Electronic Preprint for Journal of Information Processing Vol.27

Regular Paper

EasyTrack: Zero-Calibration Smart-Home Tracking System

Nathavuth Kitbutrawat[1,a)] Hirozumi Yamaguchi[1,b)] Teruo Higashino[1,c)]

Received: September 21, 2018, Accepted: April 9, 2019

Abstract: Location-based services in household enable not only estimating activities but also detecting the accident location of residents including children, parents and elderly people. Furthermore, home management systems make use of location information of residents for encouraging residents to live comfortably in their own homes. Passive infrared binary motion sensors are widely used in home tracking systems because of low energy consumption and flexibility on deployment. However, it's hard for non-specialists like residents to manage such location information of sensors which are very important for tracking systems. To mitigate human efforts and errors, we propose a sensor localization method to automatically identify the location of multiple binary motion sensors in a house from the observed motion detection event sequences by binary sensors. The method finds the movement patterns and characteristics of sojourn time of residents to identify the rooms where those sensors are located as well as the proximity relations among sensors assisted by prior knowledge such as from floorplan. The experimental results in actual houses show that our method can estimate the location of sensors placed close to the anchor locations within a one-day observation, and the accuracy of our approach is above 80% after three-day observation.

Keywords: unsupervised learning, localization, home passive sensor

1. Introduction

The drastic increase in the elderly population around the world [1] affect a rise in the number of elderly in houses. The main challenge is to support the elderly when they live alone during other people go outside on workday. The activity recognition and localization are the key technologies to monitor the activities and location of elderly in houses for their safer life.

There are several sensors such as cameras and passive sensors, which are used to estimate the activities and location of human in buildings. Basically, indoor human tracking should not require users to wear devices such as smartwatches, since they are too invasive for their daily living. Therefore, passive sensors, such as motion sensors for detecting movement of human and contact switches for detecting events when residents open doors [2], [3], [4], [5], are more reasonable solutions to measure the presence and direct motion of those people. The locations of non-embedded sensors are the important information for predicting the activities of residents by analyzing the sequence of events with probabilistic methods.

According to our past research [2], the location of such passive motion sensors is useful information for estimating activities. However, obtaining such information requires technical labor to configure. This is in general because every resident is not able to understand how to place the sensors. Some residents will make some mistakes if they misunderstand the manual. Moreover the configuration process makes more burdens on residents if they need to place plenty of sensors in their houses as instructed. For example, when elderly who will be unfamiliar with technology would like to deploy the system by themselves from cost perspective. We cannot assume that those people will use the GUI tools correctly. In order to deploy sensors in this situation, the technical employee needs to be sent to configure the system in each customer's house.

There are several drawbacks to sending technicians to configure sensors at home. Firstly, the company selling the system will need a larger budget for training and sending the technicians as well as their salary. Secondly, there is always privacy and security concern. Many residents do not want somebody to come into their houses and survey their private rooms. Even worse, the location of sensors may change after configuration by misplacement after room cleaning etc. It is desirable that the tracking systems have a self-configuring function to allow residents to easily install the system by themselves.

In this paper, we propose a method to localize the binary motion sensors by room level at home to provide a self-calibration function to tracking systems, because the room-level location of sensor is enough for such location services in smart home environment. In particular, there are some services in smart-home systems which require the location-service. Firstly, the energy management system manages the electric devices/appliances (e.g., turn on/off light and air condition) depending on human presence. Secondly, the fall detection which observes the location of elderly people. Starting this in more detail, we may regard that the elderly is falling if that person has not moved on the cor-

[1] Graduate School of Information Science and Technology, Osaka University, Osaka 565–0871, Japan
[a)] nat-kit@ist.osaka-u.ac.jp
[b)] h-yamagu@ist.osaka-u.ac.jp
[c)] higashino@ist.osaka-u.ac.jp

ridor for long time. Unlike the localization in a large building, the location services for household do not require the precise location of presence sensors. Therefore, the system which detect the resident perform activity by room level is acceptable. The objective is to reduce the configuration and verification efforts for the tracking system, and our method only requests the resident to give some information (e.g., floorplan) about their house to the system. Then it analyzes human detection events from those motion sensors to find the locations of sensors. The method does not require any supervised training procedure before the operation. Instead, it finds the sensor location only by monitoring the events from the sensors for a couple of days. This approach leverages a few sensors placed in some specific locations and the prior knowledge gained from floorplan analysis to generate an indoor digital map from a floorplan image submitted offline to the service or some other ways. Then the method calculates location similarity between every pair of sensors, and maps those sensors onto the installation places in the indoor map.

We have conducted two experiments using 20 sensors installed in (i) a 2-story house with three family members and (ii) an apartment with a single elderly. As a result, it could estimate the sensor locations in the neighbor location of the sensor in specific locations within a one day operation, and more than 80% of sensors can be mapped to their locations correctly after 3 days operation.

2. Related Work

The location information of sensors is significant to both human activity recognition and human localization. We survey those technologies and then address our motivation to solve the problem of identifying the location of sensors within the house automatically.

2.1 Activity Recognition

There are several ways for performing human activity recognition. A typical one is to use computer vision techniques for analyzing images from cameras [6], [7], [8]. However, the major problem of the methods which use RGB camera is privacy concern. Hence some researchers use depth cameras [6], [7] instead of RGB cameras. Nevertheless, many people have strong feeling of resistance to deploying cameras in the home especially in private living spaces.

Research on wearable device-based activity recognition [9], [10] leverage motion captured from embedded sensors on the devices such as accelerometers and gyroscope to estimate the activity of wearing persons. Atallah et al. [10] propose the relationship between location of a sensor on a person's body and the activity to increase accuracy in activity recognition. Nevertheless, this method requires residents to wear devices and some residents such as elderly and children may forget to wear them in their houses. Hence such approaches that use low-cost infrastructure-based activity recognition [3], [4], [11] are more reasonable in the home.

In the infrastructure-based activity recognition, the researchers deploy binary sensors to analyze the stream of events from sensors when residents walk past passive infrared sensors and interact with sensors to predict the location of humans and their activities [3], [4], [12]. A typical research effort by Hoque et al. [4] proposes a method to detect activities performed during another activity. For example, it may happen that residents take a break from watching television to walk to the toilet and come back to the living room to resume watching television. Thus, they divide activities into short period activities by grouping the sensor events by the time and location of firing sensors, and cluster the set of sensor events which frequently occur together. After that Emi and Stankovic extended the Hoque's work to deal with the multi-resident environment [3]. The idea is to regard the activities occurring in different rooms at the same time as those performed by different users. Moreover the work by Krishnan et al. [5] increases the accuracy of activity recognition by predicting events coming from a single activity by using a hybrid technique between the time window and mutual information.

2.2 Indoor Localization

To realize indoor human tracking systems, plenty of methods leveraging Wi-Fi signals in indoor environment have been proposed so far to tackle the indoor human tracking problem [13]. The basic method to estimate the location of human relies on the distance from that person's Wi-Fi device to at least 3 surrounding APs estimated by RSSI [14] or fingerprint databases. However, for accurate RSSI-based indoor localization, a number of APs should ubiquitously be installed around the building, and the calibration and configuration for all of these APs requires numerous effort. To mitigate the cost of calibration and configuration process, some methods configure the location of anchors by requesting a user to carry a Wi-Fi device for collecting the RSSI signals involved in the building. For example, the methods by Chintalapudi et al. [15] and Makki et al. [16] request people to carry Wi-Fi devices for surveying the signal propagation. It measures the distance between those devices and Wi-Fi APs by using the signal propagation model. Chintalapudi et al. leverage the GPS-fixed locations acquired when tester walk near window to calculate the actual location of Wi-Fi APs, while Makki et al. leverage the time differences of arrival to calculate the location of Wi-Fi transmitters. The other method is proposed by Jun et al., which generates the RSSI propagation model acquired by smartphones when phone holders walk in the building. Their technique is able to build an RSSI propagation model in the building without any annotation [17].

