Pervasive and Mobile Computing 19 (2015) 24–36

Contents lists available at ScienceDirect

Pervasive and Mobile Computing

journal homepage: www.elsevier.com/locate/pmc

PlaceWalker: An energy-efficient place logging method that considers kinematics of normal human walking

Dae-Ki Cho [a], Uichin Lee [b], Youngtae Noh [c,*], Taiwoo Park [d], Junehwa Song [d]

[a] Computer Science, University of California at Los Angeles (UCLA), United States
[b] Knowledge Service Engineering, Korea Advanced Institute of Science and Technology (KAIST), South Korea
[c] Cisco Systems Inc., United States
[d] Computer Science, Korea Advanced Institute of Science and Technology (KAIST), South Korea

ARTICLE INFO

Article history:
Received 4 September 2013
Received in revised form 28 December 2013
Accepted 11 April 2014
Available online 18 April 2014

Keywords:
Energy-efficient place logging
Semantic location context
Location-aware services

ABSTRACT

Fine-grained place logging with Wi-Fi beacon signatures provides a useful tool for delivering various semantic location-aware services such as reminders and advertisements. Existing solutions however heavily rely on *energy-hungry periodic Wi-Fi scanning* for place detection in resource limited mobile devices. In this paper, we present PlaceWalker, a scheme that uses a low-power duty-cycled accelerometer in the background to continuously monitor *user's significant physical activity changes* (e.g., walking to resting) as it provides a useful clue to the change of place. Unlike existing schemes, PlaceWalker triggers Wi-Fi scanning only when such an activity shift is detected and then determines a change of place by comparing Wi-Fi signatures. Our experimental results verify that detecting significant activity intensity changes can precisely capture arrival/departure times, and PlaceWalker substantially lowers the energy consumption by as much as 60.9%, when compared with the state-of-the-art method. We also analyze the experimental results with a simple analytic model and validate its efficiency under varying parameter settings.

1. Introduction

Automatically logging semantically meaningful places (e.g., home, office, and pizzeria) provides a useful tool for delivering various *semantic* location-aware services [1–4]. For instance, Alice can set the location-aware alarm to remind her of a checklist when she leaves her home, and she can also opt in *selectively* receiving advertisements from her favorite stores while visiting a shopping center. The logged data can be further processed to deliver personalized services like analyzing individual activity patterns, inferring mode of transport, recommending places/activities, etc. Part of these services require fine-grained (sub-building level) place logging with fairly accurate entrance and departure time monitoring (e.g., detecting when a user leaves the office). Other services require coarse-grained (building/region level) place logging with less precise time estimates (e.g., telling roughly when a user left campus). Given that user demands on place logging granularity vary significantly depending on the application, the support of both coarse- and fine-grained logging will enable a wide spectrum of semantic location-aware services even with resource limited mobile devices.

One way of logging places is to collect an individual's location trace (using localization like GPS, Skyhook, and Place Lab) and to perform *spatial clustering* of location samples (also known as geometry-based methods) [1,5,2,6–8]. Alternatively,

* Corresponding author. Tel.: +1 323 303 7945.
E-mail address: ynoh@cisco.com (Y. Noh).

http://dx.doi.org/10.1016/j.pmcj.2014.04.001
1574-1192/

we can use signature-based methods where ubiquitously available wireless beacons such as Wi-Fi hotspots and cell towers are used as a clue to place detection [3,4,9]. The granularity of logged places varies widely depending on logging algorithms (geometry vs. signature), density of location samples, and most importantly underlying localization methods. In general, geometry-based methods provide coarse-grained (building/region level) logging with rough entrance/departure times, whereas Wi-Fi signature based methods provide fine-grained (sub-building or room level) logging with fairly accurate entrance/departure times.

The main focus of this paper is fine-grained logging with Wi-Fi beacon signatures which is still considered very challenging as it requires energy-hungry continuous Wi-Fi scanning [3,4]. It is well known that a fully charged smartphone can last only a few hours with continuous Wi-Fi scanning. Recently Kim et al. [9] proposed an efficient place logging system called SensLoc that suspends periodic Wi-Fi scanning when a user arrives at a place and becomes stationary. User movements are then monitored with a low-power accelerometer to decide when to resume periodic Wi-Fi scanning. Given that people tend to stay indoors most of their time (e.g., on average 87% [10]), SensLoc can significantly reduce the average usage of Wi-Fi scanning. However, this efficiency (i.e., suppressing Wi-Fi scanning) is highly limited to the case that a user is in a place without substantial movements. Whenever a user performs non-stationary activities for a long duration (say walking and shopping in town during the weekend), the periodic Wi-Fi scanning of SensLoc will quickly drain the battery. In practice, prolonged mobility scenarios are not uncommon in our everyday lives as reported by Klepeis et al. [10], and thus, effectively handling such cases is very important in a ubiquitous application like place logging.

Given this observation, it is highly recommended to suppress unnecessary periodic background Wi-Fi scanning used for place detection. One option is to *always* use a low-power accelerometer as in an accelerometer-assisted localization scheme like AAMPL [11] where place-dependent activity intensity patterns are used for place detection based on the fact that a person performs a fairly unique set of activities in a given place. In practice, however, this method is laborious as users need to collect the acceleration data in each place during the training phase. Moreover, it requires a large number of acceleration samples observed over a certain period of time to recognize a place, and thus, it is very challenging to accurately estimate entrance and departure times.

In this paper, we propose PlaceWalker, a scheme that uses a low-power accelerometer in the background to continuously monitor *user's significant activity intensity changes*. As a humanistic geographer Seamon illustrated in his book [12], we typically move around a small number of places in which we perform our daily routines associated with various physical activities, ranging from low intensity activities such as resting and sitting to moderate and high intensity activities such as walking, running, and swimming. This means that the intensity level of physical activity varies widely from place to place, and further, the intensity level while transitioning from one place to another is typically different from the intensity level while staying in a place. For instance, when a person moves from one place to another, significant activity intensity changes naturally occur due to the kinematics of normal human walking (e.g., starting walking or stopping walking) [13]. Our key observation is that a change of place requires a series of physical activities, and the corresponding intensity levels would vary widely. In other words, a significant intensity level change of physical activity provides a useful clue to the change of place, and we can use it to trigger Wi-Fi scanning for place detection.

The main departure from existing work is that PlaceWalker triggers Wi-Fi scanning only when such an activity shift is detected, and a change of place is then validated by comparing Wi-Fi signatures. When compared with existing accelerometer-assisted localization [11,14], our approach obviates the need of using computationally expensive statistical classifiers for place detection. At the same time, detecting a significant change of physical activity is very different from existing *movement detection* schemes used in the literature [15,16] including SensLoc. For example, while SensLoc only considers whether stationary or non-stationary (e.g., sitting or walking), PlaceWalker keeps track of intensity level changes between stationary and non-stationary modes in a systematic way (e.g., sitting to walking and walking to sitting). By doing so, it can provide more detailed context of place logging (e.g., arriving at a place, noise while staying in a place, leaving from a place, and noise while moving), which will be further discussed in Section 3.2. Moreover, this systematic way of tracking provides better chances to suppress unnecessary periodic Wi-Fi scanning for place detection.

To validate its efficiency, we build a prototype system and evaluate the performance via extensive experimentation. Our results show that detecting a significant change of physical activity can precisely capture arrival/departure at/from a place and can trigger Wi-Fi scanning for place detection only when it is needed. As far as false triggering and detection failures are concerned, our experimental results confirm that their impacts are minimal. PlaceWalker significantly lowers the energy consumption by as much as 60.9%, when compared with the state-of-the-art method [9]. We validate the experimental results with a simple analytic model and analyze energy efficiency under varying parameter settings.

