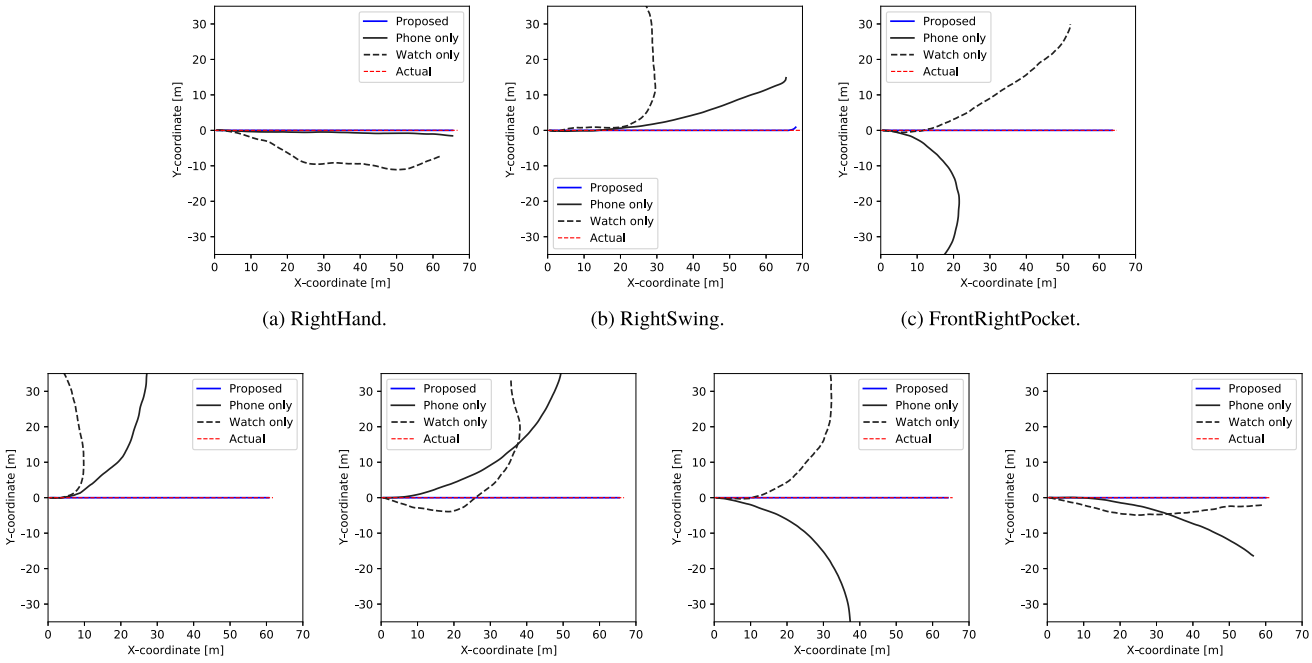
(1) Step count evaluation

Firstly, we estimate the step counts by using the proposed method on Path A and Path B. Then we compare the estimated step count with the actual step counts. We can confirm how accurately the proposed method estimates the user's step count.

(2) Direction estimation evaluation

Next, we estimate the route trajectory by the proposed method on Path A, Path B, and Path C. Then we compare the estimated route trajectory with the actual route. We can

[*4] In our preliminary experiment, the user walked 20-step forward along the straight path and the average stride length was calculated using the walking distance. Then we obtained the stride length to be 70 cm.



(a) RightHand. (b) RightSwing. (c) FrontRightPocket.

(d) FrontLeftPocket. (e) BackRightPocket. (f) BackLeftPocket. (g) Bag.

Fig. 14 Trajectory when walking on Path A.

Table 1 The results of step count estimation.

Devices		Path A (100 steps)		Path B (150 steps)	
		Estimated steps	Error	Estimated steps	Error
Smartphone's carrying modes	RightHand	101	1.0%	151	0.7%
	RightSwing	100	0.0%	149	−0.7%
	FrontPocket	98	−2.0%	148	−1.3%
	BackPocket	96	−4.0%	148	−1.3%
	Bag	100	0.0%	150	0.0%
Smartwatch		100	0.0%	149	−0.7%

confirm how accurately the proposed method recognizes the user's turn and how much the proposed method reduces the drift errors.

