iFrame: Dynamic indoor map construction through automatic mobile sensing

Article *in* Pervasive and Mobile Computing · December 2016

DOI: 10.1016/j.pmcj.2016.12.008

2 authors, including:

Chen Qiu
Michigan State University
10 PUBLICATIONS **151** CITATIONS

ARTICLE IN PRESS

Pervasive and Mobile Computing ▮ (▮▮▮▮) ▮▮▮–▮▮▮

Contents lists available at ScienceDirect

Pervasive and Mobile Computing

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

Fast track article

iFrame: Dynamic indoor map construction through automatic mobile sensing☆

Chen Qiu *, Matt W. Mutka

Department of Computer Science and Engineering, Michigan State University, East Lansing, MI, USA

ARTICLE INFO

Article history:
Available online xxxx

Keywords:
Indoor map construction
Smartphone
Mobile sensing

ABSTRACT

Many pervasive computing applications depend upon maps for support of location based services. Individuals can determine their location outdoors on these maps via GPS. Indoor pervasive applications may also need to know the layout of buildings, however indoor maps are less prevalent. This paper presents iFrame, a dynamic approach that leverages existing mobile sensing capabilities for constructing indoor floor plans. We explore how iFrame users may collaborate and contribute to constructing 2-dimensional indoor maps by merely carrying smartphones. A deployment study shows iFrame is an unattended approach that provides a skeleton map of a real building effectively and automatically.

1. Introduction

Map availability is essential for Location Based Services [1], such as tracking and localization services. In outdoor environments, Google Map, Yahoo Map and other services provide outdoor maps for users of mobile devices. Unfortunately, maps are much less commonly available for indoor environments. In order to provide indoor floor plans, many people may need to dedicate significant time to construct the floor plans and make them available to commercial channels. Google Indoor Map [2] has collected over 1000 indoor floor plans of airports and shopping malls in US and Japan. These maps are built manually, which may be tedious and high cost. Moreover, even if the indoor maps are constructed, the layouts within these buildings may often change. This leads to our question: *Can we generate a dynamic indoor map automatically and accurately without complex data training?*

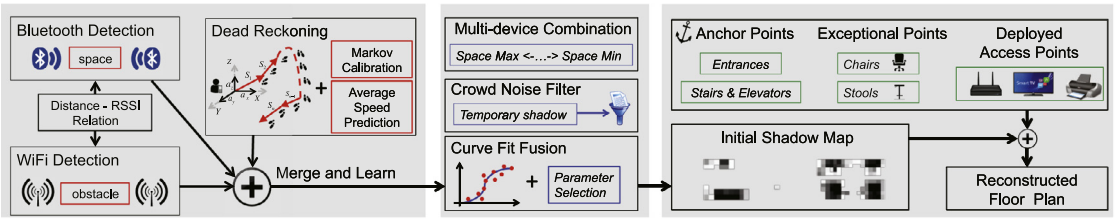
Some solutions for building indoor floor plans have been proposed by researchers. JigSaw [3] employed camera sensors to collect images. After applying point cloud algorithms, they piece together images to construct the indoor map. Although this approach can build the indoor map using smartphones, the procedure of image processing is challenging. CrowdInside [4] and MapGENIE [5] adopt dead reckoning [6] to obtain the motion traces of mobile device users. By analysis of the characteristics of accelerations, these approaches recognize different layouts within indoor environments. Nevertheless, the accuracies of dead reckoning and the computational complexity need to be further improved.

In this paper we propose `iFrame` [7], an approach that leverages mobile sensing data to construct dynamic floor plans of complex indoor environments automatically. `iFrame` does not require commercial negotiation with providers of indoor maps. By opening the `iFrame` application on smartphones and by walking around the targeted indoor environments, users passively construct the floor plan. We abstract the indoor map as an $m \times n$ matrix. Grids in the matrix are described by a value between two extreme states: 0 - the grid is vacant, 1- the grid is completely occupied by objects. The value between 0 and 1 represents the percentage of the area that an object occupies the grid.

As illustrated in Fig. 1, users leverage dead reckoning to obtain their motion traces. For locations a user visits that are identified by dead reckoning, the grid location will be marked vacant. Because dead reckoning may deviate from the

☆ Fully documented templates are available in the elsarticle package on CTAN.

http://dx.doi.org/10.1016/j.pmcj.2016.12.008
1574-1192/



(a) Inertial Sensing on Smartphones. (b) System Refinement. (c) Indoor Floor Plan Construction.

Fig. 1. Overview of iFrame.

ground truth, we seek to enhance its accuracy and therefore the proper identification of the status of grids in the matrix by employing (1) a Markov chain and average speed prediction to calibrate the trace deviation and (2) Bluetooth and WiFi radios to compensate for the deviation errors. We assume the Bluetooth RSSI received from another device to be related to distance [8]. We suppose abrupt changes in WiFi signal strength is related to obstacle detection [9]. When a user turns on the Bluetooth option and if a Bluetooth adapter can detect other Bluetooth devices, then we set the grids on the link between Bluetooth adapters to 0 (vacant). Since WiFi signal strength is more sensitive to obfuscation rather than to distances, and if the WiFi signal strength decreases abruptly, then the grids that on the link are recorded as 1. Because the common discovery procedures for Bluetooth and WiFi components are time consuming, it may cause time delay to receive sensing information. We speed up the discovery procedure by interrupting the scanning mechanism at the program level. Considering the imperfections of each method, Curve Fit Fusion (CFF) is introduced to combine the methods for improving the output matrices. By executing `iFrame` on 3–5 users' smartphones iteratively within 5–10 min, we can obtain a two dimensional floor plan for a 12 m $\times$ 6 m office room. Even if `iFrame` users walk daily by an unattended mode, the system is able to generate the floor plan of a room within 7 h. Since the generated map is built by real time sensing information, even if the layouts of rooms change, the changes can be represented on the map. By introducing Anchor Points (such as entrances, stairs, and elevators), the traces generated by dead reckoning can be initialized and calibrated. As numbers of `iFrame` users who share their information increases, the constructed floor plans generated from a larger dataset drawn from a larger set of users will have a lower error rate and will be generated within a shorter amount of time.

Our work therefore makes following contributions: First, although some approaches have adopted crowd sourcing as a means to rebuild the indoor floor plans, `iFrame` is the first of its kind to measure RSSI values with other scanned mobile devices to help construct the layout of indoor environments. Second, we design a sensor fusion approach to combine and enhance the indoor maps computed by dead reckoning, Bluetooth, and WiFi RSSI detections. Third, `iFrame` is convenient to deploy and may be used passively by smartphones' users. The mobile sensing mechanism allows `iFrame` to represent the layout changes of buildings. As more users collaborate, improved resolution and clarity of the maps are generated. In addition, `iFrame` can cooperate with common wireless Access Points to further enhance the accuracies of indoor map. Some special elements in an indoor map, such as doors, chairs, elevators can be represented on the generated floor plan. The Bluetooth and WiFi detection speeds satisfy the real-time requirement of mobile computing.

2. System design

2.1. Overview of our approach

`iFrame` leverages existing smartphone and mobile device technology without the need of complex training to construct floor plans within buildings. Before constructing an indoor map, we formulate the unknown map as a matrix. Each element in the matrix represents a grid on the map. Each grid is marked in one of two states: empty or object. As shown in Fig. 1, the procedure of map construction is composed of three phases:

(1) `iFrame` is a system based on mobile sensing. Smartphone users adopt the sensing techniques to set the status of each element in the matrix iteratively: (i) dead reckoning, (ii) Bluetooth, and (iii) WiFi detections. By interrupting the scanning periods, we improve the speed that both Bluetooth and WiFi detect other devices.
(2) Based upon the data collected by sensors on smartphones, `iFrame` fuses the output matrices of the three methods via Curve Fit Fusion (CFF), which is supported on a cloud server. Then, `iFrame` combines the matrices built from multiple mobile devices and filters the noises caused by the *temporary shadow*.
(3) `iFrame` draws the layout in each room and hallway. By introducing Anchor Points, Exceptional Points, and deployed AP, the map of a indoor building is constructed completely.

Please cite this article in press as: C. Qiu, M.W. Mutka, iFrame: Dynamic indoor map construction through automatic mobile sensing, Pervasive and Mobile Computing (2017), http://dx.doi.org/10.1016/j.pmcj.2016.12.008.

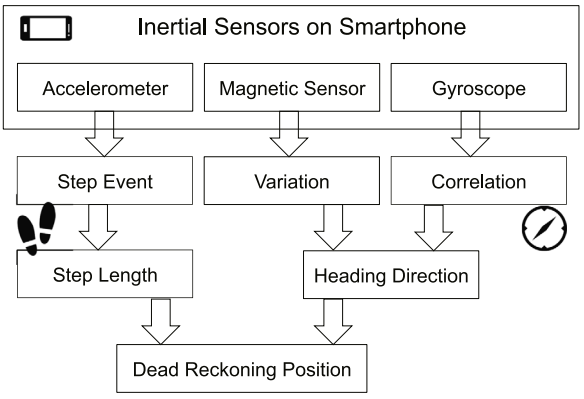


Fig. 2. Workflow of dead reckoning.