2. PlaceWalker overview

The overall architecture of PlaceWalker platform is presented in Fig. 1. A mobile device runs client software that monitors the activity intensity changes and detects places. When a client detects a place, it reports this event to the remote server. If a user does not have always-on Internet connectivity, this information can be synchronized later on (e.g., when a user re-connects to a local Wi-Fi). From this, the remote server learns places and populates a Wi-Fi fingerprint database that will be used for place matching. The web front-end allows users to review the place visit history and the associated path trails, and to manage places from their perspectives.

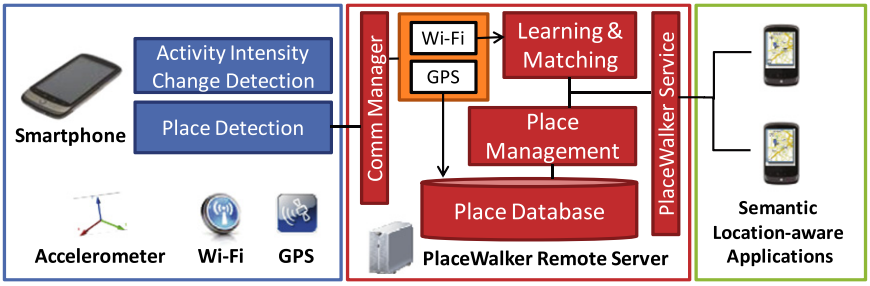


Fig. 1. Overview of PlaceWalker architecture.

We perform place logging in three steps, namely (1) activity intensity change detection, (2) place detection, and (3) place learning/matching. In activity intensity change detection, we leverage the fact that there is a strong correlation between a significant intensity change of physical activity and a change of place. For example, when a user moves from one place to another, there are clear changes of physical activity due to the kinematics of normal human (e.g., starting walking or stopping walking) [13]. PlaceWalker periodically monitors the intensity change of user's physical activity by using a low-power accelerometer in the mobile device. In PlaceWalker, a change is detected when two consecutive windows of user's acceleration data are significantly different. However, a significant change of physical activity may not always guarantee a change of place. When an activity intensity change is detected, we verify whether a place has actually changed or not, through checking if there is a significant change of the neighboring Wi-Fi beacon set.

Once the mobile device detects a place (i.e., arrival to a place), PlaceWalker performs either *place learning* or *place matching*. The Wi-Fi beacon fingerprint of a detected place is transmitted to a remote PlaceWalker server with the timestamp. The server then performs *place matching* by comparing this fingerprint with the fingerprints of existing places in the database. If there is a matched fingerprint, then the server retrieves the corresponding place information to the mobile device. Otherwise, the server registers a new place with the received fingerprint. In the latter case, for geo-tagging the server asks the mobile client of its current location, which can be obtained by GPS, cell-tower based localization, or Wi-Fi Positioning System (WPS). As soon as the server receives the geo-location from the mobile, it registers the new place to the database along with the place's beacon fingerprint so that the place can be identified in the future matching.

3. Place logging algorithm

We illustrate the key components of place logging, namely activity intensity change detection, place detection, and place learning/matching.

3.1. Activity intensity change detection

We move around a small number of places in which various activities are performed (or the interactions of an individual and space-time activities associated with places), and explore/encounter the rest of the world [12]. In most cases, we walk while moving around places, except for a few cases of entering a place with a vehicle (e.g., a drive-in theater). In physiology, normal human walking is described as a series of distinct stages: (1) development stage (from rest to some velocity); (2) rhythmic stage (some constant average velocity); and (3) decay stage (coming back to rest) [13]. When moving from one place to another, we will have a series of such stages. Even when we are interrupted in the middle while moving (e.g., waiting for a green light), we will have a sequence of this series.

Our observation is that the normal human walking pattern (when moving around places) is significantly different from the routine activities in places (e.g., standing, browsing, sitting, etc.). The walking stages can effectively characterize entrance/departure to/from a place: i.e., a user starts walking to leave a place (development stage), continues walking (rhythmic stage), and arrives at a next place (decay stage). There will be significant activity intensity changes at the development and decay stages (departure/arrival). This means that an event of a significant activity intensity change provides a reasonable clue for detecting entrance or departure points. After detecting a significant activity intensity change, we will then perform Wi-Fi scanning to confirm whether entrance or departure has actually happened.

We use a low-power accelerometer to measure the activity intensity of a user. For a given window size of T_w, we estimate the activity intensity using variance of the acceleration readings. Assume that a series of n acceleration samples $= \{a_1, a_2, a_3, a_n\}$ were collected. For each sample a_k, we calculate the magnitude of the force vector by combining the acceleration values from all three axes: i.e., $M(a_k) = \sqrt{a_k.x^2 + a_k.y^2 + a_k.z^2}$. The variance is then simply given as $1/(n-1) * \sum_{i=1}^{n}(M(a_i) - Avg(M))^2$. We use activity intensity from two consecutive windows (denoted as A_1 and A_2) and calculate the Rate-of-Change (ROC): $|A_1 - A_2|/\min\{A_1, A_2\}$. The ROC indicator has been widely used to analyze trends of stock prices in the field of technical analysis [17]. If the resulting ROC is above a threshold value T_{ac}, then we declare that there is a significant change of physical activity. One caveat is that the ROC indicator is susceptible to background noise. For instance, ROC may

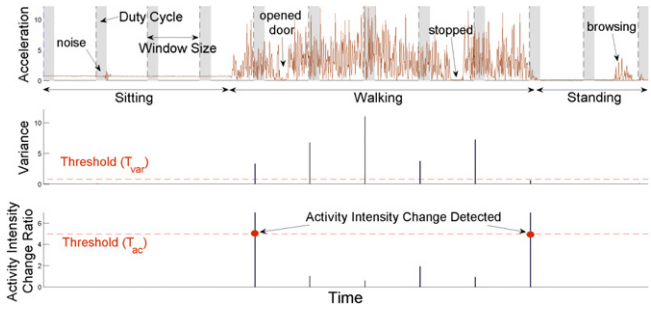


Fig. 2. Activity intensity change detection example (T_{var}, T_{ac}).

be large if we compare two small values (e.g., 0.001 vs. 0.0001). To suppress this case, we perform ROC calculation only if there is a considerable movement; i.e., at least one of the windows has activity intensity greater than T_{var}.

For accurate intensity change detection, we have to carefully tune three important parameters, namely *window size*, *duty cycling*, and *sampling rate*. Since a single value of acceleration is not meaningful to determine movement, we measure the activity intensity by computing the variance of a group of acceleration readings in a sampling window. Duty cycling is used to save energy because continuous accelerometer sampling is energy consuming. Sampling rate is defined as the number of readings returned by an accelerometer per second. As shown later, the higher the rate of sampling, the more the energy consumption, and yet we need enough readings for variance calculation. In PlaceWalker, we use 3–5 Hz sampling rate (i.e., Normal mode in Android platform).[1]

We then need to carefully select the window size (denoted T_w) as it determines the minimum duration of a detectable activity. As illustrated earlier, normal walking when moving from one place to another lasts for a certain duration (based on the distance between places and the average speed). Given that the window size is T_w, the minimum duration of an always-detectable activity is $2 * T_w$ as a window may start at an arbitrary point. In PlaceWalker, we use the window size of $T_w = 30$ s with duty cycling rate of 20% for energy efficiency (i.e., sampling only for 6 s). There is a trade-off in the window size: if we lower the window size (and accordingly increase the duty cycling rate), the minimum duration of a detectable activity will reduce (i.e., shorter walking can be captured), but this comes at the cost of more energy consumption. Also, there is a concern of false alarms. As shown later, a false alarm will trigger Wi-Fi scanning which is used to check whether a user is arriving/departing. Given that human walking patterns are not random and tend to last certain duration, the impact of false alarms would be limited. Our experimental result confirms that the occurrence of false alarms is small enough, and thus, PlaceWalker can significantly reduce the energy consumption without any impact on fidelity (on average 60.9%).