(3) Comparisons

As described in Section 1, no PDR method other than the proposed method uses a smartphone and a smartwatch simultaneously. On the other hand, the existing method [7] enables the user's indoor position estimation in various of the smartphone's carrying modes. In addition, as in the proposed method, the existing method [7] estimates the user's direction using only smartphone sensors, without using the geomagnetic fields. Hence, we compare the proposed method with Ref. [7]. In Ref. [7], we also obtain the route trajectories similarly on Path A, Path B, and Path C.

4.2 Step Count Evaluation

Table 1 shows the results of step count estimation by the smartphone and the smartwatch in Step 1 of the proposed method. In this experiment, the smartphone's carrying modes are a) holding it in the right hand in front of the body (RightHand), b) swinging it in the right hand at the side of the body (RightSwing), c) putting it in the front right trouser pocket (FrontPocket), d) putting it in the back right trouser pocket (BackPocket), and e) putting it in the shoulder bag (Bag) [*5]. The smartwatch is worn on the left wrist.

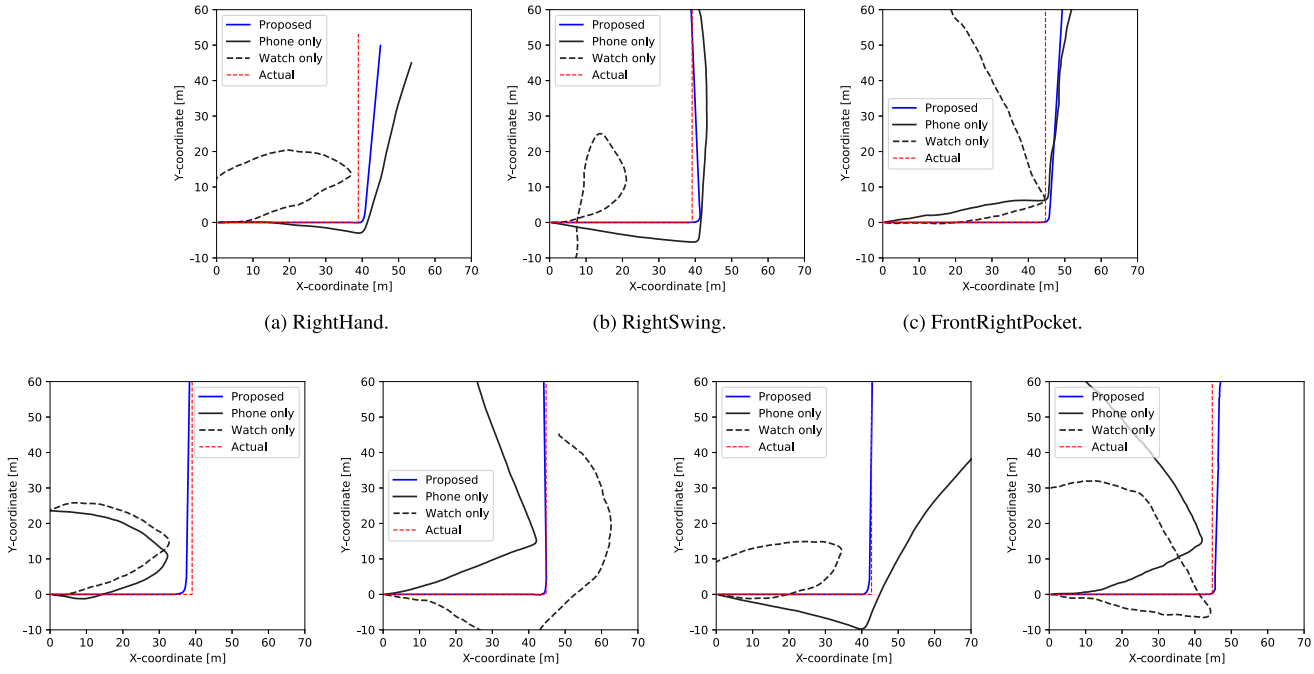
As in Table 1, in any smartphone's carrying modes, the errors between the estimated step counts and the actual step counts are around 1%, which definitely shows that the proposed step count estimation can be done with high accuracy. In the case of the smartwatch, the errors are also very small. These errors are mainly caused by the start and end of the walk, where the proposed method cannot well detect the acceleration and angular velocity peaks.