2.2. Map matrix

Initially, we divide the two dimensional map into grids, and each grid is processed as an element in a matrix $\mathbb{M}$. All grids are squares with the same size. If a grid is vacant, it is defined as an *empty* grid. We set the value of corresponding element in the matrix to 0. If a grid is totally occupied by an object, it is defined as an *object* grid. The corresponding element in the matrix is 1. The Shadow Rate is the amount that an object partially occupies the matrix element. If the size of the grid is 1 m^2 and the objects in the grid occupy 0.65 m^2, we define the shadow rate of the grid as 0.65. $V_{i,j}$ refers the value of a element in matrix $\mathbb{M}$. The matrix $\mathbb{M}$ is as follows:

$$\mathbb{M} = \begin{bmatrix} V_{1,1} & \cdot & \cdot & V_{1,n} \\ \cdot & \cdot & & \\ \cdot & & \cdot & \\ V_{m,1} & & & V_{m,n} \end{bmatrix}. \tag{1}$$

2.3. Dead reckoning detection

2.3.1. Basic dead reckoning approach

When a user enters a grid on the map, the grid has empty space for the user to occupy. The grids on the user's motion traces will have their status marked as *empty* (0). Hence, the user's motion traces in a building are able to describe the layout of the building.

`iFrame` uses dead reckoning [6,10] to obtain the user's motion traces. Based on the accurate initial position, the application executing on the mobile device computes movement distance in each segment continuously. Then, it will form the whole trace of the user's motion. For dead reckoning, as shown in Fig. 2, we leverage sensors on the smartphone (accelerometer, magnetometer, and gyroscope) to estimate the user's step length and obtain heading direction [11].

Unfortunately, dead reckoning has limitations: if the acceleration and orientation values from the smartphone do not reflect the human body's acceleration, for example, when the smartphone is held in a user's hand, and the hand shakes, the obtained accelerations are incorrect. The generated trace will deviate from the ground truth. Furthermore, the accelerometers on most smartphones are not highly accurate. A common sensor used for indoor localization [12] may have an error of 308 m within one minute due to a 0.5° deviation occurs on the orientation sensor.

2.3.2. Trace prediction and calibration by Markov Chain

To improve the accuracy of dead reckoning, we proposed a trace prediction approach relying on a Markov chain. A Markov chain is a common method to enhance pedestrian dead reckoning [13]. In our design, a Markov chain is a sequence of random variables $X_1, X_2, \ldots, X_n$. Given the present state, if both conditional probabilities are well defined, the future is conditionally independent of the past as formula (4).

$$P(X_{n+1} = x | X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n) = P(X_{n+1} = x | X_n = x_n). \tag{2}$$

Then, we abstract the Markov chain as a directed graph:

(1) Each grid in the map refers to a node. The edges in the graph are labeled by the probabilities of moving from one grid at time n to the other grid at time $n + 1$.

(2) As Fig. 3, the user can only move to a neighbor grid/node from time n to time $n + 1$. This action is represented by the transition matrix from time n to time $n + 1$. For each grid, since the number of potential transition grids are 8, the size of

Please cite this article in press as: C. Qiu, M.W. Mutka, iFrame: Dynamic indoor map construction through automatic mobile sensing, Pervasive and Mobile Computing (2017), http://dx.doi.org/10.1016/j.pmcj.2016.12.008.

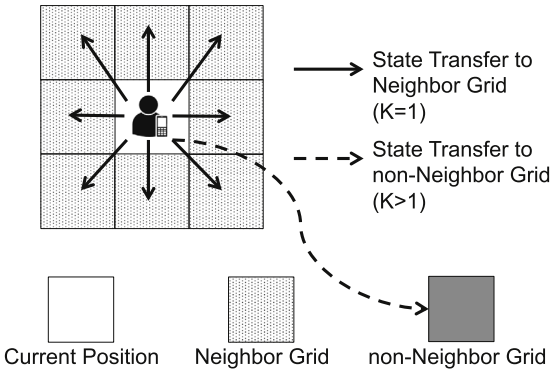


Fig. 3. Markov chain state transition. The user moves from current grid to neighbor grid.

transition matrix is 8×8 . If the user goes to a non-neighbor node, it needs more than one step. The value k is the number of steps for transferring grids.

$P\left\{X(n_m + k) = j | X(n_m) = i_m\right\}$ is the transition probability by moving k steps at time n. It is recorded as $P_{i,j}(n, n + k)$ for short. It indicates the probability that the user is in the state/grid i at time n, and after the transition of k steps, it goes to the state/grid j. $P_{i,j}(n, n + k)$ depends on the initial state/grid i, the final state/grid j, and the number of steps k instead of time n. Therefore, the transition probability of k steps can be defined as:

$$P_{i,j}(k) = P_{i,j}(n, n + k) = P\left\{X(n + k) = j | X(n) = i\right\}. \tag{3}$$

When $k = 1$, $P_{i,j}(1)$ is a one step transition probability. The matrix of one step transition is recorded as $P(1)$. For n steps transition probability, the transfer probability is $P_{i,j}(n)$. The corresponding matrix is the n step transition probability matrix, named $P(n)$. By applying C–K equation [14], we can obtain:

$$P(n) = P(1) \cdot P(n - 1) = P(n - 1) \cdot P(1) => P(n) = P(1)^n. \tag{4}$$

For $P(1)$, the corresponding $P_{i,j}$ value refers to the probability of moving from state/grid i to state/grid j. Since we can count the times from i to j as $num_{i,j}$, then

$$P_{i,j} = \frac{num_{i,j}}{\sum_{j=1}^{n} num_{i,j}} (1 \leqslant i, j \leqslant n). \tag{5}$$

After obtaining the one step transfer probability matrix, we are able to compute the transition probability matrix of k steps by $P(k) = P^k$.

To predict the user's selection for the next step, it is necessary to consider both the historical and the current information of the user's traces. The information that is closer to the current time period will have more influence on the user's decision for the next step. According to this strategy, we provide the prediction formula as follow:

$$X(t) = a_1 H(t - 1)P + a_2 H(t - 2)P^2 + \cdots + a_n H(t - n)P^n. \tag{6}$$

In formula (8), t is time period of next grid, t-1 refers to the previous time period of t. $X(t)$ is the prediction probability for the next grid/state. $H(i)$ denotes the states of the next grid's previous i grids. The factors $a_1, a_2, \ldots, a_k$ from 1 to k are used to decide their next grids ($a_1 \geq a_2 \geq \cdots \geq a_k$). Then, we select the maximum element in $X(t)$, and take the maximum element's corresponding grid as the prediction grid for the next step.

When the user predicts the grids that are his/her next moving targets, if the motion trace computed by formula (2) does not include the predicted grids, the acceleration values for this time period will be replaced by the retrieved accelerations in the previous time period. Then, our system recalculates the current segment of the motion trace. The above analysis is only for an individual device. Therefore, this probability is taken from only an individual device. The smartphone does not require the transition probabilities of the crowd.

2.3.3. Trace prediction and calibration by average speed

In this section, we introduce another approach to calibrate the deviations of dead reckoning by Analyzing the Average Speed of user's motion (AAS). When people carry smartphones or other mobile devices, the average speed of users in a certain time slot is correlated to the average speeds of previous time slots and following time slots [15]. Based upon this feature, for every i time slots, we curve fit the average speed values as the linear function and predict the range of average speed for the $i + 1$ time slot. Once the average speed in time slot $i + 1$ is out of the prediction range, the acceleration values will be recognized as incorrect values. The accelerations in $i + 1$ time slot will be replaced by the correct accelerations in the closest time slot.

Please cite this article in press as: C. Qiu, M.W. Mutka, iFrame: Dynamic indoor map construction through automatic mobile sensing, Pervasive and Mobile Computing (2017), http://dx.doi.org/10.1016/j.pmcj.2016.12.008.

2.4. Bluetooth detection

Received Signal Strength Indicator (RSSI), in units of "dBm", is a measurement of the power present in a received radio signal. RSSI can be recorded from the components of most mobile devices, such as WiFi and Bluetooth adapters. Bluetooth RSSI values received from the other devices are related to the distance between the devices. Shorter distances often represent stronger RSSI values [8]. We evaluated the RSSI-distance mapping relations for the Bluetooth Low Energy (BLE) adapters on Samsung Galaxy smartphones and Samsung Tablets. These mapping relations are pre-trained and stored as hash maps.