Fig. 2 illustrates the process of our activity intensity change detection. In the top figure, we depict the accelerometer data with 20% of duty cycling (shaded area on the top) using the magnitude of a reading. The middle figure shows the activity intensity that is the variance of the magnitude values in a sampling window. The bottom figure shows the resulting rate-of-change. This figure clearly shows that our approach could accurately detect both departure (standing to walking) and arrival (walking to sitting).

3.2. Place detection based on Wi-Fi beacons

The activity intensity change detection module continuously monitors the user's physical activity. When there is a change of physical activity, it could be one of the following scenarios: (1) arriving at a place, (2) noise while staying in a place, (3) leaving from a place, and (4) noise while moving. This section presents how the system distinguishes these cases.

Case 1—*arriving at a place*: We begin with the case when a user arrives at a place. As soon as a change of physical activity is detected, the system turns on the Wi-Fi interface to scan nearby Wi-Fi beacons. Scanning is performed twice: one scan is followed by another, and interval between scans is 30 s (called scan interval), which is an adjustable system parameter. The system then computes the cosine similarity between those two consecutive scanned beacon sets by measuring the cosine of the angle between two vectors of RSSI values. The resulting similarity ranges from 0 indicating independence, to 1 meaning exactly the same and in-between values indicating intermediate similarity. If the similarity is greater than a threshold, T_{sim}, then we consider the user has just arrived at a place, and a place flag is set to be true. For robustness, if the similarity is in range between T_{sim_retry} and T_{sim}, then we scan Wi-Fi beacons one more time and compute the similarity using the latest beacon sets. When a place is detected, the mobile combines two consecutive scan sets and builds a Wi-Fi signature for the detected place. The mobile reports the information of the detected place (i.e., Wi-Fi signature and timestamp) to the PlaceWalker remote server for a learning/matching process.

[1] Note that Android has a set of fixed sampling rates, and our measurement using a Nexus One shows the following results: Normal (3–5 Hz), UI (8–10 Hz), Game (14–16 Hz), and Fastest (22–25 Hz).

To be precise, an RSSI vector of the scan result is defined as $\vec{r} = (rssi_{1,r}, rssi_{2,r}, \ldots, rssi_{t,r})$ where t is the total number of distinct Wi-Fi hotspots, and $rssi_k$ is the RSSI value from Wi-Fi beacon index k. Given two RSSI vectors $\vec{r_1}$ and $\vec{r_2}$, the *cosine of the angle* between $\vec{r_1}$ and $\vec{r_2}$ is given as:

$$sim(\vec{r_1}, \vec{r_2}) = \frac{\vec{r_1} \cdot \vec{r_2}}{|\vec{r_1} \times \vec{r_2}|} \tag{1}$$

$$= \frac{\sum_{i=1}^{t} rssi_{i,r_1} rssi_{i,r_2}}{\sqrt{\sum_{i=1}^{t} rssi_{i,r_1}^2}\sqrt{\sum_{i=1}^{t} rssi_{i,r_2}^2}}. \tag{2}$$

Case 2—*noise while staying in a place*: The activity intensity change detection could be also triggered while a user is in a certain place or is on the move. The mobile then scans Wi-Fi beacons and computes the similarity of the current scan result and the scan result of previously detected place. If they are similar, then we consider this case as a false alarm and simply ignore it. In this case, uninteresting movements are captured within the sampling window such as short browsing.

Case 3—*leaving from a place*: When they are not similar, the user might be moving to a new place. Wi-Fi beacon sets will be significantly different since the second beacon set is scanned after the scan interval (a user has already departed). Thus, T_{sim} will be close to zero. When it happens, the mobile notifies the remote server that a user has left the place.

Case 4—*noise while moving*: While a user is on the move, false positives could occur particularly when a user is interrupted in the middle (e.g., stopping at the red light). If a user waits for a significant amount of time, then it could be detected as a new place (*Case* 1). Otherwise, the user resumes walking; the Wi-Fi beacon sets will be different, and thus, no place will be detected.

3.3. Place learning and matching

The place learning/matching is done in the remote server. As soon as a mobile transmits a beacon signature collected in a discovered place, the server finds the best match from the previously logged information using the cosine similarity. If the database is empty or the matching fails, then we consider the detected place is a new place, and the server registers the place information to the database. While the geographical coordinate is submitted by the mobile device, the details of the place information such as name and type can be manually updated by the user later or semi-automatically by the system. In the beginning, most discovered places will find no match. As time passes, the database size will grow, and the probability of matching will increase.

4. Experiment

4.1. Data gathering and energy profiling

Data gathering: We collected accelerometer, Wi-Fi, and GPS data using Google Nexus One mobile phones that run Android 3.2 (Honeycomb). The accelerometer sensor mode was set to Normal mode with sampling rate of 3–5 Hz, and the sampling intervals of Wi-Fi and GPS were set to 15 s and 2 min, respectively. We elected eight participants who use Google Nexus One as their primary phone in daily lives, and installed our logging software in their phones. We drew participants from researchers and graduate students and collected data from total 8 participants (6 males, 2 females). Participants varied in age; 3 were between 21 and 30 years; 3 were between 31 and 40 years. Participants were asked to always carry the phone with them for about a week. Since continuous logging drains the battery in 6–8 h, we asked each participant to occasionally charge the phone when the battery level is low while staying at a place. To collect the ground-truth, we developed a piece of software as a part of our logging system that allows the participants to click the media button on the earpiece for voice tagging of the current place information (i.e., entrance and departure). We use pseudonyms to protect participants' privacy, namely Allan, Bobby, Clayton, Darren, Erik, Faith, George, and Hannah who collected 8, 7, 7, 6, 4, 6, 7, and 6 days of logged data, respectively.

Energy measurement: We performed fine-grained energy profiling of Google Nexus One. We disassembled a Nexus One's battery by separating the battery pack from the battery compartment. We then inserted a 0.02 Ω measurement resistor in series between a battery terminal and its connector on the phone. We used a National Instruments USB-6210 DAQ (up to 16 channels at 250 kHz) to measure the voltage drop across the phone battery and also the voltage drop across our measurement resistor. We collected the data directly from DAQ using Matlab. We started the measurement in the airplane mode with the screen off (23.8 mW). We then activated each component to measure the power consumption and the measurement lasts for 10 s. We repeated this process 20 times and reported the average power consumption in Table 1.

Our findings can be summarized as follows. First, Wi-Fi's power consumption in the idle mode (no association or scanning) is minimal (only 5.39 mW). It appears that Android puts the Wi-Fi interface into the sleep mode when it is idle. A single Wi-Fi scan takes around 3 s and consumes 915 mJ (over the entire steps in Android: registering call-back, starting a

Table 1
Power consumption of Nexus One. Wi-Fi scanning is performed once in each measurement period, and a single Wi-Fi scanning costs 915 mJ on average.

Wi-Fi idle	Wi-Fi scan	3G idle	Wi-Fi/3G idle	Acc	Acc + NO	Acc + UI	Acc + GM	Acc + FA	GPS/In	GPS/Out	3G/Net
5.3 mW	305.0 mW	10.4 mW	16.9 mW	8.3 mW	61.1 mW	120.1 mW	160.5 mW	168.5 mW	417.4 mW	387.1 mW	405.7 mW

scan, and receiving the results via call-back, and unregistering call-back). Second, we find that an accelerometer consumes 8.25 mW in the idle mode. Nexus One is equipped with BMA150 3-axis accelerometer; its energy consumption in the specification is under 1 mW, which is much lower than the measured value. Further, we find that most energy consumption comes from reading values from the accelerometer; i.e., whenever a call-back function is invoked, we observe a spike in the measured data. Thus, energy consumption is roughly proportional to the sampling rate, but as rate increases, it starts deviating; i.e., Normal: 61 mW (<5 Hz), UI: 120 mW (<10 Hz), Game: 160.51 mW (<16 Hz), and Fastest: 168.47 (< 25 Hz). Finally, we measured the energy consumption of localization (which is used for geo-tagging). We find that turning on/off GPS takes about 0.2 and 2 s, and GPS fix takes about 2 s, which was also reported in the literature [15]. GPS consumes less power outdoors.