4.3 Direction Estimation Evaluation

Figure 14, **Fig. 15** and **Fig. 16** show the route trajectories of Path A, Path B, and Path C, respectively, obtained by the proposed method and the actual routes. In this experiment, we have also obtained the route trajectories only using the smartphone and only using the smartwatch. In these cases, we use the proposed method without combining the smartphone and the smartwatch, just using one of them. The results are also shown in Fig. 14, Fig. 15, and Fig. 16 (Phone only and Watch only).

In Fig. 14, Fig. 15, and Fig. 16, the smartphone's carrying modes are a) holding it in the right hand in front of the body (RightHand), b) swinging it in the right hand at the side of the body (RightSwing), c) putting it in the front right trouser pocket (FrontRightPocket), d) putting it in the front left trouser pocket (FrontLeftPocket), e) putting it in the back right trouser pocket (BackRightPocket), f) putting it in the back left trouser pocket (BackLeftPocket), and g) putting it in the shoulder bag (Bag). The smartwatch is worn on the left wrist. The X-axis in Fig. 14, Fig. 15, and Fig. 16 shows the straight path direction and the Y-axis shows the direction perpendicular to the straight path direction.

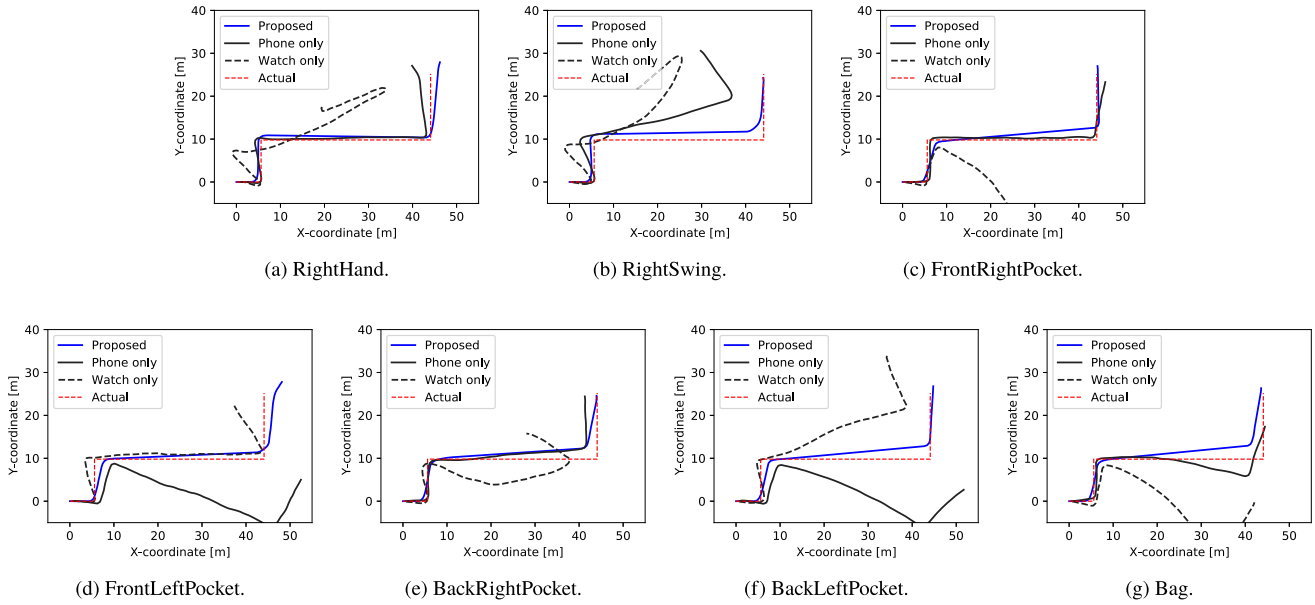
[*5] LeftHand and LeftSwing modes (holding the smartphone in the *left* hand) are discussed in Section 4.5.1.