Algorithm 1 Bluetooth Detection Algorithm

Input:
1: (x_α, y_α), (x_β, y_β), $RecRSSI_\alpha$, $RecRSSI_\beta$, $V_{BL}(x, y)$;
Output:
2: The latest $V_{BL}(x, y)$;
3: // Use the prepared Hashmap to compute
4: // the estimated RSSI by known positions
5: $Dist(\alpha, \beta) \leftarrow \sqrt{(x_\alpha - x_\beta)^2 + (y_\alpha - y_\beta)^2}$
6: $EstRSSI \leftarrow$ HashMap($Dist(\alpha, \beta)$);
7: **if** $(RecRSSI_\alpha + RecRSSI_\beta)/2 - EstRSSI < T_b$ **then**
8: **for** each $V_{BL}(x, y)$ **do**
9: // Once the RecRSSI is close to Estimated RSSI,
10: // we add 0 for the elements on Bluetooth link
11: $V_{BL}(x, y) \leftarrow V_{BL}(x, y) + 0$;
12: cnt $\leftarrow$ cnt+1 & $V_{BL}(x, y) \leftarrow V_{BL}(x, y)$ / cnt;
13: **end for**
14: **end if**

If two devices can detect each other, we assume that the link between the two devices has open space. We estimate the distance between the two devices from the initial map. The positions of the two devices are obtained through dead reckoning. If the corresponding RSSI values from RSSI-distance relation are close to the received RSSI values (the error range is 3dBm), namely, we assume no interference on the link. The elements V_{BL} between the link in $\mathbb{M}$ are set as 0. When the received RSSI is less than the corresponding RSSI values (beyond the error range), there might be some interference on the link. It is unreasonable to set all the values of elements on the link in M as 0.

The procedure of Bluetooth detection is described in Algorithm 1. For the scenarios where we conduct the experiments, the fluctuation range of Bluetooth RSSI is around 3 dBm. We are able to control two factors that can change the RSSI fluctuations: (1) temperature in the room. The different temperatures may cause the obvious change of RSSI values. For the rooms we conducted experiments, the temperature are stable because of the air condition. (2) The power of BLE devices. If the powers of BLE devices are not enough strong, the variation of RSSI is more than the BLE devices have enough power. In our experiments, the BLE devices are fully charged. Namely, we keep the conditions that minimize the RSSI variation to enhance the accuracies.

2.5. WiFi detection

Wi–Fi Direct is supported by most current smartphones. In contrast to Bluetooth RSSI, WiFi RSSI values are not as sensitive to the distance within a short range. However, the value of WiFi RSSI is susceptible to the interference between the link [16]. It is thus desirable to uses WiFi to describe the *object* on the map.

For example, Alice and Bob are two users of `iFrame`. They build connections via Wi–Fi Direct first. Then, Alice walks from position 1 to position 2. On her way from position 1 to position 2, there is a cabinet between user Alice and Bob. Once Alice crosses the cabinet, the received WiFi signal strength from Bob has an obvious abrupt signal change. Based upon this phenomenon, we formulate the WiFi Detection in Algorithm 2.

In Algorithm 2, two events are pre-defined as follow:

Event (I): The differential value between the received RSSI value and the RSSI estimated by the distance is greater than the threshold T_{w1}.

Event (II): The WiFi RSSI between the two users drops abruptly (more than the threshold T_{w2}) from the previous period.

Only when the users want to make contributions to indoor map construction, the WiFi and Bluetooth adapters are broadcasting and scanning. The adapters will be idle when users do not build an indoor floor plan.

Fig. 5 demonstrates how the three sensing approaches work and perform.

2.6. Enhance real-time property

For most BLE devices detection, the discovering period of a BLE adapter is more than 20 s [17]. However, user's motion may change continuously. For example, once a certain round of BLE detection does not finish, the user's of smartphone

Please cite this article in press as: C. Qiu, M.W. Mutka, iFrame: Dynamic indoor map construction through automatic mobile sensing, Pervasive and Mobile Computing (2017), http://dx.doi.org/10.1016/j.pmcj.2016.12.008.

Algorithm 2 WiFi Detection Algorithm

Input:
1: (x_α, y_α), (x_β, y_β), $RecRSSI_\alpha$, $RecRSSI_\beta$, $V_{WL}(x, y)$;
Output:
2: The latest $V_{WL}(x, y)$;
3: **for** each time period i **do**
4: $Dist(\alpha, \beta) \leftarrow \sqrt{(x_\alpha - x_\beta)^2 + (y_\alpha - y_\beta)^2}$
5: $EstRSSI \leftarrow$ HashMap($Dist(\alpha, \beta)$);
6: **if** Event (I) and Event (II) are satisfied **then**
7: // Event (I):
8: // $|EstRSSI - (RecRSSI_\alpha + RecRSSI_\beta)/2| > T_{w1}$
9: // Event (II):
10: // $|(RecRSSI_\alpha + RecRSSI_\beta)_i - (RecRSSI_\alpha$
11: // $+RecRSSI_\beta)_{i-1}| > T_{w2}$
12: **for** each $V_{WL}(x, y)$ **do**
13: // When RecRSSI is close to Estimated RSSI,
14: // we add 1 for the elements on WiFi link
15: $V_{WL}(x, y) \leftarrow V_{WL}(x, y) + 1$;
16: cnt $\leftarrow$ cnt+1 & $V_{WL}(x, y) \leftarrow V_{WL}(x, y)$ / cnt;
17: **end for**
18: **end if**
19: **end for**

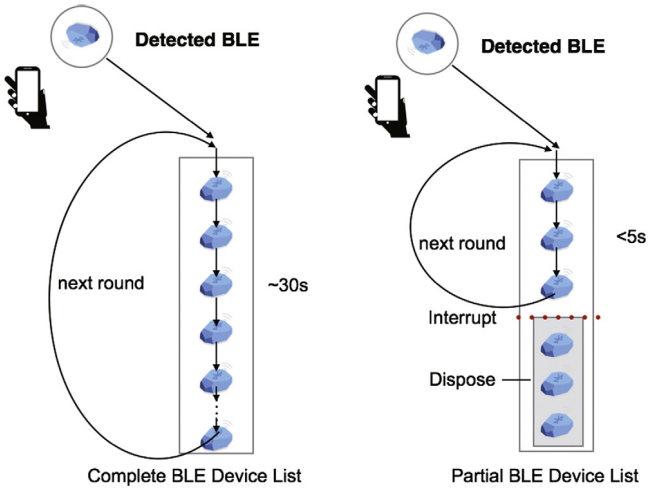


Fig. 4. Reduce the scanning period by interruption.

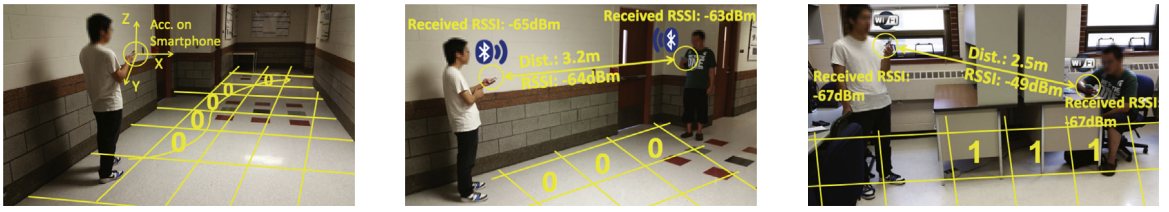
may have left the position or even the environment where he was. Namely, the BLE detection should be synchronized with smartphone's users' motion behaviors. In our approach, once our program has detected three BLE devices, we interrupt the discovering program and restart the new procedure. Based on our observation from Android BLE detection, as shown in Fig. 4, the average discovery round for 15 BLE beacons can be reduced from 40 to 5 s.

In addition, for WiFi detection, there exists the similar problem. We also "interrupt" the process of WiFi detection. By detecting three nearby WiFi devices, we restart the new scanning procedure and drop the following processing. The average discovery round for 10 WiFi devices can be reduced from 20 to 5 s.

2.7. Matrix fusion mechanism

We introduced three techniques: dead reckoning, Bluetooth detection, and WiFi detection to set the values on the map matrix. However, there are some shortcomings: (1) dead reckoning can describe the user's motion traces, but the estimated positions often deviate from the ground truth. Once a user's body motion is different from the smartphone's motion, dead reckoning computation will fail due to the deviation. (2) Bluetooth detection approach may ignore some interferences that are lower than the threshold between the link; (3) WiFi detection cannot represent some vacant grids on the link that have

Please cite this article in press as: C. Qiu, M.W. Mutka, iFrame: Dynamic indoor map construction through automatic mobile sensing, Pervasive and Mobile Computing (2017), http://dx.doi.org/10.1016/j.pmcj.2016.12.008.



(a) Motion trace detection. (b) Bluetooth detection. (c) WiFi direct detection.

Fig. 5. Approaches for constructing the map matrix $\mathbb{M}$.

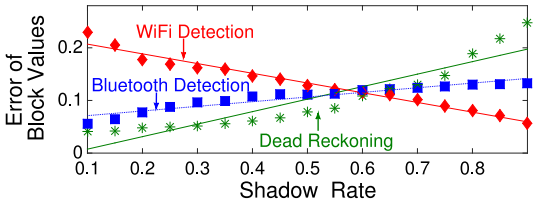


Fig. 6. CFF for collected sensing data.

detected objects. Therefore, the matrix computed by the three approaches includes errors. In order to refine the results, `iFrame` merges the three methods and generates improved results.