4.2. Experiment results

The goal of our experiments is three-fold: (1) evaluating the performance of activity intensity change detection under various configurations of system parameters, (2) evaluating the performance of place detection with activity change monitoring, and (3) comparing the energy consumption of PlaceWalker with that of existing works.

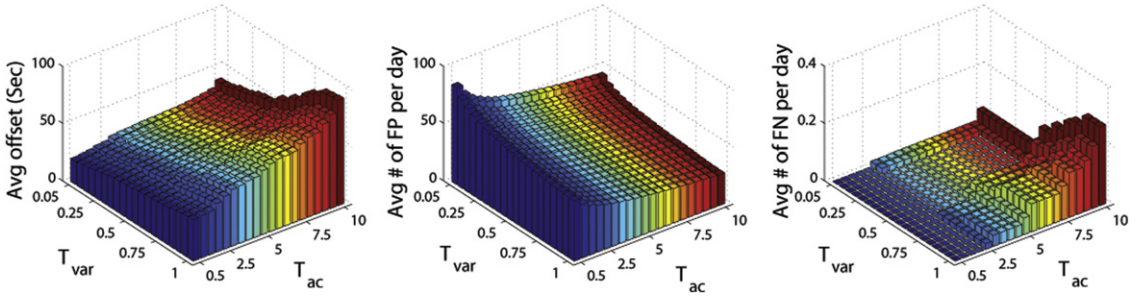
Activity intensity change detection: In Fig. 2, we presented two important threshold values: T_{var} and T_{ac}, which have a great impact on the performance of activity intensity change detection. Recall that a window with variance over T_{var} is considered for ROC calculation, and if the ROC is over T_{ac}, then an activity change is detected. We introduced T_{var} to reduce the number of false alarms (or false positives (FP)). Nonetheless if we set this value too high (hoping to reduce FPs), then we will experience many cases of failing to detect places; we call this misdetection or false negative (FN). Likewise, decreasing T_{ac} makes more false positives, while increasing T_{ac} makes more false negatives (or more cases of detection failures). We performed a sensitivity analysis of these parameters to systematically understand the impact.

For the analysis, we introduce another metric called *time offset* that is the time difference *between the ground-truth and the nearest event of a change of physical activity*. We use time offset to show how these thresholds affect the performance. We use the entire dataset collected by eight participants. We consider the two different duty cycling scenarios, 20% and 100% and use the window size of 30 s. Our manual investigation shows that there are total 671 ground-truths in the dataset. The ranges of T_{var} and T_{ac} are from 0.05 to 1 and from 0.5 and 10 with 0.05 and 0.5 intervals, respectively. We consider possible combinations of T_{var} and T_{ac} and calculate the time offset, the number of false positives, the number of false negatives (misdetection). Note that we assume that a false negative or misdetection has occurred if we find no change of physical activity within $\pm$8 min from the ground-truth. The results are reported in Fig. 3.

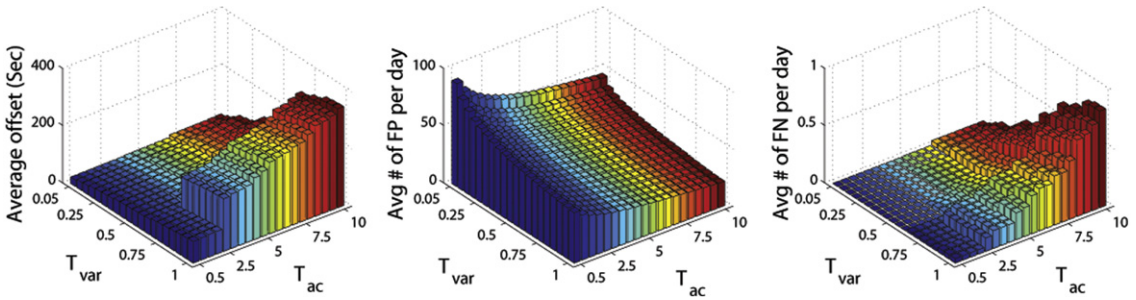
The result shows that there is a trade-off between the time offset and the number of FPs in both duty cycle settings. If T_{var} and T_{ac} are set to be low, then the detection time will be much faster (shorter time offset and lower false negatives), but we spend more energy since there are more false positives each of which triggers Wi-Fi scanning. In contrast, if both values are increased, then we have a less number of false positives, but experience more delays for detection. In PlaceWalker, we put more weight on the energy consumption because 10–20 s of delay is acceptable, as long as false negatives are reasonably low. Note that the average offsets are abnormally high in Fig. 3(b) especially when T_{var} and T_{ac} are higher than 0.75 and 2.5. This irregularity is caused by false negatives or misdetections. Since for time offset calculation, we use the nearest event of an activity change from the ground-truth, false negatives will significantly increase the average offset.

The figure also shows that we can set T_{var} and T_{ac} to larger values when using 20% of duty cycling. Surprisingly, there are more false negatives if duty cycling is not used. This phenomenon is due to the fact that in normal human walking, the activity intensity (in terms of variance) gradually increases during the development stage (start walking), reaches its maximum during the rhythmic stage, and gradually decreases during the decay stage (stop walking). Suppose walking begins in the middle of a window, the variance of this window will be small. As a user continues walking, the activity intensity gradually increases, and the variance of the next window could be greater than T_{var}. However, the resulting ROC may be smaller than T_{ac}. Therefore, both T_{var} and T_{ac} should be set to smaller values to guarantee 100% detection when using no duty cycle. If 20% of duty cycling is used (e.g., 6 s out of 30 s), then we can avoid sampling the period of development/decay stages and thus, we can better detect a change of activity (even with much lower energy consumption).

We then analyzed how the window size and the duty cycle affect the accuracy of activity intensity change detection. We set T_{var} to 0.75 and T_{ac} to 5 and measured the time offset distribution by varying the duty cycle and window size. We set the duty cycle rate from 10% to 100% with a 10% interval and use two different window sizes, 30 and 60 s. The results are illustrated in Fig. 4. When the window size is 30 s, we observe that the duty cycling rate of 20%–30% performs the best. If the duty cycling rate is set too low (to save energy), then we will have higher delay and more false negatives (because smaller number of samples results in less accuracy). When the window size is 60 s, regardless of the duty cycle rates, the occurrences of false negatives are much higher than those with 30 s. The reason is that the large window results in misdetections of the



(a) Duty cycle: 20%.



(b) Without duty cycle.

Fig. 3. Threshold analysis (avg. time offset, number of FPs, and number of FNs).