Although the Wi-Fi based localization is cost effective, it may not fit in a small area such as household, which requires finer-grained localization. RFID-based localization [18], [19] is considered useful in house, e.g., Ref. [18] localizes elderly residents who carry RF-ID tags. However, this forces elderly to wear tags, which is also invasive. Therefore, the device-free localization which leverages the signal reflection on the human body to estimate the location of human [20], [21] is more desirable. Specifically, some research place Wi-Fi transmitters and receivers in a house, and estimate the location of a resident by considering the change in RSSI values. To overcome the shadowing problem in device-free localization, [21] applies the signal multi-path profile known as "radio tomographic imaging (RTI) [22]", where several Wi-Fi receivers are deployed in one room. However, the major

problem for device-free localization technique is the human effort to obtain the "passive radio map" [20]. Although experts can make precise 3D models for houses or buildings to generate the passive radio map by simulation [23], the 3D model building task is difficult for general users. Recently CSI-based human tracking [24] attracts more attention, but they basically require configuration and calibration to fit for indivisual environment.

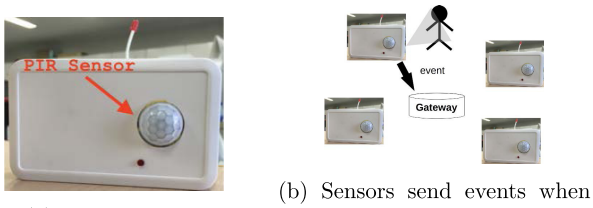
Passive sensor-based localization [12], [25] is able to detect the location of resident and estimate trajectories without wearing devices. Those techniques also require the location of sensors for estimating the location of human. Nevertheless, they cannot use the existing calibration methods such as RSSI-based sensor localization because the sensors are not able to play the role of APs due to functions and battery limitation.

2.3 Easy-to-deploy Smart Home

Some research studies provide easy-to-deploy mechanisms to ubiquitous systems which encourage the end users to deploy those systems by themselves. Specifically, Ref. [26] provides a sensor module attached with RFID to enable a way to install the general function and also provide a method to allow user to make the custom function for the sensor. Reference [27] proposed the sensor and gateway (coordinator) should already be paired before being sent to an end user, thus the end user does not have a task for adding the sensor to the system, and provide the easy-to-used software to encourage the end user to deploy the system easily. However, those methods do not mention how the end users configure the location of sensors. In our experience, there are many people who can use technology such as smartphones but they do not understand the configuration method. The closely related work is Ref. [28] which proposes the room-level accuracy for localization of Bluetooth anchors. They provide a method which is easy for a non-technical user to configure the room-level localization of anchor nodes by carrying a smartphone to collect RSSI in every room and label the RSSI data with their location (which room they collect data). Nevertheless, their target system requires users to carry some devices to estimate the locations, which cannot be applied in our assumed situation where no device and no resident is available for calibration.

2.4 Our Contributions

In summary, all the indoor localization techniques leverage the location of sensors to predict the location of residents. Therefore, each system has been configured with the location of sensors before it is operated. In this viewpoint, our main contributions can be summarized into a 3-fold approach. Firstly, this is the first approach to localize binary sensors without RF signal manner and human effort. We use only the sensor events (presence of human) with timestamps to estimate the location of each sensor. Secondly, our approach can find the sensor locations without need of labeled data and supervised techniques. Nevertheless, it can deal with multi-resident environment. Finally, the accuracy has been validated by actual in-home experiments. To the best of our knowledge, the technique for predicting binary sensor locations in home environment has never been proposed.



(a) Our prototype (b) Sensors send events when detecting human

Fig. 1 PIR Sensor (Binary Sensor).

3. System Architecture

3.1 Human Presence Sensors

According to our past experience [2], we use portable passive infrared (PIR) binary motion sensors (simply called *binary sensors*) which have a detection range 7 meters and the detect angle 110 degrees to recognize the activities. For example, the events from binary sensors in the kitchen represent the cooking activity and the events from those sensors in the living room represent that the residents relax in the living room. As we discussed in Section 1, the main problem is a method to assign the location of binary sensors without configuration and calibration. We focus on finding the location of binary motion sensors as they are penetrating to many households and many products have already been in market because of cost-effective. Another benefit is that the presence of residents can be detected without a need of human-object interaction. Since power line supply will be an obstacle for their location-free installation, sensors are built in wireless nodes with low-power communication technology such as ZigBee and are operated by small batteries or energy harvesting as seen in **Fig. 1** (a). Those sensor nodes form a wireless sensor network and send events to the gateway, shown in Fig. 1 (b). The sensor events are triggered when they detect the presence of the human, but they are not triggered when the presence of resident disappears because of energy saving.

We attempt to find the location of binary sensors in multi-resident environment in a general house. The number of sensors which we assume is normally between 10 to 20 depending on the size of house, but it is not limited to this range. Basically, more sensors will provide finer-grained mobility information, and we will later discuss the impact of the number of sensors to the accuracy. The typical scenario is that firstly a resident obtains (buys or rents) a set of sensors. After the resident deploys those sensors on their own, we request that resident to give some information to our system discussed in Section 3.2. Then we leverage those information to find the location of sensors by analyzing the sequence of sensor events collected through the gateway to the cloud server, without requiring the labeled data discussed in Section 4.4.

3.2 Indoor Floorplan

The indoor map is important to provide the information about rooms, hallways, entrance etc. in which residents may place motion sensors. We note that, some research studies provide a solution for large buildings such as office buildings and commercial complex to generate floorplan images by using crowdsourcing techniques. Although the SLAM techniques [29], [30], [31] or

PDR techniques [32], [33] to trace human walking paths can be used to construct the floorplan images, the drawback is inaccuracy caused by accumulated distance and direction errors caused by orientation change of smartphones while they are walking, inaccuracy of stride length estimation and some other unexpected noise.

We assume the scenario to receive the floorplan image. When they buy or rent the set of sensors, they can see the instruction for the residents to send a floorplan. The residents can request a physical copy of floorplan of their houses from the housing manager or they can draw it by themselves. After that they are requested to send it to the service provider. There are two options to send the floorplan. Firstly, they can send the digital versions of given indoor map information which are generated by taking a photo of a floorplan image using a smartphone camera, and upload the photo to the service provider through e-mail service or mobile application service. Secondly, they can send the physical copy of their floorplan to the service company via mail service.

The given picture of floorplan should be drawn by using well-known symbols such as walls, doors and stairs (such simple illustrations can even be hand-drawn). We are able to analyze this picture manually or by some existing floorplan analysis technique. For example, the work by Heras et al. [34] provides a technique to generate the classification model to identify walls, doors and windows in images, and to recognize the rooms and space from those identified components. This indicates that the floorplan contains complete room layout with room types as well as room entrance (doors). In addition, if the scale information (size) is available, it helps for more accurate estimation, but the rough scale can be estimated by size of typical things like entrance door or some others. For example, we can estimate the scale information by calculating a ratio between the size of the front door in the image and the average size of front door and windows such as the front door in Japan (80–90 centimeters).