We proposed Curve Fit Fusion (CFF) to combine the matrices computed by the three methods. According to the samples from the training data, we select a proper proportion to assign different weights to the three methods.

$$\begin{cases} a+b+c=1 \\ a\times M_D + b\times M_B + c\times M_F = M \\ a:b:c = {B_D}^{-1} : {B_B}^{-1} : {B_F}^{-1} \end{cases} \tag{7}$$

$$B_{m\times n} = \sqrt{\sum_{j=1}^{n}\sum_{i=1}^{m}(\Delta V_{ij})^2},\ \Delta V = |V_g - V_e|. \tag{8}$$

As given in Eq. (9), M_D, M_B, M_F denote the matrices computed after using dead reckoning, Bluetooth and WiFi detections. M is the matrix computed by combining the three techniques. The factors a, b, and c decide the proportion of M_D, M_B, M_F. S_D, S_B, S_F refer to the shadow rates of the three approaches in a room.

A special metric is introduced. Error of Block Value (EBV) is the error value of the estimated shadow rate in each grid. V_g refers to the real shadow rate in each grid. V_e is the shadow rate computed by our approach. If we use an $m \times n$ matrix, the Error of Block Value is defined as $B_{m\times n}$, which is the average error value of all the grids.

We collect a small-scale training dataset: by changing the layouts in three rooms, for various types of layouts, each has a different shadow rate. Then, we compute the shadow rate and error value for each case. By applying different approaches, in Fig. 6, each sample on the figure is a case we tested. The figure illustrates the relation between the shadow rates and the error values for these cases.

As given in Eq. (11), B_D, B_B, B_F refer to the error values caused by dead reckoning, Bluetooth and WiFi detections. After applying the curve fit tool, we can compute the values of α and β. By knowing α and β, once we obtain the shadow rates, it is easy to calculate the proper a, b, and c values from Eq. (8).

$$(B_D, B_B, B_F) = (\alpha_1, \alpha_2, \alpha_3)\cdot(S_D, S_B, S_F) + (\beta_1, \beta_2, \beta_3) \tag{9}$$

2.8. Multi-device combination

Since `iFrame` is a crowd sourcing mechanism, all the users of mobile devices collect and upload sensing data. Therefore, it is significant how we organize data computed from each mobile device. Our system provides three types of organizations:

(1) Maximum Space: $M_{d_1} \bigcup M_{d_2} \ldots \bigcup M_{d_{n-1}} \bigcup M_{d_n}$
(2) Minimum Space: $M_{d_1} \bigcap M_{d_2} \ldots \bigcap M_{d_{n-1}} \bigcap M_{d_n}$
(3) Mean Value: $\sqrt{(M_{d_1}^2 + M_{d_2}^2 + \cdots + M_{d_{n-1}}^2 + M_{d_n}^2)/n}$.

Please cite this article in press as: C. Qiu, M.W. Mutka, iFrame: Dynamic indoor map construction through automatic mobile sensing, Pervasive and Mobile Computing (2017), http://dx.doi.org/10.1016/j.pmcj.2016.12.008.

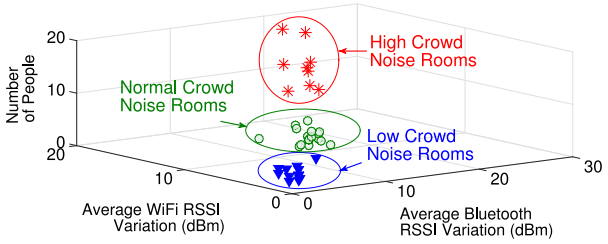


Fig. 7. Different levels of crowd noise for rooms.

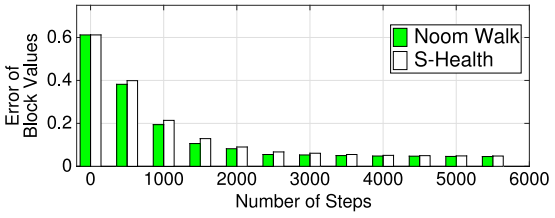


Fig. 8. Relation between the steps number and map rebuilt accuracy.

M_{d_i} denotes the matrix computed by device i. The Space Maximum represents the combination matrix that has the most space. The Space Minimum is the combination matrix that remains the least space. The Mean Value is between these two extremes. We adopt the Mean Value as the default mechanism to merging sensing data from different devices.

2.9. Crowd noise filter

One situation is non-negligible: if there exist so many users who are stationary in one room or a hallway, the generated shadow map might include some errors. The people's bodies may tend to yield some shadows in the generated map. To reduce these "temporary shadows", `iFrame` checks the motion trace of each user periodically and considers if a user does not change his/her position within 5 min, `iFrame` will not use his/her data until he/she leaves the position.

In addition, if many people are in a room or hallway, regardless if the people's bodies are stationary or not, there might be interference for the smartphones' radio signals. The presence of large numbers of people often cause variations of RSSI values [18]. Based on this phenomenon, we need to detect the room with high crowd density and amend the variations of the RSSI caused by the presence of many human's bodies.

Three steps are taken as a heuristic solution for filtering crowd noises:

(1) We divide the matrix $\mathbb{M}$ into sub-matrices from M_1 to M_n. Each keeps the same size. For M_i $(0 \leq i \leq n)$, in a certain time period, we focus on three features: the number of users, the sum of variation of Bluetooth RSSI for all the devices, and the sum of variation of WiFi RSSI for all the devices.

(2) By the three proposed features, we assume each M_i is a sample on a three dimensional space. The three features represent the three dimensions. By employing the k-means algorithm, we divide the sub-matrices into three types: high crowd noise level, normal crowd noise level, and low crowd noise level as shown in Fig. 7.

(3) For the matrices tagged with high crowd noise level, we take actions to reduce the noise: since the matrices M_B and M_F made by Bluetooth and WiFi RSSI detections are interfered by human bodies, these data will not be adopted. Only dead reckoning is acceptable for the matrices with high crowd noise level. Matrices with normal or low level crowd noise still use the three key techniques together.

2.10. Extend one room to multiple rooms or hallways

In order to apply `iFrame` to a real scenario, we extend our approach from one room to a building containing multiple rooms and hallways.

In a real building, some rooms or hallways have high shadow rates and others have low shadow rates. As mentioned by Curve Fit Fusion (CFF), the three detection techniques have their own features. The three techniques have different performance in rooms due to different shadow rates. Therefore, we need to assign different weights for the three methods.

We train our data by three basic types of rooms: a room with a low shadow rate, a room with a medium shadow rate, and a room with a high shadow rate. We use the formula in CFF to obtain three groups of a, b, c values: one for a crowded room, one for a normal room, and one for the room with few objects.

When a user enters a room or a hallway, we set the parameters a, b, c as 1/3 initially. Then, the user runs `iFrame` and computes the average shadow rate of each room for the first time period. If the shadow rate satisfies the low/normal/high shadow rate, for the next period, it will be computed by the corresponding group of a, b, and c.

Please cite this article in press as: C. Qiu, M.W. Mutka, iFrame: Dynamic indoor map construction through automatic mobile sensing, Pervasive and Mobile Computing (2017), http://dx.doi.org/10.1016/j.pmcj.2016.12.008.

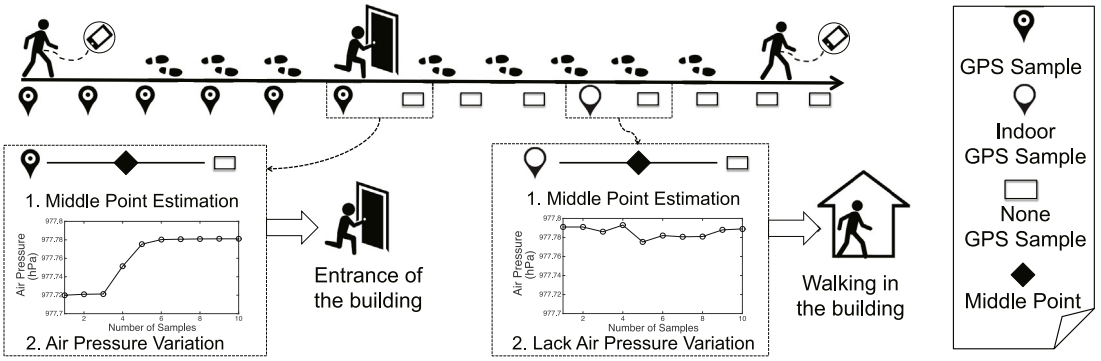


Fig. 9. Leverage GPS information and barometer to detect the building entrances.

2.11. Extend rooms to a building

2.11.1. Anchor points analysis

In addition to the rooms and hallways of a building, there are some special places, such as entrances, elevators, and stairs. When people walk in a building, these places should be recognized as the initial positions of dead reckoning and the joints of rooms/hallways. These places are named Anchor Points. Our existing approach can detect the layout of a room and hallway, nevertheless, it has not recognized Anchor Points.