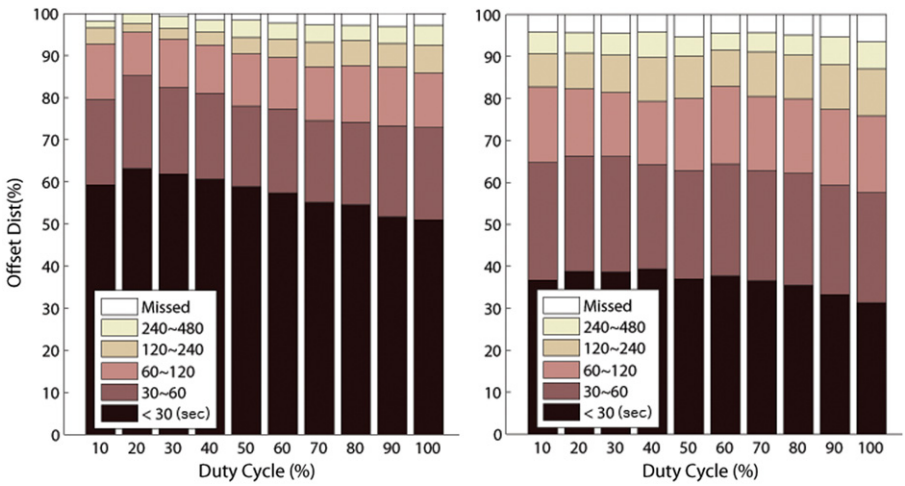


Fig. 4. Impact of duty cycling and window size (left: 30 s, right: 60 s).

activities whose duration is not long enough (say less than 2 min). To this reason, we decided to set the default values of our system parameters as follows: $T_{var} = 0.75$, $T_{ac} = 5$, duty cycle $= 20\%$, and 30 s of window size.

When the participants collect data, they are asked to log when they arrive/leave to/from a place. Through the experiment, we notice that people have a diverse sense of place. For instance, a participant stopped by a bus stop for a while but she did not log the place as a place while another participant did. This diversity of place definition makes it hard for us to evaluate the system because our system will find the bus stop as a place as long as there is a change of activity and stable Wi-Fi beacon observations. To make this clear, we divide the detected places into two groups: Interested Place, and Not Interested Place (NI). Not-interested places are the places that the system detects, but participants did not label them with voice tagging. Note that the chance of missing labeling is low (as we asked users to record twice: entrance and departure). In Fig. 5, a true positive (TP) means the correct detection of a place of interest. We consider the event of detecting a not-interested place as a false positive (FP-NI). A false positive can also occur in a place (FP-IP) or while moving (FP-WM). For instance, a user might move for a very short time to grab a cup of coffee. Or, the user might frequently stop while moving from one place to another at the stop signals.

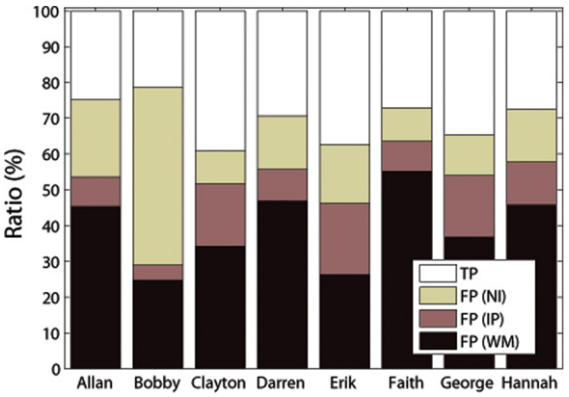


Fig. 5. Distribution of True Positive (TP) and False Positive (FP) with Not-Interested (NI), In Place (IP), and While Moving (WM).

Table 2
Results on place detection, learning, and matching.

	Allan	Bobby	Clayton	Darren	Erik	Faith	George	Hannah
Interested	73	50	48	43	12	34	43	30
Not Interested	18	9	26	11	6	11	23	11
False	0	0	0	0	1	0	0	0
Merged	1	1	2	0	0	0	1	0
Divided	4	12	3	7	3	1	5	4
Learned	39	55	41	34	15	43	49	34
Matched	56	24	33	24	6	5	24	15

We now compare the rates of TP and FP. In our dataset, we do not find any false negative with the current parameter setting. The results are shown in Fig. 5. This graph reveals several interesting observations. First, activity intensity change detection works well in general (no false negative). The rate of false positives is not significant; around the half of the detected activity changes identify the change of place, i.e., TP + FP (NI). Bobby has higher false positives compared with others. After an interview with him, we found that he cleaned his office for several hours (most FPs caused by this), and he also visited several large places such as a grocery store, and a large sushi buffet restaurant. Even with this false positive rate, we were able to significantly save energy on average when compared with other schemes. Second, the definition of place is very diverse. We observe that the range of places that Allan, Bobby, Darren, Erik, and Hannah defined is quite wide (i.e., low rate of non-interested places, (FP-(NI))), whereas that of the rest is somewhat strict; for instance, for them, a bus stop and a place for smoking are not considered as a place.

Place detection, learning, and matching: We present the place detection, learning, matching results in Table 2. We counted the number of detected interested, detected not interested places, false negatives (i.e., misdetection), merged places, divided places, learned places, and matched places. If two places are too close, then the Wi-Fi beacon signature may fail to differentiate one from another, and the system will report that a user was at a single place (we call this a *merged place*). If a place has a number of sub-places (e.g., a building and a mall with distinct names), then we simply call these sub-places *divided places*. As time passes, the system will learn new places. The number of learned places is the number of distinct places that the system learned over time. The number of matched places counts places for which the system found matched places in the database.

Most of detected-interested places were home, work, and restaurants. Detected-not-interested places include parking lots and waiting/stopping places while moving. PlaceWalker found all the interested places except one restaurant visited by Erik where there were no Wi-Fi beacons nearby. This error can be corrected by using GPS and 3G network mixed localization, which is part of our future work. Bobby and Darren have a number of divided places, because they often visited grocery or department stores. For place learning, we observe that the learning rate dramatically decreases after a few days and the rate increases over the weekends. This confirms the fact that people move around a small number of familiar places.

Benchmark study: We compared PlaceWalker with continuous Wi-Fi scanning and SensLoc that is the most recent work on place logging and is known as the most energy efficient scheme [9]. Given that both PlaceWalker and SensLoc use Wi-Fi signature for place detection, and PlaceWalker does not have any false negatives (i.e., misdetection), the performance of place detection would be on par. We now shift our focus on the energy consumption by comparing PlaceWalker with SensLoc and a naïve periodic Wi-Fi scanning.

The following configurations were used for the evaluation. In *PlaceWalker*, we uses two thresholds T_{var} and T_{ac} for the activity detection and one threshold T_{sim} for the Wi-Fi beacon comparison. We set T_{var} and T_{ac} as 0.75 and 5, respectively. T_{sim} was set to 0.65. The window size was 30 s and the accelerometer duty cycle was 20%. For *SensLoc*, we used the same parameter setting as shown in the original paper [9]. The Wi-Fi scanning interval was set to 10 s, and a window size of

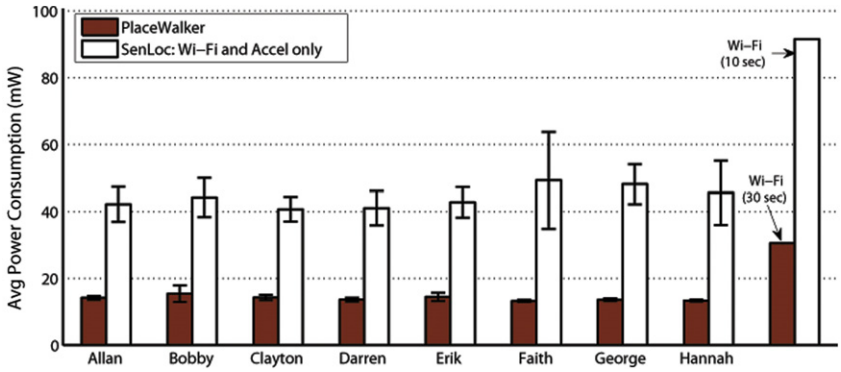


Fig. 6. Energy consumption comparison.