(a) RightHand. (b) RightSwing. (c) FrontRightPocket.

(d) FrontLeftPocket. (e) BackRightPocket. (f) BackLeftPocket. (g) Bag.

Fig. 15 Trajectory when walking on Path B.



(a) RightHand. (b) RightSwing. (c) FrontRightPocket.

(d) FrontLeftPocket. (e) BackRightPocket. (f) BackLeftPocket. (g) Bag.

Fig. 16 Trajectory when walking on Path C.

In the case of Path A, as Fig. 14 shows, the proposed method recognizes a straight path, and estimates the user's positions with almost no errors in any smartphone's carrying modes. In the case of Path B and Path C, the proposed method also estimates the user's positions with almost no errors in several of the smartphone's carrying modes. The errors are much reduced by combining the smartphone and smartwatch, compared with just using only the smartphone, or only the smartwatch.

Note that the proposed method only uses the smartphone and smartwatch sensor data. As in Refs. [21], [22], if we further apply another external method to our proposed method, the errors can be reduced and a more accurate user's position estimation will be realized.

4.4 Comparison

Table 2 shows the comparison results between Ref. [7] and the proposed method. In this experiment, the smartphone's carrying modes are the same as in the previous experiment. The smartwatch is worn on the left wrist.

In Table 2, the average estimation error err is obtained by Eq. (19) below:

$$err = \frac{1}{n}\sum_{i=0}^{n}\sqrt{(x_i^{est} - x_i^{act})^2 + (y_i^{est} - y_i^{act})^2} \tag{19}$$

where n shows the total number of steps, x_{est}^i and y_{est}^i show the X-coordinate and Y-coordinate of the estimated user's position at i-th step, respectively, and x_i^{act} and y_i^{act} show the X-coordinate and

Table 2 Comparison of estimated position error *err*.

		err (cm)							Reduction rate (%)
		(a) RightHand	(b) RightSwing	(c) FrontRight Pocket	(d) FrontLeft Pocket	(e) BackRight Pocket	(f) BackLeft Pocket	(g) Bag	
Path A	Proposed	0.0	2.6	0.0	0.0	0.0	0.0	0.0	99.94
	Ref. [7]	89.8	271.9	455.4	1,452.0	1,579.0	559.4	333.5	
Path B	Proposed	125.5	64.0	712.8	80.2	433.3	450.1	478.3	75.14
	Ref. [7]	625.1	171.2	1,839.1	612.4	1,980.3	1,899.2	2,300.7	
Path C	Proposed	118.1	195.2	347.1	274.7	338.1	335.5	378.2	71.16
	Ref. [7]	172.3	2,380.4	431.6	479.5	1,289.5	550.9	1,576.0	

Y-coordinate of the actual user's position at i-th step, respectively. Table 2 demonstrates that the proposed method successfully reduces the average estimation errors by 99.94% on Path A, 75.14% on Path B, and 71.16% on Path C. In any of the smartphone's carrying modes, the estimation errors can be much reduced. As a result, the proposed method can satisfy (Condition A) to (Condition C) in Section 1.

4.5 Discussion

4.5.1 LeftHand and LeftSwing modes

In this paper, we assume that (1) the user wears a smartwatch on the wrist and (2) he/she slightly swings it at the side of the body. By adopting this assumption, we can clearly obtain the periodical sensor data from the smartwatch and hence PDR can be successfully realized. When the smartwatch is not well swung, for example, it is held in front of one's body and almost fixed there, the step count cannot be well obtained. This is because the step count of the smartwatch is calculated based on the angular velocity sensor data as described in Section 3.3.2. How to solve this problem is one of the major future projects. Combining the angular velocity data and acceleration data may be one of the solutions.