In order to detect the Anchor Points in indoor buildings, first, we employ the techniques found in CrowdInside [4] and note that the motions of users have their own acceleration ranges: (1) stationary: 0–5 m/s^2; (2) elevators: 0–4.1 m/s^2; (3) walking: 0–13.2 m/s^2; (4) stairs: 0–20.2 m/s^2. `iFrame` analyzes the types of Anchor Points via their acceleration signatures. Then, we adopt (1) correlation between the acceleration values on different axes and (2) variance of acceleration magnitude to further classify Anchor Points. According to the obtained accelerations and analysis patterns proposed in CrowdInside, we identify Anchor Points on our map approximately.

Considering the entrances of a building are the initial positions of users' motion traces, we not only need to recognize them but also need to obtain their locations. When a smartphone receives a GPS sample, the user can assume he/she is in an outdoor environment. When a user enters a building, the GPS samples will disappear. A common method of estimating the entrance positions is recording the position where the latest GPS signal was received and the position where the GPS disappeared, and then compute the middle point as the position of the entrance. Actually, while a user is walking in a building, e.g., when he/she is close to a window, he/she might receive GPS samples on his/her smartphone occasionally. These GPS samples are named indoor GPS samples (iGPS for short). Therefore, the middle point between the position of receiving the latest GPS and the position where the GPS disappeared may contain errors.

Since the different temperatures and layouts in indoor and outdoor environments, the values of air pressure between indoor and outdoor environments may be different. When a user of smartphone enters a building, the values of air pressure will vary on the barometer that is integrated on the smartphone. As Fig. 9 shows, we adopt this phenomenon to detect the entrance of the building. When we compute the middle point between the position where the user received the latest GPS and the position where the GPS disappeared, and there exist a variation in air pressure, the middle point can be recognized as the entrance of the building. Based upon our observation, the variation of air pressure should be greater than 0.03 hPa. An additional explanation is necessary: this approach only can be applied for the places where indoor and outdoor environments have differential air pressures caused by temperature or other factors.

To identify stairs and elevators in a building, we leverage two types of information to recognize them: acceleration and air pressure. First, as the measurement in CrowdInside [4], when users of smartphones go on stairs and stand on elevators, the accelerations represent their own ranges. Second, when the users of `iFrame` cross levels in a building, the air pressure will change continuously. Since most modern smartphones integrate a barometer, the values of air pressure can be detected. As in Fig. 10, the variations of air pressure caused by changing levels in a building can be recorded by the barometer on smartphones. The numbers near the arrows refer to how many levels the user crossed. Therefore, when both of the two conditions are satisfied, we can identity the positions are elevators or stairs.

2.11.2. Hallway assembling

In a building, hallways are separated by walls. Computer vision based approaches often use complex image algorithms to gather the layouts separated by walls and the walls themselves. `iFrame` solves this problem by: (1) users cannot cross the wall, therefore, dead reckoning detection will not set the related element in $\mathbb{M}$ as 0; (2) if a user in a room and a user out of a room can detect each other, WiFi and Bluetooth detections cooperate together to mark the wall related elements as 1. More samples can rebuild the wall more clearly. This method does not need extra computation and can avoid image gathering.

Please cite this article in press as: C. Qiu, M.W. Mutka, iFrame: Dynamic indoor map construction through automatic mobile sensing, Pervasive and Mobile Computing (2017), http://dx.doi.org/10.1016/j.pmcj.2016.12.008.

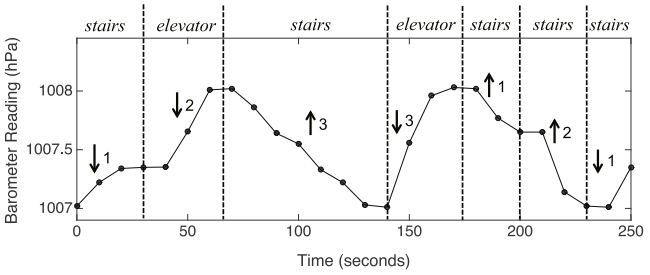


Fig. 10. The air pressure recordings of a smartphone user goes up and down in a building.

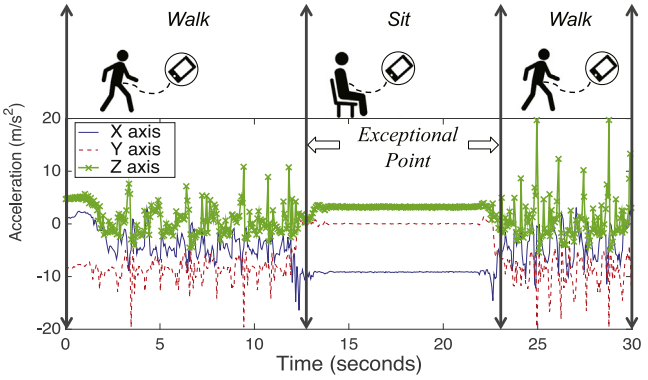


Fig. 11. The accelerometer recordings of a smartphone user passed by a chair (smartphone in left hand).

2.11.3. Exceptional points

In real environments, there exists some grids that are occupied by objects but still can be passed by users. These points includes three features: (1) from the perspective of dead reckoning, it can be passed through; (2) for Bluetooth and WiFi detections, it can be detected as an obstruction; (3) in a certain time periods, as illustrated in Fig. 11, the values of accelerations keep a relatively stable state. The parameters that define the stable state are discussed in the evaluation section. If these three conditions can be satisfied concurrently, we recognize the grid as an "Exceptional Point" and mark the corresponding grid as "1".

For example, if a chair is in the grid, even if the grid can be described as "occupied" on the shadow map, the user can sit on the chair and leave the chair at any time. For a certain chair, we observe the detecting results by using the three proposed conditions. We tested the case 100 times. Our approach could detect the chair 92 times as an "Exceptional Point". Some other objects on the floorplan can be treated as "Exceptional Point". For example, a shopping cart, mobile cabinet (cabinet including wheels) satisfy the three conditions because of their movable features. Although the chair can be detected by the definition of "Exceptional Point", the "Exceptional Point" are not limited to chairs. The shopping carts and mobile cabinets (cabinet including wheels) also satisfy the three conditions because of their movable features.

2.12. Leverage deployed infrastructures to enhance shadow map

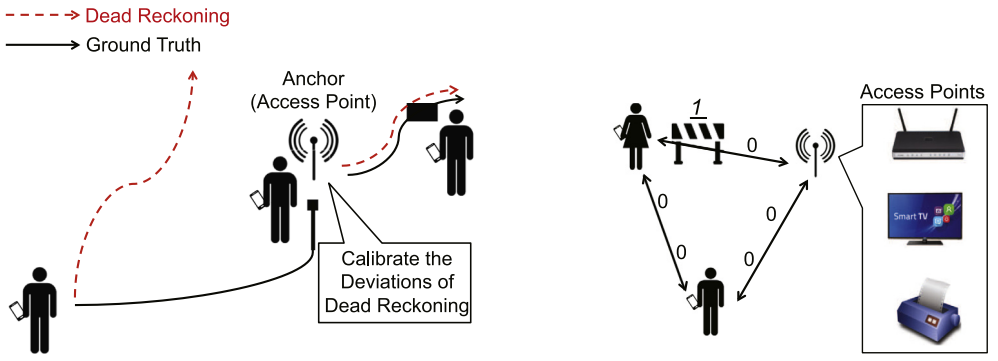
Although `iFrame` does not depend on the pre-installed devices (such as WiFi Access Points and Bluetooth Beacons) in a building, `iFrame` is able to leverage these devices to improve dead reckoning. If the pre-installed wireless devices can share their positions, once a fixed device is close (within one grid) to a user of a smartphone, the user's computed position from dead reckoning will be replaced by the device's accurate position. The distance between the wireless device and the user's smartphone is computed by the prepared RSSI-distance relations. Considering the position of the installed wireless device is known and accurate, the device also can be treated as an Anchor Point in Fig. 12(a).

In addition, even if the pre-installed devices are stationary, they are able to communicate with mobile devices via Bluetooth and WiFi. As presented in Fig. 12(b), the wireless links between the pre-installed devices and mobile devices can describe the states of the grids on the shadow maps. Namely, although `iFrame` is not an infrastructure-based system, it is able to cooperate with the fixed infrastructures that are deployed in indoor environments.

2.13. Energy saving

Most smartphone sensing approaches consume much energy. For example, the smartphones that use a camera sensor to generate street scenarios have to face the challenge of high energy consumption if the camera is always running. However,

Please cite this article in press as: C. Qiu, M.W. Mutka, iFrame: Dynamic indoor map construction through automatic mobile sensing, Pervasive and Mobile Computing (2017), http://dx.doi.org/10.1016/j.pmcj.2016.12.008.



(a) Reduce the error of dead reckoning by using the anchor.

(b) Fixed device communicates with mobile devices.