30 s was used. As it uses multiple scans in a window, they use the similarity threshold of 0.7 for the place detection.[2] After a user enters a place and spends at least 5 min with less movement (e.g., sitting), SensLoc suppresses Wi-Fi scanning, and an accelerometer is then turned on to detect a movement. The accelerometer was duty cycled at 50%, and a variance threshold v_{mov} was set to 0.2 for the movement detection. In *continuous Wi-Fi scanning*, we use two different periodic scanning intervals, i.e., 10 and 30 s.

For each participant, we compute the average power consumption in each day (in Watts) by dividing the total energy consumption (in Joules) by the duration of logged data (in seconds). Daily power consumption statistics of participants are averaged, and we present the average value with standard deviation in Fig. 6. As the result shows, PlaceWalker significantly reduces the energy consumption when compared to continuous Wi-Fi scanning and SensLoc. More specifically, PlaceWalker lowers the energy consumption by 60.9% on average when comparing to SensLoc. Comparing to continuous Wi-Fi scanning, PlaceWalker improves the energy consumption by 82% and 46% when the scanning intervals are 10 and 30 s, respectively. From the study, we have found that SensLoc consumes more energy particularly when a user stays outdoor longer. PlaceWalker's energy efficiency is attributed to the following reasons: (1) PlaceWalker uses duty cycling aggressively (only 20% of a window) and yet, this provides better change detection when compared with no duty cycling, (2) an accelerometer in the mobile device is energy efficient, and its cost is cheaper than Wi-Fi scanning especially when it is duty-cycled, and (3) the use of activity intensity change detection significantly reduces the number of Wi-Fi scanning.

5. Energy efficiency analysis

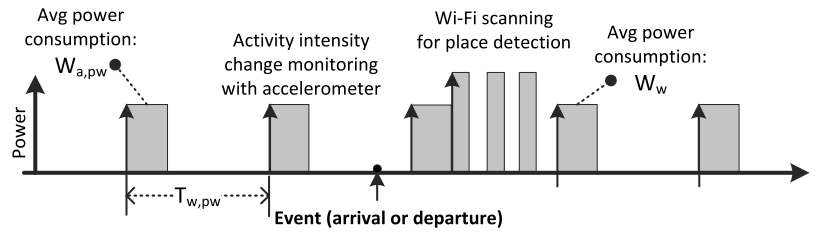
We perform a simple mathematical analysis on the energy consumption of PlaceWalker and SensLoc to systematically understand the impact of the false positives and the fraction of time being outdoors (or for non-stationary activities).

As illustrated in Fig. 7, we use the following configuration for modeling. In PlaceWalker, we use the window size of $T_{w,pw}$ for activity monitoring, which consumes on average $W_{a,pw}$ Watts (accelerometer). In SensLoc, we use the window size of $T_{w,sl}$, and Wi-Fi scanning is periodically performed for every $T_{w,sl}/m$ seconds where m is the number of Wi-Fi scans per window. For the fair comparison, we simply assume the same window size in both cases, namely $T_{w,pw} = T_{w,sl}$. In the following, we simply use T_w to denote the window size. If an event (arrival or departure) happens, then both schemes will attempt to detect the event. For the sake of analysis, we simply assume that both schemes will perform Wi-Fi scanning for $m = 3$ times, which consumes $T_w * mW_w$ mJ.[3] If a user arrives at a place, then we assume that SensLoc immediately switches to the movement detection mode; i.e., it monitors the accelerometer for every $T_{w,sl}$ seconds to detect whether a user becomes non-stationary.

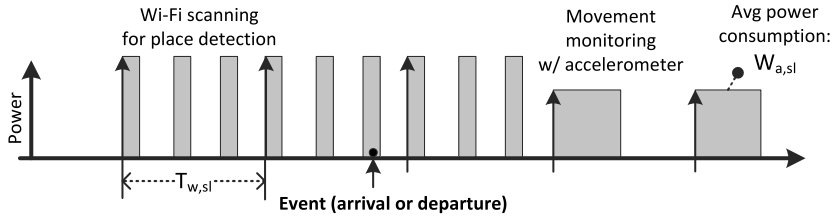
As shown in the figure, event detection happens over a block of two consecutive windows. We can generally assume that event detection in a block is *independent* from previous blocks. Thus, we can simply treat that an event actually happens randomly in any window (or random shuffling of windows has happened), and the analysis of a single block of consecutive windows is sufficient for energy modeling. For a given day, if a user has visited total N_P places over $N * T_w$ period of time, then the total number of events (entrance and departure) is $2N_P$, and a random window observes an event with probability $P_t = 2N_P/N$. The probability of false alarms is denoted as $P_f = N_F/N$ where N_F is the average number of false alarms observed. The probability of false positives of PlaceWalker and SensLoc is defined as $P_{f,pw}$ and $P_{f,sl}$ respectively. Given that SensLoc's performance depends on the probability of being outdoors, we assume that a random window is being observed outdoors with probability P_o.

[2] If we use two scans in a window with the same threshold as ours, PlaceWalker still outperforms SensLoc by 40.1% on average.

[3] PlaceWalker basically performs Wi-Fi scanning twice by default and an additional Wi-Fi scanning is performed only when the results are less satisfactory. In practice, the average number of Wi-Fi scanning should be smaller than three.



(a) PlaceWalker.



(b) SensLoc.

Fig. 7. Illustration of model parameters. The scan window of PlaceWalker and SensLoc is given as $T_{w,pw}$ and $T_{w,sl}$ respectively. The average power consumption of accelerometer monitoring of PlaceWalker and SensLoc is given as $W_{a,pw}$ and $W_{a,sl}$ respectively. The average power consumption of single Wi-Fi scanning over the scan interval T_w is W_w. Here, the subscript "*pw*" denotes PlaceWalker, and "*sl*" denotes SensLoc.

We first model the energy consumption of PlaceWalker. We consider the energy consumption in a block of two consecutive windows. If an event happens in the first window (probability of P_t), then a series of Wi-Fi scans will be performed in the second window. The expected energy consumption is given as $T_w P_t (W_{a,pw} + mW_w)$. Wi-Fi scanning can be also triggered by a false alarm, which happens with probability $P_{f,pw}$. Thus, the expected energy consumption is $T_w P_{f,pw}(W_{a,pw} + mW_w)$. If nothing happens, then the expected energy consumption is simply given as $T_w(1 - P_t - P_{f,pw})2W_{a,pw}$. The expected power consumption can be derived by dividing the total expected energy consumption by the duration of a block (i.e., $2T_w$).

$$W_{PW} = \frac{1}{2}\left(W_{a,pw} + P_t mW_w + P_{f,pw} mW_w + (1 - P_t - P_{f,pw})W_{a,pw}\right). \tag{3}$$

This equation shows that PlaceWalker's energy consumption is largely dependent on $W_{a,pw}$ and $P_{f,pw}$. PlaceWalker can use duty cycling more aggressively to reduce the power consumption. However, this comes at the cost of increased false positives and negatives. Our experiment results clearly show this trade-off. When comparing duty cycling of 20% and 10%, we find that the average number of false positives/negatives per day has changed from 19.9 and 0 to 20.7 and 0.23, respectively. Thus, we conclude that if users are willing to tolerate the false negatives, then it is better to perform duty cycling more aggressively.