Note that, when a smartphone is held in front of the body as in the RightHand mode, still its step count can be obtained. This is because the step count of the smartphone is calculated based on the acceleration sensor data.

Now we discuss how our proposed method works when the smartphone and smartwatch are held and worn in the *same* hand.

LeftHand mode

The proposed method cannot well estimate the user's position when holding the smartphone and the smartwatch in the same hand. The RightHand mode refers to holding the smartphone in the right hand in front of the body. The LeftHand mode refers to holding the smartphone in the left hand in front of the body. The smartwatch is always worn on the left wrist. When the user holds the smartphone and the smartwatch in the same hand, both the smartphone and the smartwatch are held in front of the body and almost fixed there. In this situation, the user does not swing the smartwatch at the side of the body.

Figure 17 shows the estimation result of RightHand and LeftHand modes. The experimental conditions are the same as the ones in Section 4.1. As Fig. 17 clearly indicates, the proposed method fails to estimate the user's positions when he/she holds the smartphone and the smartwatch in the same hand.

LeftSwing mode

The proposed method can estimate the user position even when holding the smartphone and the smartwatch in the same hand.

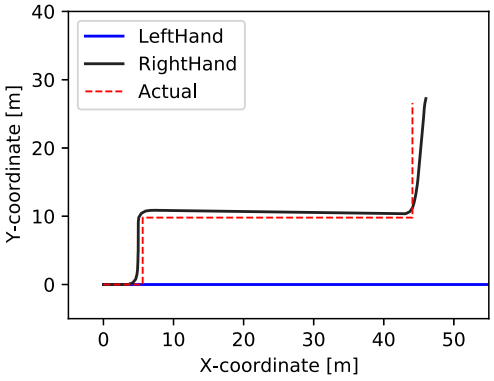


Fig. 17 Estimation result of LeftHand and RightHand modes (the smartwatch is worn on the left wrist).

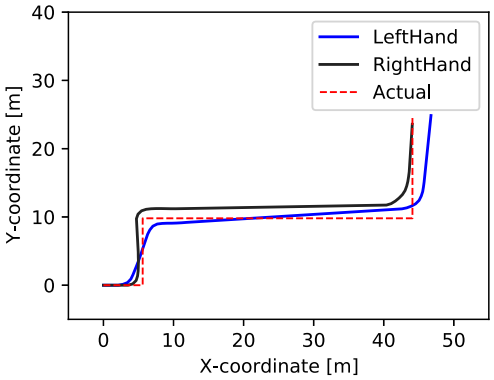


Fig. 18 Estimation result of RightSwing and LeftSwing modes (the smartwatch is worn on the left wrist).

The RightSwing mode refers to holding the smartphone in the right hand and swinging it at the side of the body. The LeftSwing mode refers to holding the smartphone in the left hand and swinging it at the side of the body. The smartwatch is always worn on the left wrist.

Figure 18 shows the estimation result of RightSwing and LeftSwing modes. Figure 18 clearly shows that the proposed method successfully estimates the user's positions in both RightSwing and LeftSwing modes. This is because the smartwatch can be swung in both RightSwing and LeftSwing modes and the sensor data from the smartwatch can be properly obtained. For comparison purposes, we calculate the average estimation error *err* given by Eq. (19). **Table 3** summarizes the results. The conventional method [7] only uses the smartphone and we hold it in the left hand. Table 3 indicates that the estimation errors of RightSwing and LeftSwing modes are not much different from those of other modes, referring to Table 2. They are much smaller than that of the Swing mode of Ref. [7]. The proposed method successfully applies to the RightSwing and LeftSwing modes.