Given the floorplan and its analysis, we create the *floorplan graph*, which is a graph representing connections between rooms as well as room types and estimated Euclidean distance between the rooms. More specifically, after we analyze the floorplan to detect the areas from the floorplan, we generate a set $L_{floor} = \{l_1, l_2, ..., l_n\}$ of square partitions called *locations*, each of which corresponds to a room or a space like hallway. Then we generate a floorplan graph $G_{floor} = (L_{floor}, E_{floor}, f_t, f_d)$ where we create an edge between two locations if they are adjacent but separated by a wall with a door or they originally share the same space (i.e. without walls between them). $f_t : L_{floor} \rightarrow T$ is a room type assignment where T is a set of room types such as T ={ entrance, hallway, kitchen, dining_room, living_room, bedroom }. Finally, we estimate the Euclidean distance between the centers of l_i and l_j for each $(l_i, l_j) \in E_{floor}$, and build the distance function $f_d : E_{floor} \rightarrow R^+$. A typical floorplan image, the room dimension, and the path distance are drawn in **Fig. 2**.

In summary, we use the given sensor event sequences and floorplan graph as inputs. We will explain our localization algorithm in the following section.

4. Sensor Localization Method

4.1 Identifying Sensors at Particular Locations

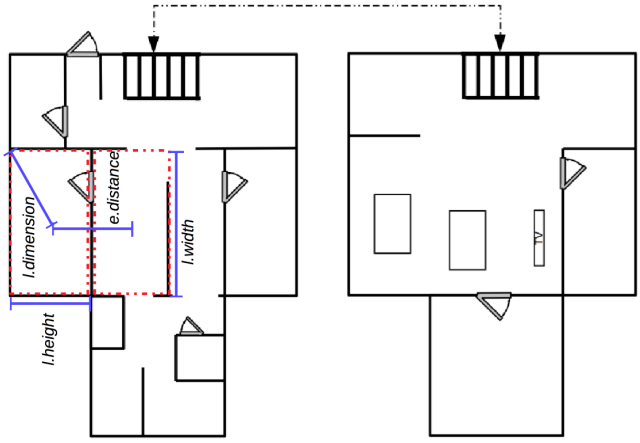
We call such locations where the residents perform activities with particular time patterns *key locations*. In this paper, we consider (i) bedroom, (ii) kitchen and (iii) entrance are such key locations. In our previous work that has been presented in a conference [35], we have proposed a technique to recognize common activities on the specific time in the specific location. Assuming the obtained common activity information, the location of the sensors detecting such common activities can be estimated with more confidence. For example, most residents may cook in the morning, (noon) and evening. So the sensor(s) that detects residents many times in those time zones can be estimated as that in the kitchen. The sensor at a bedroom can be identified by finding the sensor detecting events at night with long intervals (caused by sleeping activity). Similarly, the sensor at an entrance can be identified by finding the one detecting events in daytime with long interval (caused by go-out activity). For this purpose, we need to identify the typical daily pattern of such activities, and such a pattern can be obtained by the crowdsourced survey or some other techniques. In our previous work, we exploited the result of the crowdsourced survey which was done by our research group [2]. We note that we may also regard other room types as key locations if we can identify the typical daily activities that are associated with those rooms.

4.2 Finding Geographical Relation of Sensors

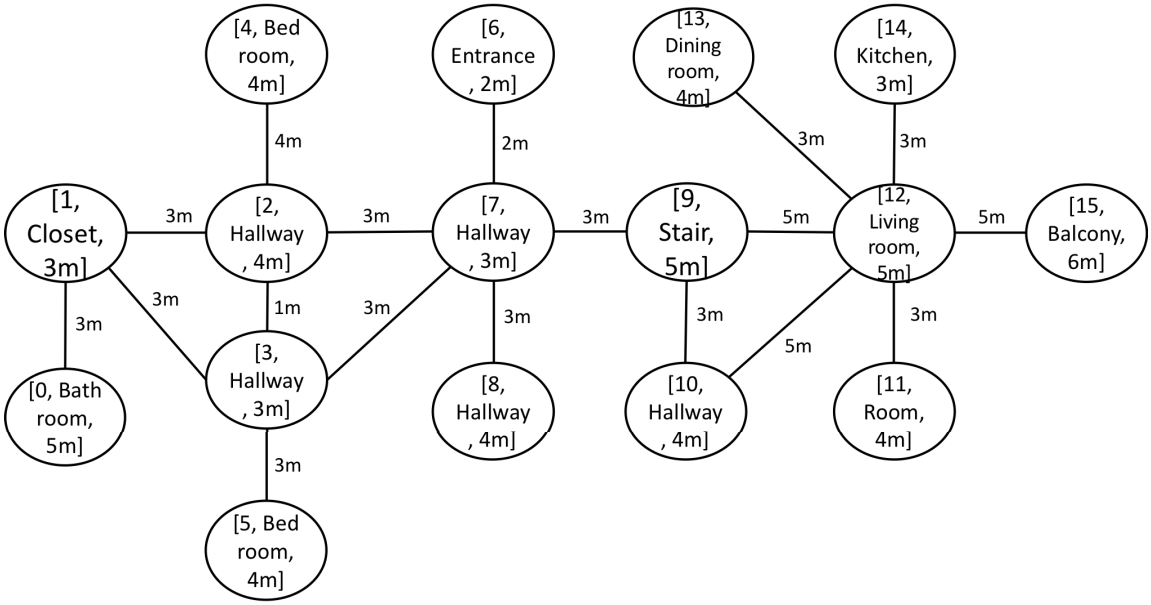
As the second step, we generate a sensor graph, where each edge represents the fact that there was a direct trip between the pair of sensors. We also estimate the physical distance between them based on the two events from different sensors. Specifically, if two events from sensors s_i and s_j occur time-subsequently, we regard that there was a direct trip by a resident from s_i to s_j. Then the physical distance between s_i and s_j can be estimated by the time interval of the two events and a presumed walking speed.

However, the sensor graph generated by event sequences may be inaccurate due to several reasons. The most significant factor that causes the errors is multi-residents environment, because multi-residents may generate a disordered sequence of events. If the time intervals between two events from two sensors, say s_i and s_j, differ from time to time, we can regard that the two events are caused by multi-residents and are not used for sensor graph generation. This is based on the observation that if two events are caused by walking of a resident, the time intervals are almost the same.

Even though we eliminate such situations, we may still incorrectly estimate the edge presence and physical distance. Especially, when two residents perform activities in different rooms and one of them does not move a lot (e.g., listening to music in the living room) and another does, we may not be able to see such deviations of time intervals due to lack of sufficient event samples, and instead, we may observe the frequent event occurrence of one sensor and much less event occurrence from another. In order to overcome this problem, we introduce the concept of *similarity* of event sequences of two sensors, based on the observation that two



(a) Floorplan.



(b) Floorplan graph whose nodes cantain attribute [room id, room type, dimension] and edges contain the distance between room.

Fig. 2 Floorplan and Floorplan Graph Example (Two-story House).

near sensors may generate similar event sequences.

Based on the observation, the algorithm for making a sensor graph consists of the following three steps; (i) we initiate the sensor graph using the correlation of event patterns, (ii) we update the sensor graph based on the correlation of event sequence association, and (iii) we calculate the topology of the sensor graph with the physical distance between each pair of the sensor. Each step will be explained below.

4.2.1 Similarity of Event Patterns

In the initial step, we exploit the similarity among event patterns between pairs of sensors. Our hypothesis is that, if two sensors are placed in close proximity, the event patterns of those two sensors will be similar. Thus the similarity in event patterns represents how close they are to each other as exemplified in **Fig. 3**.