Fig. 12. Use pre-installed infrastructures to improve the indoor map.

if the camera is turned off, important images may be lost. `iFrame` improves the energy issue by the follows aspects: First, for dead reckoning, considering the sampling frequencies are highly related to energy consumption, when the smartphone detects that the acceleration variation on any one of axises changes more than 2 m/s^2 within 1 min, we assume the smartphone is not stable. Namely, dead reckoning may include more errors and the sampling frequency of the smartphone will be reduced. Once the smartphone detects that the acceleration variation all the axises changes within 2 m/s^2 in 1 min, the sampling frequency will increase. For Bluetooth and WiFi detection, because the proposed "interrupt mechanism" is more energy consuming than default device discovery, if the radio adapters can discovery any nearby device within 3 min, they will adopt the "interrupt mechanism" to detect devices. Otherwise, they will use the default duty cycle to detect devices.

3. Evaluation and discussion

3.1. Experiment setup

We prototype `iFrame` on Samsung Galaxy S5 smartphones and Google Nexus 7 tablets, which support various types of inertial sensors. The version of Android is 4.4. In the experiments and simulations, users leverage Bluetooth adapters, WiFi adapters, and accelerometers. The corresponding sampling frequencies of these components are 0.2–0.06 Hz, 0.25–0.5 Hz, and 5 Hz respectively. We implement dead reckoning to generate a user's motion trajectory: (1) we calculate the movement direction and the motion distance in each segment and (2) merge computed segments in each time period. When the smartphone detects the acceleration variations on all the of axises change less than 2 m/s^2 within 1 min, to save energy, the sampling frequency of accelerometer will adjust to 0.5 Hz. Once the smartphone detects the acceleration variations on any axis change more than 2 m/s^2 within 1 min, the sampling frequency will return 5 Hz. After turning on these sensors, the users carry the mobile devices and walk freely in the indoor scenarios. Initially, we set the shadow value of each grid as 1. For the size of each grid, if the size is too large, the accuracy of the rebuilt map will be constrained. On the contrary, once the size of grid is too small, the complexity of computation will increase sharply. In our evaluation, we set the size of each grid as 0.5 m $\times$ 0.5 m .

For Bluetooth and WiFi detections, the distance–RSSI relations for the smartphones are trained off-line and stored in hash tables. We estimate the layouts between each of the connected smartphones via the proposed algorithms. Empirically derived thresholds $T_b = 3$ dBm, $T_{w1} = 4$ dBm, $T_{w2} = 4$ dBm are proposed in Algorithm 1 and 2. When we combine the three sensing approaches, the initial values of a, b and c are 1/3. We adopt Mean Value pattern to merge the matrices computed from all the mobile devices.

In our evaluation, we seek to answer these questions: (1) Does `iFrame` construct the indoor layout of a building successfully? (2) Can `iFrame` detect the change of layout of a room? (3) Can our matrix fusing mechanism improve the output results? (4) Can an increase in the number of users enhance the accuracy or speed up the rebuilding process? (5) Can `iFrame` cooperate with deployed wireless Access Points in indoor environments? (6) Are anchor points recognized successfully in the generated floor plan?

3.2. Indoor environment measurement

3.2.1. Room measurement

We conducted an experiment in the eLANS Lab of Michigan State University. As shown in Fig. 13, we transform the floor plan of a room into a two dimension map described by a shadow based matrix. The gradient color (from black to white) in each grid represents the shadow state. Three users carry smartphones and walk freely in the room. The smartphones are

Please cite this article in press as: C. Qiu, M.W. Mutka, iFrame: Dynamic indoor map construction through automatic mobile sensing, Pervasive and Mobile Computing (2017), http://dx.doi.org/10.1016/j.pmcj.2016.12.008.

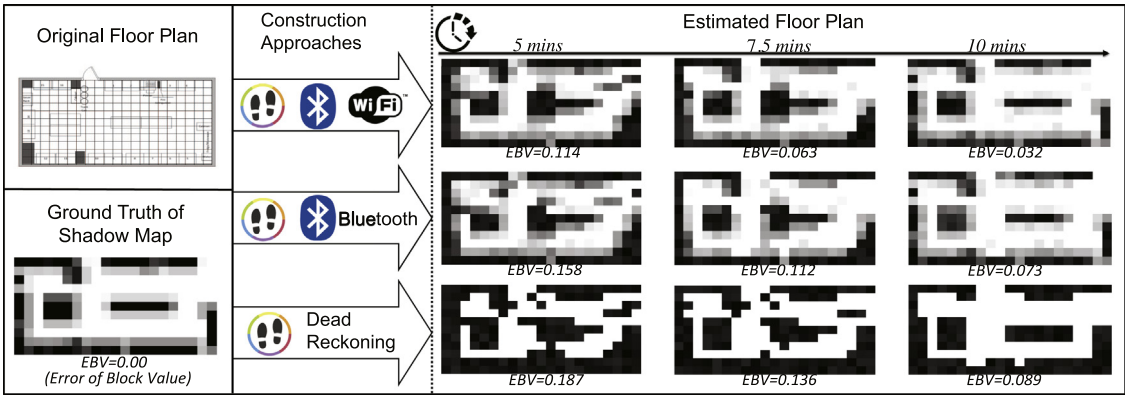


Fig. 13. The case study of indoor floor plan reconstruction. Within the time increase, the generated shadow map is closer to the ground truth. The group including the three methods outperforms other two groups.

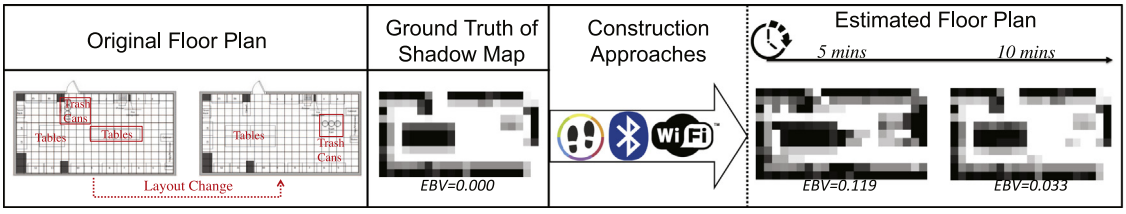


Fig. 14. The case study of rebuilding the changed floor plan. After we changed the positions of trash cans and tables, `iFrame` still can build the accurate indoor shadow map within 5–10 min.

put into the users' pockets. It is easier to detect the objects between wireless links. We only leverage the samples collected in the pockets to build a motion trace. To distinguish the relative position of smartphones, two types of sensors are used: embedded proximity (IR) and light sensors. By using the detection mechanism proposed by Yang [19], we are able to detect the smartphone in pocket or out of pocket.

By executing our approach for 10 min, `iFrame` constructs the indoor floor plan. With the time increasing, the estimated map reflects the ground truth better. For example, the generated shadow map in the 10th minute is more accurate than the shadow map in the 5th minute.

In order to verify our approach is able to represent the latest version of the indoor map, we changed the layout of the room. The proposed algorithm in `iFrame` ran continuously with unattended mode. Then, we conducted the same experiment in the changed scenario. Even if the trash cans and tables have been moved, as Fig. 14, the new generated map is still close to the ground truth.

Note that when people work or live in a room, there is little possibility to walk continuously. Therefore, we did an experiment that more closely resembles people's daily life. There were three users of `iFrame` in a room. They can walk, stop, sit, and chat in the room for 1 h.

In this case, each user adopts Noom Walk [20] and S-Health [21] Pedometer applications on their smartphones, and record the number of steps they have walked. Fig. 8 depicts that even if there exists some differences for the step numbers counted by the two applications, within the increasing of the number of steps, the Errors of Block Values are reduced gradually. Then, we executed a similar experiment within 30 min, the experimental results are close to what we obtained in the 1 hour's group.

To analyze the effectiveness of each technique, we did the following experiments. First, we only used dead reckoning approach to build the map. Then, we added Bluetooth detection and did the experiment. In the end, we combined the three technologies together. As shown in Figs. 13 and 16(a), we conclude each technique can improve the accuracy of the map. By repeating the comparison 10 times, Fig. 16(b) provides the confidence interval for each approach and shows that our conclusion is not coincidental.

There are three participants in the above experiments to collect samples to construct the map. We observe two other control groups for 10 min: (1) One participant has `iFrame`. Since no other person can establish a connection to him/her, only the dead reckoning technique is available; (2) Five participants use `iFrame`, combining the three approaches together. We repeat the experiments 12 times. Fig. 16(c) shows that with an increasing number of users, namely, more samples of matrices computed by `iFrame` can boost the accuracy of map construction. Additionally, the groups with more users cost less time to achieve a low error rate.

Please cite this article in press as: C. Qiu, M.W. Mutka, iFrame: Dynamic indoor map construction through automatic mobile sensing, Pervasive and Mobile Computing (2017), http://dx.doi.org/10.1016/j.pmcj.2016.12.008.