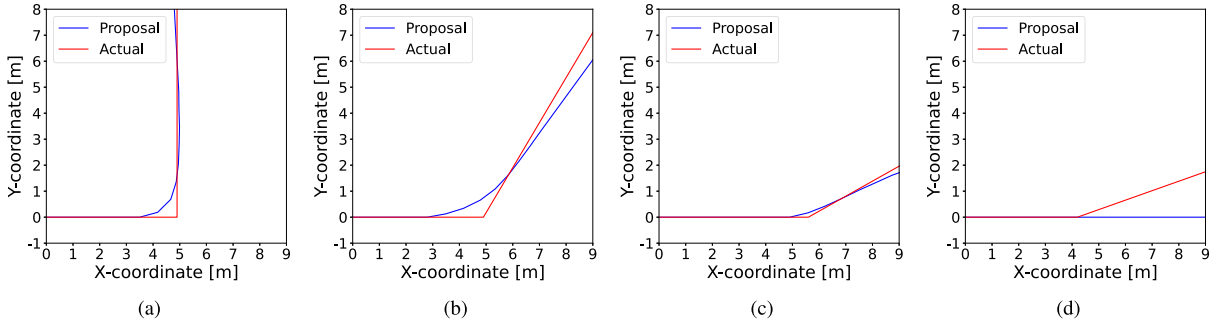


Fig. 19 Estimation result of turning left in RightSwing mode. (a) Turning by 90 degrees. (b) Turning by 60 degrees. (c) Turning by 30 degrees. (d) Turning by 20 degrees.

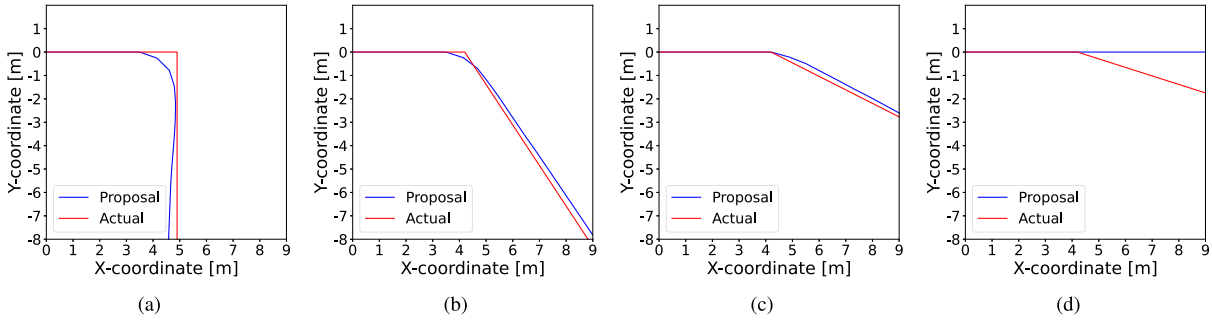


Fig. 20 Estimation result of turning right in RightSwing mode. (a) Turning by 90 degrees. (b) Turning by 60 degrees. (c) Turning by 30 degrees. (d) Turning by 20 degrees.

Table 3 Average estimation errors in RightSwing and LeftSwing modes.

Method and mode	Errors [cm]
Proposed in RightSwing mode	195.2
Proposed in LeftSwing mode	321.6
Ref. [7] (Swing mode)	2,380.4

4.5.2 Turning to an Angle Other than 90 Degrees

As described in Section 3.7, the proposed method first recognizes whether the user *turns or not* by using the covariance of the direction change values of the smartphone and smartwatch, using Eq. (16). At that time, we use the threshold values for the covariance. Once the proposed method recognizes that the user *turns*, it calculates *how many degrees* the user turns based on Eq. (17). This means that the proposed method can be applied to the case where the user turns at an angle other than 90 degrees.

However, in the case where the user turns at a very small angle, the proposed method cannot recognize that the user turns, because the covariance of the direction change values of the smartphone and smartwatch becomes too small.

We experimentally evaluate whether the proposed method can be applied to the case where the user turns at an angle other than 90 degrees. In the experiment, the user has the smartphone in the RightSwing mode and the smartwatch is worn on the left wrist. Here, the RightSwing mode is chosen because it must be one of the most usually used modes.