This figure shows the patterns of sensor events seen in the real environment. Such patterns of sensor events from sensor ids 16 and 17 placed in the living room and 18 placed in the kitchen are similar where the kitchen is close to the living room. Meanwhile, the pattern from sensor id 8 placed far from the living room has

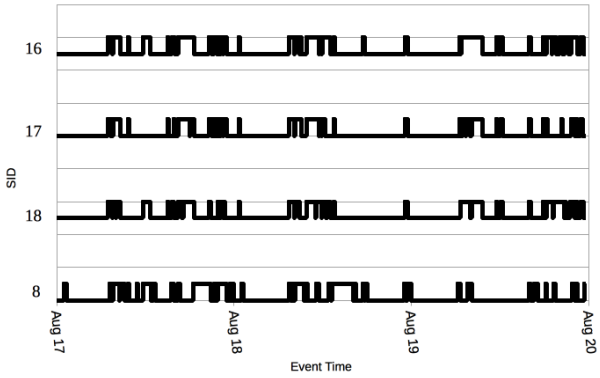


Fig. 3 Similarity of Two Sensor Event Patterns.

less similarity to them. In summary, the sensors installed in the same room or close location are likely to detect the same movement of residents.

To initiate the sensor topology by using the similarity of event between sensors, we generate a *sensor graph* G_{sensor} =

(S, E_{sensor}) where S is a set of sensors and $E_{sensor} = S \times S$. That is, G_{sensor} is a fully connected graph. Then we define the value associated with each edge e_{s_i,s_j} (denoted as $e_{s_i,s_j}.value$), which represents the similarity of event patterns of s_i and s_j and remove such an edge e_{s_i,s_j} that $e_{s_i,s_j}.value$ is below a threshold. To calculate the similarity of event patterns, we use the method by Twomey et al. [36]. The method leverages the mutual information in the event patterns to generate the topology of the sensors without a given threshold.

We regard that two sensors, which detect events in similar time patterns, are close to each other as being close to each other since they are likely to detect the same activity of a resident. We introduce δt as a time difference threshold to regard events from two sensor as caused by a single activity (δt is set to 1 minute in this paper). Based on this hypothesis, we analyze the relationship of event patterns from the multiple sensors. We introduce a timeslot of δt period, and the event sequences of sensors during T timeslots can be described as $RD^T = \{rd^1, rd^2, ..., rd^T\}$, which is a time series of m-dimensional binary vectors rd_t ($1 \leq t \leq T$). The i-th ($1 \leq i \leq m$) element of rd_t is denoted as rd_i^t and the value of rd_i^t is 1 if an event from sensor s_i is observed during t-th timeslot and 0 otherwise. Then we calculate the ratio of the number of timeslots where the pair of sensors s_i and s_j equals a given pair of binary values, such as (0, 1), over T. Similarly, we calculate the ratio of the number of 0's or 1's from a single sensor over T. We denote such a ratio for sensor pair (s_i, s_j) with a value pair (a, b) $(a, b \in 0, 1)$ by $R^T_{(i,j)}(a, b)$ and that for a sensor s_i with a value a as $R^T_i(a)$. For example, given RD^5 ={ (0,1), (0,0), (1,1), (0,1), (0,0)} for s_1 and s_2, $R^5_{(1,2)}(0, 1) = 2/5$ as (0,1) appears twice in 5 timeslots. Finally, we calculate *mutual information* $I(s_i; s_j)$ for each pair of sensors s_i and s_j using equation (1).

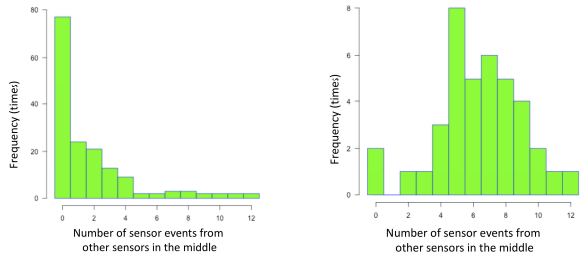
$$I(s_i; s_j) = \sum_{(a,b)\in\{0,1\}^2} R^T_{(i,j)}(a, b) \log \frac{R^T_{(i,j)}(a, b)}{R^T_i(a)R^T_j(b)} \quad (1)$$

After our system calculated the mutual information for each pair of sensors, our system calculates the sensor topology by using a threshold proposed in the paper from Twomey et al. [36]. They remove the edges between a pair of sensors unless the mutual information of that pair of sensors is over $\frac{E[I(s_i;*)]+E[I(*;s_j)]}{2}$, and we think this is reasonable to be used for initializing the sensor graph because it is a moderate threshold.

4.2.2 Event Sequence Association

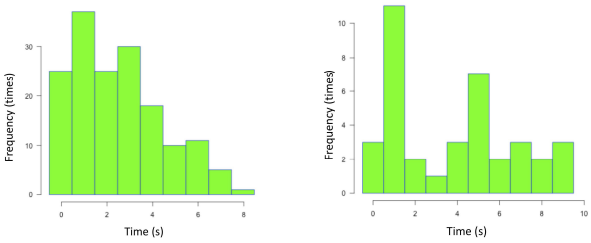
The edges in the sensor graph created by the algorithm in the previous section mean that there is similarity of event patterns between the two nodes of the edge. However, we would like to have correspondence between the physical direct paths and the edges of the sensor graph for final matching between the floor graph and the sensor graph.

In order to eliminate such links that do not correspond to walking paths, we make the hypothesis that the events from two sensors placed near to each other are usually along with each other when a resident walks past those sensors. As a preliminary experiment to estimate the direct walking path between two sensors s_i and s_j, we observed two cases of two sensor placements; in one case, s_i and s_j are located close to each other and in another case, they are far away from each other. Then we focused on the



(a) near locations (b) faraway locations

Fig. 4 Number of events from other sensors that occurred between two events of s_i and s_j.



(a) neighboring sensor pair (b) non-neighboring sensor pair

Fig. 5 Gap time between two events of the pair of sensors (2 days data collection).

number of events from the other sensors, which occurred between two events from s_i and s_j. The histgram of such number is shown in **Fig. 4**.

The results show if two sensors are near to each other, most events of s_i and s_j are time-adjacent to each other (i.e., no other events exist between two) as shown in Fig. 4 (a). Meanwhile, there are many cases that other events are between as shown in Fig. 4 (b). Here we say that two sensors s_i and s_j are *neighboring* if two events of them are time-subsequent (there is no event between them). For example, if we observe a sequence of four events in the order of s_1, s_3, s_1 and s_2, (s_1, s_2) and (s_1, s_3) are neighboring pairs, and (s_2, s_3) is not. We simply assume the direct walking path between s_i and s_j exists if s_i and s_j are neighboring. Thus, we update G_{sensor} by using this neighboring relationship.

However, there may be some links generated wrongly if multi-residents perform the activities at the same time in different rooms. Those activities may also create events that correspond to neighboring relationship for non-neighboring sensor pairs. To deal with this issue, we assume that the traveling time between the neighoring sensors, which is seen when a single resident directly walks passing those two sensors, is relatively constant, while the time intervals caused by multi-residents are dispersed. We investigation the subsequence travel time distribution from a neighboring sensor pair and a non-neighboring sensor pair. Fortunately, the distribution of time gap in the multi-resident environment vary and have no pattern seen in **Fig. 5**.

Figure 5 (a) shows the trip time of events from a pair of sensors will usually be short if they are placed near to each other. For example, two sensors placed on the corridor between the bedroom and the toilet detect the resident who walks passing within the short time. Meanwhile, Fig. 5 (b) shows that the traveling time has no pattern when the two sensors are placed far from

each other, because the sensor events from those sensors might be caused by different residents who did activities in different rooms. Hence, we introduce the temporal standard deviation of traveling time $temp\_travel_{std}(s_u, s_v)$ and temporal minimal traveling time $temp\_travel_{min}(s_u, s_v)$ for each pair of sensors s_u and s_v in T_2 minutes. In this paper, we use $temp\_travel_{std}(s_u, s_v)$ to detect time period during which a single resident is present. If there is no such $temp\_travel_{std}(s_u, s_v)$ that is higher than a threshold β_1, that period time is the confidence time seen in Algorithm 1.