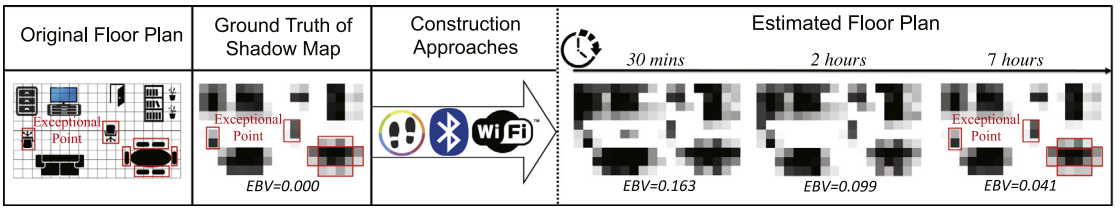
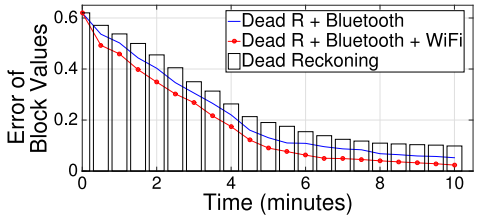
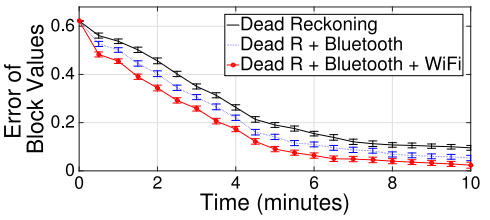


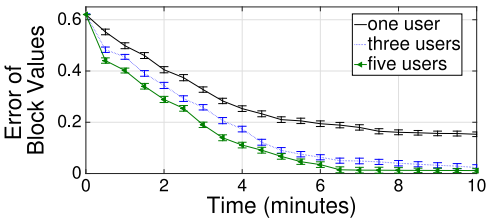
Fig. 15. Floor plan case study for the rooms over a long time.



(a) One time measurement by three approaches.



(b) Repeated measurements by three approaches.



(c) Control group for different number of users.

Fig. 16. Experimental results for single room case study.

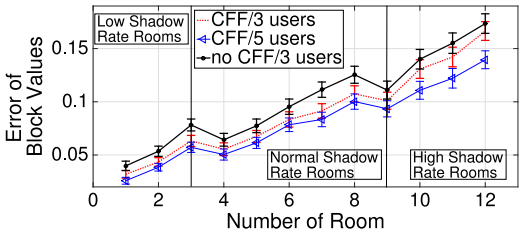


Fig. 17. Results of the extended experiments.

In our measurement, since the users of `iFrame` cover the rooms fully, the indoor maps are generated completely. However, in certain cases, the users may not pass by some areas in a room. These areas (named "blind area") cannot be described by our sensing approaches. To reduce the "blind area" on the map, once `iFrame` cannot receive the data samples from certain areas for 1 h, `iFrame` sends a message to users to suggest to visit the "blind area".

In contrast to letting volunteers walk frequently and cover the full space of the eLANS lab within 10 min, we did an experiment in a living room of apartment (the size is known), which is closer to a real world scenario. Three users of `iFrame` enter and leave the room freely. Our system collected the data from three `iFrame` users in one day (from 10AM to 5PM), as shown in Fig. 15, by applying the "blind area" messages, the shadow map can achieve a low Error of Block Value (0.041).

3.2.2. Building measurement

In Fig. 20, we extend our approach to a building with multiple rooms and hallways. When `iFrame` builds the map of each single room, we also collect a, b, c values (Low/Normal/High shadow rate) for assigning weights to dead reckoning, Bluetooth and WiFi detections. The three groups are shown in Table 1. Once a user enters a room, after the first time period scanning, if the average shadow rate of a room is less than 0.1, it will choose a low shadow rate related to a, b, c values. If the average shadow rate is more than 0.25, it will use a high shadow rate related values. Other cases will adopt the normal shadow rate

Please cite this article in press as: C. Qiu, M.W. Mutka, iFrame: Dynamic indoor map construction through automatic mobile sensing, Pervasive and Mobile Computing (2017), http://dx.doi.org/10.1016/j.pmcj.2016.12.008.

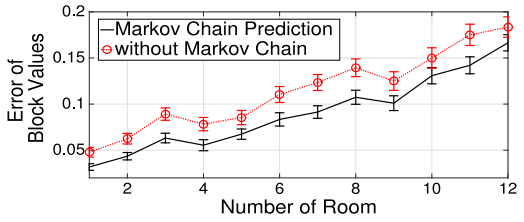


Fig. 18. Markov chain prediction for multiple rooms.

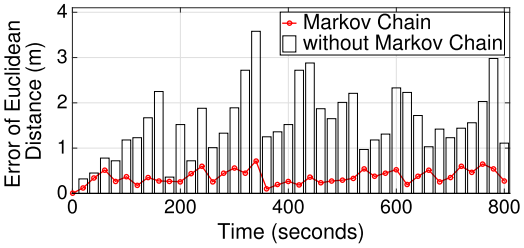


Fig. 19. Markov chain prediction in one room.

Table 1
Different groups of a, b, and c values.

	a	b	c
Low shadow rate	0.430	0.353	0.217
Normal shadow rate	0.412	0.324	0.264
High shadow rate	0.345	0.225	0.430

related values. As shown in Fig. 17, although all of these experimental scenarios stay at a low error levels, the rooms with lower shadow rates have less errors.

`iFrame` obtains improved results from combining the three types of output matrices. The selection of a, b, c is based on CFF. To verify the effectiveness of CFF, we did the following comparisons: under the same condition as the previous evaluation, we set the values of a, b and c as 1/3, in Table 3, rather than selecting the values of a, b and c by the proposed method. Fig. 17 shows the Errors of Block Values of 12 places without changing the initial values of a, b, c and the control group using CFF. We can make two interesting observations from the comparisons. First, the evaluation results confirm CFF enhances the accuracy of map construction effectively. Another observation is that the group with five users has better performance than the group having three users.

Next, we study the benefit of the Markov chain and Average Speed predictions, which aim to correct the deviations from dead reckoning. We keep the above experiment conditions, but delete the Markov chain prediction mechanism and conduct `iFrame` 10 times in 12 rooms. As represented in Fig. 18, we observe that the group with the Markov chain prediction outperforms the other one. Then, we only concentrate on room 1 and run `iFrame` for 800 s. Two groups of results are illustrated in Fig. 19. With the time increase, the group with Markov Chain prediction has less error of deviation distance. Error of Euclidean Distance is the distance (in meters) between the ground truth and the estimated position. Besides, we concentrate on how ASM improves the dead reckoning. We curve fit the historic average speeds and predict the average speed in the current time period. Each time period is 30 s. As in Fig. 21, we set 5, 2, 0.5, and 0.1 m/s as the error ranges of predicted average speeds and compare these control groups. We conclude (1) AAS improves the accuracies of dead reckoning and (2) 2 m/s is the most suitable error range of AAS.

We select the Mean Value mechanism to merging sensing data from different users. When we use Maximum Space and Minimum Space organizations to collecting data, the average Errors of Block Values in the building are reduced by 0.012 and 0.009.

The Anchor Points (e.g., entrances, stairs, and elevators) were identified in our system as shown in Fig. 20. Table 2 lists the recognition results for these Anchor Points. Only one stair and one entrance on our map were mis-classified.

The above evaluations do not contain fixed wireless Access Points (AP), such as wireless routers and bluetooth beacons. To validate `iFrame` is able to work with fixed access points, we add Access Points in the experimental scenario as in Fig. 22. In the first group, each room does not include any AP. In the second group, there is one fixed AP in each room. There are 2 AP in each room for the third group. As presented in Fig. 22, we show (1) the accuracies of reconstructed map can be improved by adding fixed AP and (2) the accuracies will increase by using more APs.

Please cite this article in press as: C. Qiu, M.W. Mutka, iFrame: Dynamic indoor map construction through automatic mobile sensing, Pervasive and Mobile Computing (2017), http://dx.doi.org/10.1016/j.pmcj.2016.12.008.

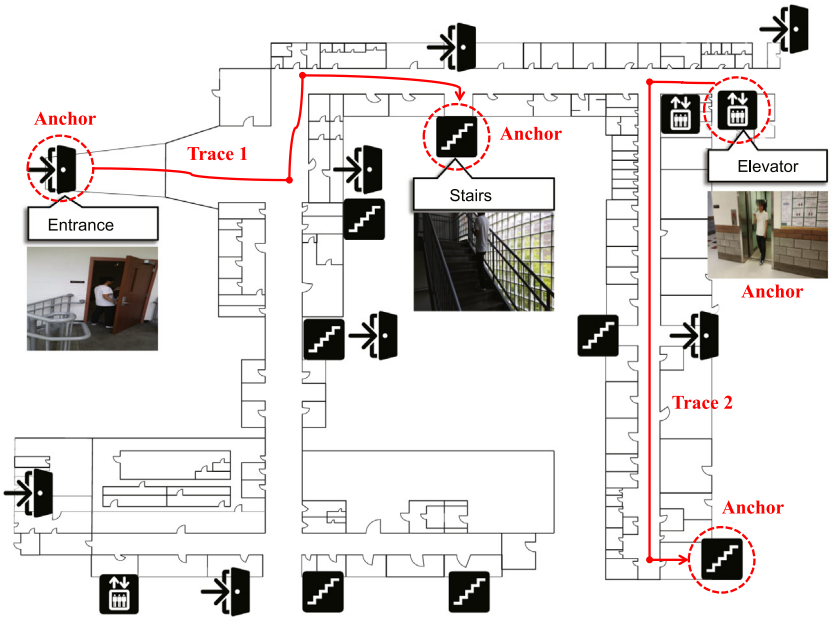


Fig. 20. Extend our evaluation from single room to the whole building via recognizing the anchor points.