Figure 19 and **Fig. 20** show the results of the position estimation with turning. In Fig. 19, the user turns to the left by 90 degrees, 60 degrees, 30 degrees, and 20 degrees. In Fig. 20, the user turns to the right by 90 degrees, 60 degrees, 30 degrees, and 20 degrees. In Fig. 19 and Fig. 20, the blue lines show the estimation results by the proposed method while the red lines show the actual route on which the user walks. Figures 19 (a) to (c) and Figs. 20 (a) to (c) well demonstrate that the proposed method can estimate the user's positions. However, the proposed method cannot recognize the user's turning in case of turning to the left or right by 20 degrees as in Fig. 19 (d) and Fig. 20 (d). The results are almost the same in the other carrying modes. According to these results, the proposed method can estimate the user's turning at the degree of approximately 30 degrees or more.

Note that, the proposed method may not estimate the user's positions in a gentle curve because it must be too difficult to distinguish the gentle curve from drift errors. How to solve this problem is also one of the important future works. Map matching [22] can be used to solve this problem. For example, we define the aisles in the gentle curve in advance and correct the user's positions based on them.

4.5.3 User's Stride Length

Our proposed method does not realize the user's stride length estimation dynamically, i.e., the stride length is not adoptively changed during PDR. However, many methods for dynamical stride length estimation have been proposed so far. For example, the stride length is dynamically estimated by using vertical acceleration [23] and by using the frequency of walking [3].

We expect that dynamical stride length estimation is realized in our proposed method by incorporating these methods into our proposed method. This is also one of important future works.

5. Conclusions

In this paper, we have proposed a PDR method based on a combination of a smartphone and a smartwatch. The proposed method successfully estimates the user's indoor positions in various of the smartphone's carrying modes. The positioning errors are reduced by approximately 82.1% on average compared to the existing method.

In the future, we will combine our method with other PDR

methods such as map matching to further accurately estimate the user's positions. Combining indoor positioning and outdoor positioning is another future work.

Acknowledgments This work was supported in part by JST CREST Grant Number JPMJCR19K4, Japan.

References

[1] Kamisaka, D., Muramatsu, S., Iwamoto, T. and Yokoyama, H.: Dead reckoning method by hand for pedestrian navigation system (in Japanese), *Journal of Information Processing Society of Japan*, Vol.52, No.2, pp.558–570 (2011).

[2] Lee, J.-S. and Huang, S.-M.: An experimental heuristic approach to multi-pose pedestrian dead reckoning without using magnetometers for indoor localization, *IEEE Sensors Journal*, Vol.19, No.20, pp.9532–9542 (2019).

[3] Tian, Q., Salcic, Z., Wang, K.I-K. and Pan, Y.: A multi-mode dead reckoning system for pedestrian tracking using smartphones, *IEEE Sensors Journal*, Vol.16, No.7, pp.2079–2093 (2016).

[4] Xu, L., Xiong, Z., Liu, J., Wang, Z. and Ding, Y.: A novel pedestrian dead reckoning algorithm for multi-mode recognition based on smartphones, *Remote Sensing*, Vol.11, No.3 (2019).

[5] Park, S.Y., Heo, S.J. and Park, C.G.: Accelerometer-based smartphone step detection using machine learning technique, *Proc. 2017 International Electrical Engineering Congress* (*iEECON*), pp.1–4 (2017).

[6] Klein, I., Solaz, Y. and Ohayon, G.: Pedestrian dead reckoning with smartphone mode recognition, *IEEE Sensors Journal*, Vol.18, No.18, pp.7577–7584 (2018).

[7] Hoshi, H., Fuji, M., Hatano, H., Ito, A. and Watanabe, Y.: Traveling direction estimation for pedesrian dead reckoning using smartphone (in Japanese), *Journal of Information Processing Society of Japan*, Vol.57, No.1, pp.25–33 (2016).