We also collect the distribution of traveling time $travel_{dist}(s_u, s_v)$ for each pair of sensor s_u and s_v. This distribution is used to calculate the physical distance of each edge described in the below part. The event sequence seen when a resident performs activity alone is used to update the topology of sensor graph seen in Algorithm 2.

After G_{sensor} is updated, we remove those edges that are marked with removal by Algorithm 3. The distance and topology calculation will be explained in the following section.

4.3 Physical Distance Estimation

After the sensor graph is updated, we estimate the physical distance of each edge of the sensor graph by multiplying the mean traveling time of the pair of sensors and the average walking speed (set to 1 m/s in our experiment).

Here the walking speeds of residents may vary depending on situations. For example, it may be fast when the resident hurries on, and slow when she/he stays at a place during the walking. To avoid the unreliable case due to such a stop-and-go case, We cluster the walking time by k-means with $k = 3$ into 3 classes: fast, normal and slow speeds. Then we calculate the mean of traveling time $travel_{dist}(s_u, s_v)$, and update the sensor graph by ignoring the slow speed class.

4.4 Indentifying Sensor Locations

In this section, we create the mapping function $\mathcal{A} : S \rightarrow L_{floor}$ where L_{floor} is a set of room locations from the floor plan graph, and S is a set of sensors in the sensor graph. We try to map the sensors into the floor plan graph by leveraging the room dimension and the walking distance between the center of two rooms. For this purpose, we introduce a sensor-room association score $assoc_{s,l}$ which represents the number of times where sensor s is expected to be placed in that location l. The process to increase the $assoc_{s,l}$ and select the sensor location is shown in the following three steps.

For the algorithm, we use the notation of the mapping function $\mathcal{A} : S_{mapped} \rightarrow L_{mapped}$ where S_{mapped} is a set of sensors which have already been mapped onto a set L_{mapped} of locations ($S_{mapped} \subseteq S$ and $L_{mapped} \subseteq L$. We start with $S_{mapped} = S_{key}$ where S_{key} is the set of sensors estimated to be in the key locations. Then we pick up one sensor $r \in S_{key}$ for each location $l \in L_{mapped}$ and call it *reference sensor of location l*. We assume r_l is located at the center of location l and classify the rest of sensors in the sensor graph which have a relation with r_l into 3 groups; *same location*, *neighbor location* and *unknown*. More specifically, we classify the relationship of sensors with the reference sensor by using the walking distance attribute in the floor plan graph and the distance between sensors in the sensor graph. We regard the distance between sensors s_i and s_j on the sensor graph as the summation of the distances in its shortest path between the two sensors and denote it as $d(s_i, s_j)$. Sensor s_i is supposed to be placed in the same location with reference r_l of location l if $d(s_i, r_l)$ is shorter than the dimension of location (room) l, in location $l_{neighbor}$ close to l if $d(s_i, r_l)$ is longer than the dimension of $l_{neighbor}$ but shorter than the summation of the dimension of $l_{neighbor}$ and the walking distance from l to $l_{neighbor}$, as shown in Algorithm 4.

Algorithm 1 *isConfidence_Time*(O)

Require: $O = (o_0, o_1, ..., o_t)$ is an observation of events in the past T_2 minutes.
Ensure: *isSingle* which represents the confidence time when it is TRUE.
Ensure: *time_diff* is a set of minimum traveling time between sensors
$t_{start} \leftarrow o_0.time$
for $\forall i \in (1, 2, ..., t)$ **do**
  if $o_i.sid \neq o_{i-1}.sid$ **then**
    if $o_i.sid > o_{i-1}.sid$ **then**
      $s_u \leftarrow o_{i-1}.sid$ and $s_v \leftarrow o_i.sid$
    else
      $s_v \leftarrow o_{i-1}.sid$ and $s_u \leftarrow o_i.sid$
    end if
    $tdif_{s_u,s_v}[i] \leftarrow o_i.time - o_{i-1}.time$
  end if
end for
update $temp\_travel_{std}(s_u, s_v)$ by $tdif_{s_u,s_v}$
update the distribution $travel_{dist}(s_u, s_v)$ by $tdif_{s_u,s_v}$
for $\forall e_a \in E_a = (s_u, s_v) \wedge u \neq v$ **do**
  if $temp\_travel_{std}(s_u, s_v) > \beta_1$ **then**
    $isSingle = FALSE$
  end if
end for
return $isSingle$, $temp\_travel_{min}$

Algorithm 2 *update_sensor_graph*(O)

Require: $O = (o_0, o_1, ..., o_t)$ is an observation in the past T_2 minutes.
Require: $G_{sensor} = (S, E_{sensor})$ is the previous sensor graph.
Ensure: $G_{sensor} = (S, E_{sensor})$ is the updated sensor graph.
isSingle, *time_dif* = *isConfidence_Time*(O)
if *isSingle* =TRUE **then**
  for $\forall s_u, \forall s_v \in S$ $(s_u \neq s_v)$ **do**
    if $\exists e = (s_u, s_v) \in E_{sensor}$ **then**
      $e.remove = 0$
    else
      $e.remove = 1$
    end if
  end for
end if

Algorithm 3 *finalize_sensor_graph*(O)

Require: G_{sensor} is the updated sensor graph.
Ensure: G_{sensor}.
for $e \in E_{sensor}$ **do**
  if e.remove == 1 **then**
    remove e
  end if
  calculate *e.distance* and *e.sameroom*
end for

Algorithm 4 *feasible_location*

Require: S is a set of sensors
Require: $S_{mapped} \subset S$ is a set of sensors having been already mapped.
Require: $\mathcal{A}$ is a current mapping function.
Require: G_{sensor} and G_{floor} are the sensor graph and the floor plan graph, respectively.
for $\forall s_{mapped} \in S_{mapped}, \forall s \in S\ (s_{mapped} \neq s \wedge e_{sensor} = (s, s_{mapped}) \in E_{sensor})$ **do**
  $l_{mapped} = \mathcal{A}(s_{mapped})$
  calculate $d(s, r_{l_{mapped}})$
  if $d(s, r_{l_{mapped}}) \leq l_{mapped}.dimension$ **then**
    $assoc_{s,l_{mapped}} = assoc_{s,l_{mapped}} + 1$
  else
    for $\forall l' \in L\ (l' \neq l_{mapped} \wedge e' = (l', l_{mapped}) \in E_{floor})$ **do**
      if $d(s, r_{l_{mapped}}) \leq e'.distance + l_{mapped}.dimension$ **then**
        $assoc_{s,l'} = assoc_{s,l'} + 1$
      end if
    end for
  end if
end for

Algorithm 5 *matching*$(G_{floor}, G_{sensor}, \mathcal{A}_{key})$

Require: $G_{floor} = (L_{floor}, E_{floor})$ is a floor plan graph.
Require: $G_{sensor} = (S, E_{sensor})$ is a sensor graph.
Ensure: $\mathcal{A} : S \rightarrow L_{floor}$
satisfy = FALSE
Initial $\mathcal{A} : S_{mapped} = S_{key} \rightarrow L_{key}$
while $\neg$*satify* **do**
  call $feasible\_location(S, S_{mapped}, \mathcal{A}, G_{sensor}, G_{floor})$
  satify =TRUE
  for $\forall s \in S \wedge s \notin S_{mapped}$ **do**
    $L_{list} = \arg\max_l assoc_{s,l}$
    if $|L_{list}| == 1$ **then**
      $\mathcal{A}_{mapped}(s) = head(L_{list})$
      if $root_{head(L_{list})}$ is not set **then**
        $root_{head(L_{list})} = s$
      end if
      satify =FALSE
    end if
  end for
end while
return $\mathcal{A}$

We check the valid constraint of the sensor-room association score in the second step. The idea behind this step is the first step may increase an invalid score of the sensor-room association because we rely on only one mapped sensor. Therefore we will check rooms l' which are mapped to each sensor s_{mapped} and we will set $assoc_{s,l'}$ to zero if there is no edge $e = (s, s_{mapped}) \in E_{sensor}$.