To model the energy consumption of SensLoc, we consider a block of two consecutive windows as earlier. When a user is outdoors and performs non-stationary activities, the average power consumption is always mW_w regardless of whether an event happens or not. Recall that in SensLoc, Wi-Fi scanning is periodically performed in the background to detect a place. Thus, the expected energy consumption of a block is simply $P_o * 2T_w * mW_w$. We now consider the case when a user is indoors and becomes stationary. The overall calculation of energy consumption is very similar to that of PlaceWalker except that the departure events are only considered in SensLoc. When a departure event from a place happens in the first window (with probability of $P_t/2$), a series of Wi-Fi scans will be performed in the second window—SensLoc is then transitioning to the continuous Wi-Fi scanning mode. In this case, the expected energy consumption is given as $T_w P_t/2(W_{a,pw} + mW_w)$. Wi-Fi scanning triggered by a false alarm costs $T_w P_{f,pw}(W_{a,pw} + mW_w)$ mJ. If none of these events happen, then the expected energy consumption is simply given as $T_w(1 - P_t/2 - P_{f,pw})2W_{a,pw}$. Thus, the expected power consumption of SensLoc is simply given as follows:

$$W_{SL} = P_o mW_w + \frac{(1 - P_0)}{2}\left(W_{a,sl} + \frac{P_t}{2} mW_w + P_{f,sl} mW_w + \left(1 - \frac{P_t}{2} - P_{f,sl}\right) W_{a,sl}\right). \tag{4}$$

Given these equations, we now find the upper bound probability of false positives in PlaceWalker. If the false positive probability is greater than this threshold, then the energy consumption of PlaceWalker becomes worse than that of SensLoc. We can find the upper bound by using the inequality, namely $W_{PL} \leq W_{SL}$. By arranging this inequality, we have the following:

$$P_{f,pw} \leq \underbrace{\frac{2(W_{a,sl} - W_{a,pw})}{mW_w - W_{a,pw}}}_{T_1} + \underbrace{\frac{mW_w - W_{a,sl}}{mW_w - W_{a,pw}}}_{T_2} \underbrace{\left(2P_o + (1 - P_o)\left(\frac{P_t}{2} + P_{f,sl}\right)\right)}_{T_3} - P_t. \tag{5}$$

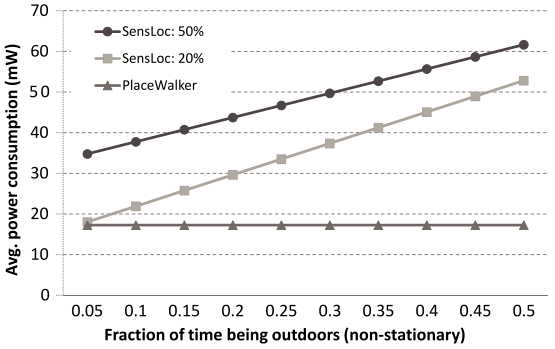


Fig. 8. Expected power consumption under varying fractions of time being outdoors (or non-stationary activities).

From this inequality, we find that the upper bound probability of false positives in PlaceWalker depends on several factors. First of all, accelerometer duty cycling efficiency is critical. This factor is captured by T_1 and T_2 in Inequality 3. As PlaceWalker uses more aggressively duty cycling, these values will increase, and so does the upper bound. Yet, the caveat is that as illustrated earlier, aggressive duty cycling will in turn increase false positive/negative probabilities. The term T_3 shows that the upper bound is also dependent on the probability of being outdoors (or performing non-stationary activities). When a user is outdoors, the gain is a factor of $2T_2$. Otherwise, the gain is mainly attributed to SensLoc's false positive probability. Interestingly when both systems use the same value for accelerometer duty cycling, namely $W_{a,sl} = W_{a,pw}$, the inequality reduces to $P_{f,pw} \leq P_o(2 - P_t) + (1 - P_o)P_{f,sl} - P_t/2$—as before, the upper bound is mainly dependent on the probability of being outdoors, and SensLoc's false positive probability.

We now plug the measured values of one of the participants Bobby into the model. From Table 1, we have $W_w = 30.5$ mW, $W_{a,sl} = 30.5$ mW (50% duty cycling over a 30 s interval), $W_{a,pw} = 12.2$ mW. Since he visited 6.58 places per day over 10.06 h on average (i.e., 1208 windows), the probability of an event is given as $P_t = 0.011$ (i.e., 13.16 events over 1208 windows). The false positive probability is calculated using N_F/N_O where N_F is the number of false positives, and N_O is the number of windows in which accelerometer monitoring has been used. Recall that our experimental results show that we have fairly low false alarm probabilities, namely $P_{f,pw} = 0.091$ (on average 109.98 events per day), and $P_{f,sl} = 0.033$ (on average 32.71 events per day). In Fig. 8, we plot the expected power consumption under varying fraction of time being outdoors. Our model well approximates the actual power consumption (e.g., 16.8 mW vs. 18.0 mW for PlaceWalker). The figure shows that as the fraction of time being outdoors increases, SensLoc's power consumption linearly increases, whereas PlaceWalker's power consumption is not influenced by this factor. For Bobby, SensLoc's power consumption is given as 44.15 mW, and this means that the average fraction of time for non-stationary activities is around 21%, which is close to the measured value. In Fig. 9, we plot the cumulative distribution of the fraction of time for non-stationary activities for all the participants. The figure clearly shows that participants occasionally performed prolonged non-stationary activities, and PlaceWalker effectively handles such cases.

To summarize, our analytic model shows that (1) aggressive duty cycling can be used to save energy as long as users can tolerate the errors, namely false negatives, and (2) the larger the fraction of non-stationary activities per day, the higher the energy saving of PlaceWalker over SensLoc.

6. Related work

Most place learning methods use spatial clustering over periodically sampled location trace data to discover significant places that a user has visited (known as geometry-based methods). A number of different spatial clustering strategies were used in the past, ranging from simple fixed radius based clustering [1] to more sophisticated clustering based on multiple radii [5], *k*-means [2], density [18], and Dirichlet processes [6]. Temporal information of a location trace can be also leveraged to realize spatio-temporal or multi-level clustering of location traces [19,7,20].

Unlike spatial clustering, Hightower et al. [3] showed that Wi-Fi beacons can be used to enable fine-grained place logging. This approach is much simpler than the existing room-level Wi-Fi localization in that we do not need to build Wi-Fi signature database for every place a priori [21]. The main benefit of the Wi-Fi signature based method is that it supports fine-grained place learning/detection (sub-building level). Further, Kim et al. [4] systematically showed that it is possible to capture entrance and departure times. If accurate time estimates are not required and short stops can be safely ignored, then a long scan interval (say as large as 2–5 min) can be also used for place logging as in i-Loc [22]. For accuracy existing schemes mostly use short scan intervals of 10–30 s [3,4,9], all leading to quick battery drainage. To lower the energy consumption, Kim et al. [9] recently proposed SensLoc that uses a low-power accelerometer to suppress Wi-Fi scanning while a user stays at a place with less mobility (e.g., sitting). As illustrated earlier, this approach is less effective when users engage in outdoor activities (e.g., travelling and shopping). The main departure from existing work is that PlaceWalker removes the root cause of battery drainage, namely continuous background Wi-Fi scanning. We use a low-power duty-cycled accelerometer to

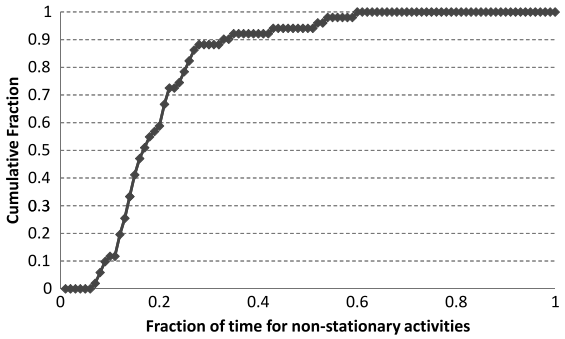


Fig. 9. Cumulative distribution of the fraction of time for non-stationary activities.

detect a significant activity intensity change. Further, PlaceWalker is much simpler and more energy efficient than existing accelerometer-assisted localization schemes [11,14] in that it does not require laborious training, energy hungry periodic sensing with multiple sensors, and sophisticated classification algorithms.