[8] Ministry of Internal Affairs and Communications, Japan, 2018 White Paper on Information and Communications in Japan, available from ⟨https://www.soumu.go.jp/johotsusintokei/whitepaper/ja/h28/pdf/n2200000.pdf⟩.

[9] Mobvoi: TicWatch Pro, available from ⟨https://www.mobvoi.com/jp/pages/ticwatchpro⟩.

[10] Loh, D., Zihajehzadeh, S., Hoskinson, R., Abdollahi, H. and Park, E.J.: Pedestrian dead reckoning with smartglasses and smartwatch, *IEEE Sensors Journal*, Vol.16, No.22, pp.8132–8141 (2016).

[11] Correa, A., Diaz, E.M., Ahmed, D.B., Morell A. and Vicario, J.L.: Advanced pedestrian positioning system to smartphones and smartwatches, *Sensors*, Vol.11, No.11 (2016).

[12] Wakaizumi, T. and Togawa, N.: An Indoor Positioning Method using Smartphone and Smartwatch Independent of Carrying Modes, *Proc. IEEE 39th International Conference on Consumer Electronics* (*ICCE*) (2021).

[13] Apple: iPhone XR, available from ⟨https://www.apple.com/jp/iphone-xr/specs/⟩.

[14] Google: Google Pixel 3, available from ⟨https://store.google.com/jp/product/pixel_3⟩.

[15] Apple: Apple Watch, available from ⟨https://www.apple.com/jp/watch/⟩.

[16] Zou, H., Chen, Z., Jiang, H., Xie, L. and Spanos, C.: Accurate indoor localization and tracking using mobile phone inertial sensors, WiFi and iBeacon, *Proc. 2017 IEEE International Symposium on Inertial Sensors and Systems* (*INERTIAL*), pp.1–4 (2017).

[17] Li, Y., Zhuang, Y., Lan, H., Zhou, Q., Niu, X. and El-Sheimy, N.: A hybrid WiFi/magnetic matching/PDR approach for indoor navigation with smartphone sensors, *IEEE Communications Letters*, Vol.20, No.1, pp.169–172 (2015).

[18] Borenstein, J., Ojeda, L. and Kwanmuang, S.: Heuristic reduction of gyro drift for personnel tracking systems, *The Journal of Navigation*, Vol.62, No.1, pp.41–58 (2009).

[19] Nozaki, J., Hiroi, K., Kaji, K. and Kawaguchi, N.: Compensation scheme for PDR using sparse location and error model, *Proc. 2017 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proc. 2017 ACM International Symposium on Wearable Computers*, pp.587–596 (2017).

[20] Rice, J.A.: *Mathematical Statistics And Data Analysis*, *Cengage Learning* (2006).

[21] Anzai, K., Okajima, S. and Tsubokawa, H.: The estimate of the indoor position that used a smartphone and the suggestion of the walk navigation system (in Japanese), *Proc. Multimedia, Distributed, Cooperative, and Mobile Symposium*, pp.921–927 (2011).

[22] Shin, S.H., Park, C.G. and Choi, S.: New map-matching algorithm using virtual track for pedestrian dead reckoning, *ETRI Journal*, Vol.32, No.6, pp.891–900 (2010).

[23] Weinberg, H.: Using the ADXL202 in pedometer and personal navigation applications, *Analog Devices AN-602 Application Note*, Vol.2, No.2, pp.1–6 (2002).

Tomoya Wakaizumi received his B.Eng. degree from Waseda University in 2020 in communications and computer engineering, where he is working towards his M.Eng. degree. His research interests include indoor positioning, especially pedestrian dead reckoning with smartphones.

Nozomu Togawa received his B.Eng., M.Eng., and Dr.Eng. degrees from Waseda University in 1992, 1994, and 1997, respectively, all in electrical engineering. He is presently a Professor in the Department of Computer Science and Communications Engineering, Waseda University. His research interests are VLSI design, graph theory, and computational geometry. He is a member of IEEE, ACM, and IEICE.