In the third step, we estimate location l for each sensor s by using $\arg\max_l assoc_{s,l}$. This means the location with the highest score is regarded as the location of sensor s. Then we will update the mapping function $\mathcal{A}$ by appending "$\mathcal{A}(s) = l$". After that, our algorithm repeats the step one until the algorithm is unable to find the maximum score of $score_{s,l}$ shown in Algorithm 5.

5. Experiment

5.1 Datasets and Evaluation

We conducted experiments by utilizing four datasets. The first three datasets, called DT1, DT2 and DT3, consist of the sensor events which we collected from two real houses. In particular, we installed sensors in a two-story house (house *A*) with three family members and a single-story house (house *B*) where a single elderly person lives. DT1 was collected from 19 motion sensors in house *A* in mid of 2016 (shown in **Fig. 6** (a)). DT2 was collected from 18 motion sensors in the same house *A* in early 2015. DT3 was collected from 17 motion sensors in the house *B* in February 2017 (shown in Fig. 6 (c) and **Fig. 7**).

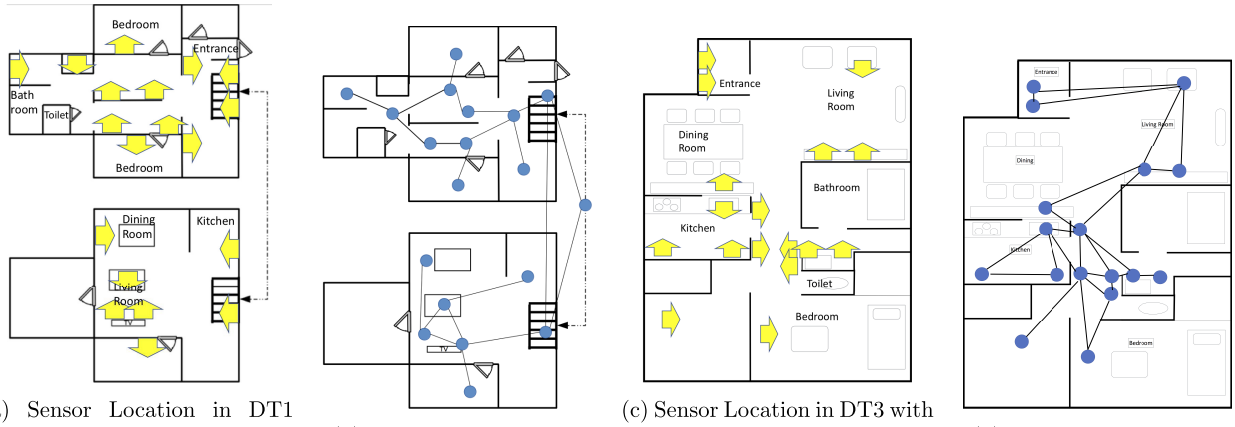
The fourth dataset (called DT4) is the public dataset (CASAS dataset). In this dataset, the sensor events were collected in the Japanese house where two residents lived [37]. Even though we cannot see the scale information of the floorplan in this work clearly, we apply the average size of the front door in the Japanese house to estimate the scale information. In this dataset, we regard only the motion sensor events caused by the presence of the residents as "ON" events. In this work, many binary sensors were deployed in the house, but it is not realistic to assume that a normal resident buys many sensors for their typical 2-story home. Therefore, we picked up 21 sensors for this dataset. We randomly picked a maximum of two sensors from each room, and we have many sets of sensors where we pick up. For example, there are 6 sensors in a room and we pick up to 2 sensors, then we have 6 combinations. After that, we apply our method to every combination and take the average of the result from every combination.

We consider the only the cases that we know or can estimate the floor plan in more detail such as room size. According to Section 3.2, we, therefore, mention the floor plan at least the third-level floor plan information. The fourth-level and fifth-level floor plan information have enough information to be classified into the same categories. The performance of matching result is evaluated by an accuracy. We assume that the floor plans of these houses have already been identified.

5.2 Matching Performance

We conducted the performance testing on 4 datasets by assuming the sensors in key locations were already identified. Since we need to see the effect of the number of identified sensors in specific locations on the matching performance, we assumed that we could estimate sensors in 5 key locations ("bedroom", "living room", "kitchen", "front entrance" and "wash basin") and evaluated our matching method in the following 3 cases. The first case named "3keys" is that the resident places sensors in the bedroom, kitchen and front entrance. In the second case named "4keys", one more sensor is placed to the living room in the "3keys" case. In the "5keys" case, one more sensor is placed in the wash basin in "4keys" case. The results of our room-level sensor localization are shown in **Table 1**.

According to Table 1, we found our method can obtain good accuracy in most cases after 3 days observation. Since there are fluctuations in the performance in some datasets, we assessed the event sequences and found that the resident might not do the same activities every day. As a result, those activities of resident only from 3 days were insufficient to affect the sensor graph topology. Additionally, there were some locations where the resident rarely visited. Consequently, sensors in those locations could not



(a) Sensor Location in DT1 with sensing direction
(b) Sensor Graph in DT1
(c) Sensor Location in DT3 with sensing direction
(d) Sensor Graph in DT3

Fig. 6 Floor plan with Sensor Locations.

Fig. 7 Snapshot from experiment site.

Table 1 Matching Accuracy with sufficient floor plan information.

Dataset		Running Days				
		1	2	3	4	5
DT1	3keys	0.58	0.74	0.79	0.79	0.79
	4keys	0.79	0.79	0.84	0.84	0.84
	5keys	0.84	0.89	0.95	0.95	0.95
DT2	3keys	0.56	0.72	0.78	0.78	0.78
	4keys	0.56	0.78	0.83	0.83	0.83
	5keys	0.67	0.89	0.89	0.89	0.89
DT3	3keys	0.58	0.50	0.83	0.83	0.75
	4keys	0.67	0.83	0.83	0.83	0.83
	5keys	0.92	0.92	0.92	1.00	0.92
DT4	3keys	0.43	0.81	0.71	0.71	0.76
	4keys	0.48	0.81	0.86	0.90	0.95
	5keys	0.52	0.90	0.95	0.95	0.90

be in the sensor graph due to insufficient event sequences from those sensors. For example, the result in datasets "DT2" and "DT4" with 1-day case, in which the resident did not pass all the sensors on the first day, the accuracy was rather low. But as we mentioned, the accuracy is totally good as it reaches 0.70 in most cases and is close to 0.80 even in 3keys case, after 5-days observation.

We note that the major issue that affects the performance is symmetric structure. Our algorithm is unable to match a sensor into a location from those with similar conditions. In such a case, we should increase the number of key locations to reduce the similarity in the searching space. Consequently, the results have been better when we have enough sensors in key locations as in Table 1.

Furthermore, there is another benefit of having key locations. Such benefit is that we can match some sensor into a location close to key locations. Specifically, our method can estimate the location of one or two sensors which are placed close to the key locations correctly after sensor events are collected for 1 day.