Recently Chon et al. [23] proposed SmartDC that uses mobility prediction to adaptively schedule Wi-Fi scanning based on the learned mobility patterns. However, the energy saving of SmartDC comes at the cost of detection failures and less accurate time estimates. While predicting next significant places to visit can be done fairly accurately [24], it is challenging to accurately predict arrival/departure times due to the innate uncertainties of human behavior. While SmartDC does not use a low-power accelerometer, PlaceWalker's activity intensity change detection module can be integrated to improve the energy efficiency and detection accuracy of SmartDC. It is part of our future work to enhance PlaceWalker with mobility prediction capabilities.

7. Discussion

As illustrated earlier, one of the key system components of PlaceWalker is activity intensity change detection. We mainly used the metric to detect activity change and thereby to discover a change of place. In general, this metric would be applicable to activity detection such as transport mode recognition (still, walk, run, bike, motor) [25]. We expect that activity intensity would be very different from one another (e.g., still vs. walk; and walk vs. run). For a given activity change, ROC values would vary widely. We can measure the range of these values and properly set the threshold values to accurate activity recognition. Moreover, prior knowledge about activity changes can be used to further optimize the detection algorithms; e.g., considering transition probabilities from one to another activity. One concern is that if a user is on a motor, then we may falsely recognize it as "still" due to low acceleration in a motor. Our measurement data show that there is constant background acceleration due to vehicle movements. Yet, due to low variances (when compared with those of other activities such as walk, run, and bike), accurate detection would be challenging. Fortunately, as long as we can detect a change of activities (e.g., from walking to still in a motor), we can use Wi-Fi scanning to confirm whether a user is in a stationary place. If Wi-Fi signatures continue to change rapidly, then we can infer that a user is in a motor. Alternatively, when GPS is used, we can use speed of movements for more accurate detection. It would be an interesting future work to experiment with activity detection.

For localization, Wi-Fi fingerprints have been widely used due to fine-grained location learning/detection (e.g., RADAR). In the context of localization, we can treat PlaceWalker as a localization algorithm that provides room-level location information. As in existing localization schemes that consist of two steps: learning and detection, PlaceWalker has learning and detection components. Since users are moving from one place to another, our algorithm can be considered that a location map is continuously built, although labels of detected places should be manually tagged by the users. Existing localization algorithms have varying range of detection latency (e.g., from few hundreds of milliseconds to tens of seconds). PlaceWalker's detection latency would be much greater than existing localization algorithms. However, as shown in our analysis section, we can trade latency improvements for energy consumption. Depending on the application types, we can set proper system parameters in order to meet the application requirements.

However, key application of PlaceWalker is logging user's place transitions in an energy efficient manner so that it only requires a place-level granularity. To achieve this, Wi-Fi beacons can be used to enable place logging, which does not require building laborious Wi-Fi signature database for every place a priori, energy hungry periodic sensing with multiple sensors, and sophisticated classification algorithms. As PlaceWalker uses a low-power duty-cycled accelerometer to detect a significant activity intensity change, this will cause 10–20 s of delay, which limits real-time pedestrian navigation. Further investigation on localization granularity and system analysis on user activity monitoring also remain as our future work.

8. Conclusion

We proposed PlaceWalker, an energy-efficient, fine-grained place logging platform. Given that there exists a strong correlation between a significant change of physical activity and a change of place, PlaceWalker monitors any significant

activity intensity changes using a low-power duty-cycled accelerometer and triggers Wi-Fi scanning only if such an activity shift is detected. Our experimental results showed that activity intensity change detection can precisely capture arrival/departure times and PlaceWalker significantly lowers the energy consumption without a noticeable fidelity loss. We validated the experimental results with a simple analytic model and analyzed the energy efficiency under varying parameter settings.

Acknowledgment

This work was supported by the IT R&D program of MSIP/KEIT [10041313, UX-oriented Mobile SW Platform].

References

[1] N. Marmasse, C. Schmandt, Location-aware information delivery with comMotion, in: HUC'00.
[2] D. Ashbrook, T. Starner, Using GPS to learn significant locations and predict movement across multiple users, in: Pers Ubiquit Comput 2003.
[3] J. Hightower, S. Consolvo, A. LaMarca, I. Smith, J. Hughes, Learning and recognizing the places we go, in: Ubicomp'05.
[4] D.H. Kim, J. Hightower, R. Govindan, D. Estrin, Discovering semantically meaningful places from pervasive RF-beacons, in: Ubicomp'10.
[5] N. Toyama, T. Ota, F. Kato, Y. Toyota, T. Hattori, T. Hagino, Exploiting multiple radii to learn significant locations, in: LoCA'05.
[6] P. Nurmi, S. Bhattacharya, Identifying meaningful places: the non-parametric way, in: Pervasive'08.
[7] J.H. Kang, W. Welbourne, B. Stewart, G. Borriello, Extracting places from traces of locations, in: WMASH'04.
[8] S. Isaacman, R. Becker, R. Caceres, S. Kobourov, M. Martonosi, J. Rowland, A. Varshavsky, Identifying important places in people's lives from cellular network data, in: Pervasive'11.
[9] D.H. Kim, Y. Kim, D. Estrin, M.B. Srivastava, SensLoc: sensing everyday places and paths using less energy, in: Sensys'10.
[10] N.E. Klepeis, et al. The National Human Activity Pattern Survey (NHAPS): a resource for accessing exposure to environmental pollutants, US EPA.
[11] A. Ofstad, E. Nicholas, R. Szcodronski, R.R. Choudhury, AAMPL: accelerometer augmented mobile phone localization, in: MELT'08.
[12] D. Seamon, A Geography of the Lifeworld: Movement, Rest and Encounter, Palgrave Macmillan, 1979.
[13] D.H. Sutherland, K.R. Kaufman, J.R. Moitoza, Kinematics of normal human walking, in: Human Walking, Lippincott Williams & Wilkins, 2014.
[14] M. Azizyan, et al. SurroundSense: mobile phone localization via ambience fingerprinting, in: Mobicom'09.
[15] M. Baun, J.L. Kjærgaard, T. Godsk, T. Toftkjær, EnTracked: energy-efficient robust position tracking for mobile devices, in: Mobisys'09.
[16] J. Paek, J. Kim, R. Govindan, Energy-efficient rate-adaptive GPS-based positioning for smartphones, in: Mobisys'10.
[17] Rate of change (ROC). http://en.wikipedia.org/wiki/Momentum_%28technical_analysis%29.
[18] C. Zhou, D. Frankowski, P. Ludford, S. Shekhar, L. Terveen, Discovering personally meaningful places: an interactive clustering approach, in: GIS'04.
[19] R. Hariharan, K. Toyama, Project lachesis: parsing and modeling location histories, in: GISience'04.
[20] R. Montoliu, D. Gatica-Perez, Discovering human places of interest from multimodal mobile phone data, in: MUM'10.
[21] P. Castro, P. Chiu, T. Kremenek, R.R. Muntz, A probabilistic room location service for wireless networked environments, in: Ubicomp'01.
[22] Y. Ma, R. Hankins, D. Racz, iLoc: a framework for incremental location-state acquisition and prediction based on mobile sensors, in: CIKM'09.
[23] Y. Chon, E. Talipov, H. Shin, H. Cha, Mobility prediction-based smartphone energy optimization for everyday location monitoring, in: Sensys'11.
[24] C. Song, Z. Qu, N. Blumm, A.-L. Barabasi, Limits of predictability in human mobility, Science (2014).
[25] S. Reddy, M. Mun, J. Burke, D. Estrin, M. Hansen, M. Srivastava, Using mobile phones to determine transportation modes, ACM Trans. Sensor Netw. (TOSN) (2014).