6. Discussion

In this section, we found there are some factors that affect accuracy of our method and we would like to discuss on that factors. The factors rendering the greatest effect are walking speed of resident and limitation of binary event for sensor in key locations.

6.1 Walking Speed of Resident

In this work, we assume the average walking speed is about 1 m/s, which is normal for most of people, because there may be family members in the same house (e.g., kids, father, mother as well as elderly). In such a house, some may move fast and some others may move slow. Therefore, we consider using 1 m/s as the average is reasonable. On the other hand, there are such houses where only elderly people live. In such a house, if we adopt 1 m/s as the average walking speed, the accuracy will be worse. Therefore, we come up with the idea of estimating the walking speed by observing the event sequence of resident who walks from one key location to another key location directly (without passing the other sensors). If we can recognize the event sequence with known distance between those locations, we will be able to estimate the average walking speed automatically.

6.2 Limitation of Binary Event for Sensor in Key Locations

In our method, the key locations are such locations where a resident stay for some time to perform a well-known activity. Since such activity has an individual characteristic of event pattern, we

can easily identify the type of the activity, and correspondingly, those sensors which perceive the event pattern are considered to be located in the "key location". Our goal is automatic identification of all the sensors in the key locations without any help from the resident side. Nevertheless, some sensors in key locations are difficult to identify only from the binary event patterns. For instance, we see a few binary events of the resident who sits on a sofa in a living room. However, our method actually can identify three sensors in key locations. We, also, show if at least three sensors in key locations can be identified, our method is able to capture a good result. Our method, consequently, does not require any input is necessary from the resident side.

7. Conclusion

This paper propose a method to find the location of motion sensors that are used to detect the daily life activities in the home environment. Our technique requires a prior knowledge such as floor plan and can estimate the mapping function between sensors and locations with about 80% or above accuracy after 3 days data collection without any labeled data.

In the future, we will improve our technique for the more precise localization of binary motion sensors. Specifically, the sensor graph which is generated from our method contains some feature such as the distance between two sensors. Not only the room-level localization but also the order of sensors can be estimated. Besides localization of precision sensors, we will consider the smart-home system in which the many types of sensors are involved.

Acknowledgments The work was supported by "Research and development of Innovative Network Technologies to Create the Future (No. 191)", the Commissioned Research of National Institute of Information and Communications Technology (NICT), Japan.

References

[1] United Nations: World Population Ageing 2015, *Department of Economic and Social Affairs, Population Division*, No.ST/ESA/SER.A/390 (2015).

[2] Nakamura, S., Shigaki, S., Hiromori, A., Yamaguchi, H. and Higashino, T.: A Model-based Approach to Support Smart and Social Home Living, *Proc. 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, *UbiComp '15*, pp.1101–1105, ACM (online), DOI: 10.1145/2750858.2805835 (2015).

[3] Emi, I.A. and Stankovic, J.A.: SARRIMA: Smart ADL Recognizer and Resident Identifier in Multi-resident Accommodations, *Proc. Conference on Wireless Health*, *WH '15*, pp.4:1–4:8, ACM (online), DOI: 10.1145/2811780.2811916 (2015).

[4] Hoque, E. and Stankovic, J.: AALO: Activity recognition in smart homes using Active Learning in the presence of Overlapped activities, *2012 6th International Conference on Pervasive Computing Technologies for Healthcare* (*PervasiveHealth*) *and Workshops*, pp.139–146 (online), DOI: 10.4108/icst.pervasivehealth.2012.248600 (2012).

[5] Krishnan, N.C. and Cook, D.J.: Activity recognition on streaming sensor data, *Pervasive and Mobile Computing*, Vol.10, pp.138–154 (online), DOI: 10.1016/j.pmcj.2012.07.003 (2014).

[6] Wang, P., Li, W., Gao, Z., Zhang, J., Tang, C. and Ogunbona, P.O.: Action Recognition From Depth Maps Using Deep Convolutional Neural Networks, *IEEE Trans. Human-Machine Systems*, Vol.46, No.4, pp.498–509 (online), DOI: 10.1109/THMS.2015.2504550 (2016).

[7] Oreifej, O. and Liu, Z.: HON4D: Histogram of Oriented 4D Normals for Activity Recognition from Depth Sequences, *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pp.716–723 (online), DOI: 10.1109/CVPR.2013.98 (2013).

[8] Song, B., Kamal, A.T., Soto, C., Ding, C., Farrell, J.A. and Roy-Chowdhury, A.K.: Tracking and Activity Recognition Through Consensus in Distributed Camera Networks, *IEEE Trans. Image Processing*, Vol.19, No.10, pp.2564–2579 (online), DOI: 10.1109/TIP.2010.2052823 (2010).

[9] Ermes, M., Prkk, J., Mntyjrvi, J. and Korhonen, I.: Detection of Daily Activities and Sports With Wearable Sensors in Controlled and Uncontrolled Conditions, *IEEE Trans. Information Technology in Biomedicine*, Vol.12, No.1, pp.20–26 (online), DOI: 10.1109/TITB.2007.899496 (2008).

[10] Atallah, L., Lo, B., King, R. and Yang, G.Z.: Sensor Positioning for Activity Recognition Using Wearable Accelerometers, *IEEE Trans. Biomedical Circuits and Systems*, Vol.5, No.4, pp.320–329 (online), DOI: 10.1109/TBCAS.2011.2160540 (2011).

[11] Helaoui, R., Niepert, M. and Stuckenschmidt, H.: Recognizing interleaved and concurrent activities using qualitative and quantitative temporal relationships, *Pervasive and Mobile Computing*, Vol.7, No.6, pp.660–670 (online), DOI: 10.1016/j.pmcj.2011.08.004 (2011).

[12] Wilson, D.H. and Atkeson, C.: Simultaneous Tracking and Activity Recognition (STAR) Using Many Anonymous, Binary Sensors, *Proc. 3rd International Conference on Pervasive Computing*, *PERVASIVE'05*, pp.62–79, Springer-Verlag (online), DOI: 10.1007/11428572_5 (2005).

[13] Morelli, C., Nicoli, M., Rampa, V., Spagnolini, U. and Alippi, C.: Particle Filters for Rss-Based Localization in Wireless Sensor Networks: An Experimental Study, *2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*, Vol.4, pp.IV–IV (online), DOI: 10.1109/ICASSP.2006.1661129 (2006).

[14] Ji, Y., Biaz, S., Pandey, S. and Agrawal, P.: ARIADNE: A Dynamic Indoor Signal Map Construction and Localization System, *Proc. 4th International Conference on Mobile Systems, Applications and Services*, *MobiSys '06*, pp.151–164, ACM (online), DOI: 10.1145/1134680.1134697 (2006).

[15] Chintalapudi, K., Padmanabha Iyer, A. and Padmanabhan, V.N.: Indoor Localization Without the Pain, *Proc. 16th Annual International Conference on Mobile Computing and Networking*, *MobiCom '10*, pp.173–184, ACM (online), DOI: 10.1145/1859995.1860016 (2010).

[16] Makki, A., Siddig, A., Saad, M., Cavallaro, J.R. and Bleakley, C.J.: Indoor Localization Using 802.11 Time Differences of Arrival, *IEEE Trans. Instrumentation and Measurement*, Vol.65, No.3, pp.614–623 (online), DOI: 10.1109/TIM.2015.2506239 (2016).

[17] h. Jung, S., c. Moon, B. and Han, D.: Unsupervised Learning for Crowdsourced Indoor Localization in Wireless Networks, *IEEE Trans. Mobile Computing*, Vol.15, No.11, pp.2892–2906 (online), DOI: 10.1109/TMC.2015.2506585 (2016).

[18] Kim, S.-C., Jeong, Y.-S. and Park, S.-O.: RFID-based Indoor Location Tracking to Ensure the Safety of the Elderly in Smart Home Environments, *Personal Ubiquitous Comput.*, Vol.17, No.8, pp.1699–1707 (online), DOI: 10.1007/s00779-012-0604-4 (2013).

[19] Buettner, M., Prasad, R., Philipose, M. and Wetherall, D.: Recognizing Daily Activities with RFID-based Sensors, *Proc. 11th International Conference on Ubiquitous Computing*, *UbiComp '09*, pp.51–60, ACM (online), DOI: 10.1145/1620545.1620553 (2009).

[20] Youssef, M., Mah, M. and Agrawala, A.: Challenges: Device-free Passive Localization for Wireless Environments, *Proc. 13th Annual ACM International Conference on Mobile Computing and Networking*, *MobiCom '07*, pp.222–229, ACM (online), DOI: 10.1145/1287853.1287880 (2007).

[21] Luo, Y., Huang, K., Guo, X. and Wang, G.: A hierarchical RSS model for RF-based device-free localization, *Pervasive and Mobile Computing*, Vol.31, pp.124–136 (online), DOI: 10.1016/j.pmcj.2016.03.002 (2016).

[22] Wilson, J. and Patwari, N.: Radio Tomographic Imaging with Wireless Networks, *IEEE Trans. Mobile Computing*, Vol.9, No.5, pp.621–632 (online), DOI: 10.1109/TMC.2009.174 (2010).

[23] Eleryan, A., Elsabagh, M. and Youssef, M.: Synthetic Generation of Radio Maps for Device-Free Passive Localization, *2011 IEEE Global Telecommunications Conference - GLOBECOM 2011*, pp.1–5 (online), DOI: 10.1109/GLOCOM.2011.6134052 (2011).

[24] Wang, W., Liu, A.X., Shahzad, M., Ling, K. and Lu, S.: Device-Free Human Activity Recognition Using Commercial WiFi Devices, *IEEE Journal on Selected Areas in Communications*, Vol.35, No.5, pp.1118–1131 (online), DOI: 10.1109/JSAC.2017.2679658 (2017).

[25] Athalye, A., Savic, V., Bolic, M. and Djuric, P.M.: Novel Semi-Passive RFID System for Indoor Localization, *IEEE Sensors Journal*, Vol.13, No.2, pp.528–537 (online), DOI: 10.1109/JSEN.2012.2220344 (2013).

[26] Kawsar, F., Nakajima, T. and Fujinami, K.: Deploy Spontaneously: Supporting End-users in Building and Enhancing a Smart Home, *Proc. 10th International Conference on Ubiquitous Computing*, *UbiComp '08*, pp.282–291, ACM (online), DOI: 10.1145/1409635.1409673 (2008).

[27] Al-Kuwari, A.M.A.H., Ortega-Sanchez, C., Sharif, A. and Potdar,

V.: User friendly smart home infrastructure: BeeHouse, *5th IEEE International Conference on Digital Ecosystems and Technologies* (*IEEE DEST 2011*), pp.257–262 (online), DOI: 10.1109/DEST.2011.5936635 (2011).
[28] Tegou, T., Kalamaras, I., Votis, K. and Tzovaras, D.: A low-cost room-level indoor localization system with easy setup for medical applications, *2018 11th IFIP Wireless and Mobile Networking Conference* (*WMNC*), pp.1–7 (online), DOI: 10.23919/WMNC.2018.8480912 (2018).
[29] Shin, H., Chon, Y. and Cha, H.: Unsupervised Construction of an Indoor Floor Plan Using a Smartphone, *IEEE Trans. Systems, Man, and Cybernetics, Part C* (*Applications and Reviews*), Vol.42, No.6, pp.889–898 (online), DOI: 10.1109/TSMCC.2011.2169403 (2012).
[30] Hardegger, M., Roggen, D., Mazilu, S. and Trster, G.: ActionSLAM: Using location-related actions as landmarks in pedestrian SLAM, *2012 International Conference on Indoor Positioning and Indoor Navigation* (*IPIN*), pp.1–10 (online), DOI: 10.1109/IPIN.2012.6418932 (2012).
[31] Grzonka, S., Karwath, A., Dijoux, F. and Burgard, W.: Activity-Based Estimation of Human Trajectories, *IEEE Trans. Robotics*, Vol.28, No.1, pp.234–245 (online), DOI: 10.1109/TRO.2011.2165372 (2012).
[32] Philipp, D., Baier, P., Dibak, C., Drr, F., Rothermel, K., Becker, S., Peter, M. and Fritsch, D.: MapGENIE: Grammar-enhanced indoor map construction from crowd-sourced data, *2014 IEEE International Conference on Pervasive Computing and Communications* (*PerCom*), pp.139–147 (online), DOI: 10.1109/PerCom.2014.6813954 (2014).
[33] Alzantot, M. and Youssef, M.: CrowdInside: Automatic Construction of Indoor Floorplans, *Proc. 20th International Conference on Advances in Geographic Information Systems*, *SIGSPATIAL '12*, pp.99–108, ACM (online), DOI: 10.1145/2424321.2424335 (2012).
[34] de las Heras, L.-P., Ahmed, S., Liwicki, M., Valveny, E. and Sánchez, G.: Statistical segmentation and structural recognition for floor plan interpretation, *International Journal on Document Analysis and Recognition* (*IJDAR*), Vol.17, No.3, pp.221–237 (online), DOI: 10.1007/s10032-013-0215-2 (2014).
[35] Kitbutrawat, N., Yamaguchi, H. and Higashino, T.: Localization of binary motion sensors in house, *2017 13th International Wireless Communications and Mobile Computing Conference* (*IWCMC*), pp.1132–1137 (online), DOI: 10.1109/IWCMC.2017.7986444 (2017).
[36] Twomey, N., Diethe, T., Craddock, I. and Flach, P.: Unsupervised learning of sensor topologies for improving activity recognition in smart environments, *Neurocomputing*, Vol.234, pp.93–106 (online), DOI: 10.1016/j.neucom.2016.12.049 (2017).
[37] Cook, D.J., Crandall, A.S., Thomas, B.L. and Krishnan, N.C.: CASAS: A Smart Home in a Box, *Computer*, Vol.46, No.7, pp.62–69 (online), DOI: 10.1109/MC.2012.328 (2013).

Nathavuth Kitbutrawat received his B.Eng. degree in electrical engineering from Kasetsart University, Thailand, in 2001, and the M.E. degree in computer engineering from Kasetsart University, Thailand, in 2008. Currently, he is now a Ph.D. student under the supervision of Prof. Teruo Higashino in Mobile Computing Laboratory at Graduate School of Information Science and Technology, Osaka University.

Hirozumi Yamaguchi received his B.E., M.E., and Ph.D. degrees in information and computer sciences from Osaka University, Japan, in 1994, 1996, and 1998, respectively. He is currently an associate professor at Osaka University. His current research interests include design, development, modeling, and simulation of mobile and wireless networks and applications. He is a member of the IEEE.

Teruo Higashino received his B.S., M.S., and Ph.D. degrees in information and computer sciences from Osaka University, Japan, in 1979, 1981 and 1984, respectively. He joined the faculty of Osaka University in 1984. Since 2002, he has been a professor in Graduate School of Information Science and Technology at Osaka University. His current research interests include design and analysis of distributed systems, communication protocol, and mobile computing. Dr. Higashino is a senior member of IEEE and a fellow of IPSJ.