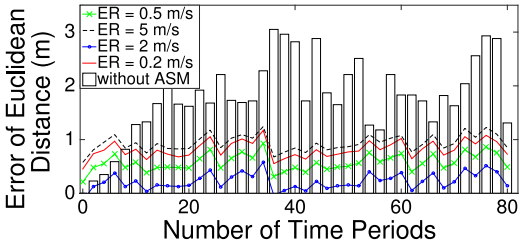


Fig. 21. ASM comparison with different thresholds.

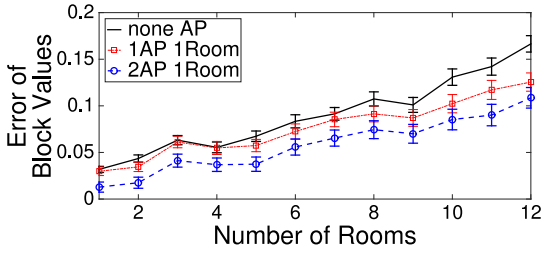


Fig. 22. Experimental results of using fixed AP.

Table 2
Confusion matrix for classifying different anchor positions.

	Elevator	Stairs	Entrance	Other	FP	FN
Elevator	9	0	0	0	0%	0%
Stairs	0	20	0	1	0%	4.8%
Entrance	0	0	20	1	0%	4.8%
Other	0	0	0	40	5%	0%

We focus on the recognition of Exceptional Points. Within 10 s, if the variations of accelerations on each axis are within certain thresholds, the state is defined as “stable state”. Based upon the above experimental scenario, we deploy 20 chairs

Please cite this article in press as: C. Qiu, M.W. Mutka, iFrame: Dynamic indoor map construction through automatic mobile sensing, Pervasive and Mobile Computing (2017), http://dx.doi.org/10.1016/j.pmcj.2016.12.008.

Table 3
Different groups of a, b, and c values.

	a	b	c
Low shadow rate	0.430	0.353	0.217
Normal shadow rate	0.412	0.324	0.264
High shadow rate	0.345	0.225	0.430

Table 4
Control groups of using different number of detected devices.

EBV	2 min	4 min	6 min	10 min
One device	0.409	0.233	0.126	0.031
Three devices	0.356	0.185	0.090	0.015
Five devices	0.365	0.189	0.093	0.017
Finished	0.376	0.192	0.096	0.019

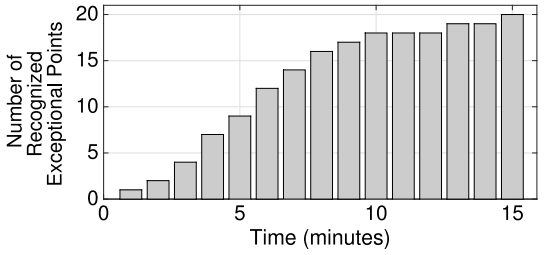


Fig. 23. Number of recognized exceptional points.

as the Exceptional Points. Once a user of `iFrame` passes by a chair, if the "stable state" can be detected by our system, the chair will be distinguished as an Exceptional Point. We tested different threshold of defining the "stable state" (0.1, 1, and 10 m/s^2). When we choose 1 m/s^2 as the variation range of accelerations on each axis, the results are more accurate. Fig. 23 illustrates that three users of `iFrame` walk freely and ran our approach within 15 min. All the Exceptional Points on the map are recognized gradually. Although there is only one false positive case, it is the empty grid that is close to a chair that is distinguished as a real chair.

In the end, we discuss the real-time features of existing approaches. For the above experiments, the Bluetooth and WiFi adapters resume when the whole discovery round finishes. By keeping the same experimental conditions, we modified the original detection models by interrupting the duty cycle of device discovering, once the program obtains a certain number of devices, the serving round of device discovery will resume immediately. As shown in Table 4, we compare the control groups with different number of detected devices in a partial list. When we set three as the number of devices in the detection list, we obtain the best results.

4. Related work

SLAM: Many researchers have designed systems to rebuild indoor maps. SLAM (Simultaneous Localization And Mapping) [22,23] is a classical problem in the robotics area, which reconstruct maps in an unexplored environment. By using different sensors, such as gyroscopes, cameras, and laser sensors, the robots or other mobile devices can detect the layout of indoor environment. For example, Google Tango combines 3D motion tracking with depth sensing to give your mobile device the ability to know where it is and how it moves through space [24]. However, SLAM requires strong computational power to sense a sizable area and process the complex collected data.

Smartphone approaches: Since people use smartphones and other mobile devices in their daily lives, some approaches propose to build indoor maps by smartphones. In the computer vision area, researchers activate camera sensors on smartphones to construct the map [25,3,26]. JigSaw [3] leverages camera sensors to obtain images and adopts point cloud algorithms to piece together images to reconstruct the indoor floor plan. Sextant [26] combines the photos obtained by smartphones with gyroscope information to draw the map. Though these vision based approaches can build the indoor map by smartphones or other specific devices, the procedure of image processing is still challenging. Some work uses the dead reckoning approach by accessing data from accelerometers and gyroscopes to generate the trajectories of users [4,5]. They analyze the trajectories and sensing data of elevators and entrances to build the map. Nevertheless, the deviations and errors of dead reckoning often reduce the accuracies. Other approaches use WiFi signatures and a series of algorithms to discover the layout of buildings, but the WiFi noise is difficult to process [27,28].

Apart from SLAM or other smartphone-based approaches, `iFrame` does not require to process complex sensing and training data. `iFrame` users do not carry any extra devices except for smartphones. By combining RSSI analysis and dead

Please cite this article in press as: C. Qiu, M.W. Mutka, iFrame: Dynamic indoor map construction through automatic mobile sensing, Pervasive and Mobile Computing (2017), http://dx.doi.org/10.1016/j.pmcj.2016.12.008.

reckoning of complementary strengths, iFrame constructs indoor maps and detects the changes of the indoor layouts automatically.

5. Conclusion and future work

We introduce an unattended and high-speed indoor map construction approach called iFrame. After abstracting the unexplored map as a matrix, and by combining dead reckoning and RSSI detection techniques, iFrame judges whether the subareas in an indoor environment are empty or not. Each of proposed technologies compensates the shortcomings of others by adopting a matrix fusing mechanism. Our approach selects proper parameters for merging data automatically and yields a clear shadow map for each room within 5–10 min. By acceleration analysis and limited data training, we implemented iFrame in a real indoor building. iFrame reconstructs the layout of different rooms, hallways, and some special positions (as elevators, chairs, stairs, and entrances) successfully. Although we have achieved the goal of floor plan reconstruction on a two dimensional level, we hope iFrame represent the heights of different objects to the future. Therefore, we plan to employ a three dimensional array to describe the environment. One dimension will record the height of the environment.

Acknowledgments

The authors would like to thank Professor Dr. Max Mühlhäuser for his helpful comments. This work is supported in part by National Science Foundation Grant No. CNS-1320561.

References

[1] A. Küpper, Location-Based Services: Fundamentals and Operation, John Wiley & Sons, 2005.
[2] Google Indoor Maps, available: http://www.google.com/maps.
[3] R. Gao, M. Zhao, T. Ye, F. Ye, Y. Wang, K. Bian, T. Wang, X. Li, Jigsaw: Indoor floor plan reconstruction via mobile crowdsensing, in: MobiCom, ACM, 2014, pp. 249–260.
[4] M. Alzantot, M. Youssef, Crowdinside: automatic construction of indoor floorplans, in: SIGSPATIAL, ACM, 2012, pp. 99–108.
[5] D. Philipp, P. Baier, C. Dibak, F. Durr, K. Rothermel, S. Becker, M. Peter, D. Fritsch, Mapgenie: Grammar-enhanced indoor map construction from crowd-sourced data, in: PerCom, IEEE, 2014, pp. 139–147.
[6] U. Steinhoff, B. Schiele, Dead reckoning from the pocket-an experimental study, in: PerCom, IEEE, 2010, pp. 162–170.
[7] C. Qiu, M.W. Mutka, iFrame: Dynamic indoor map construction through automatic mobile sensing, in: PerCom, IEEE, 2016.
[8] W.-S. Soh, et al., A comprehensive study of bluetooth signal parameters for localization, in: PIMRC, IEEE, 2007, pp. 1–5.
[9] S. Hilsenbeck, D. Bobkov, G. Schroth, R. Huitl, E. Steinbach, Graph-based data fusion of pedometer and wifi measurements for mobile indoor positioning, in: UbiComp, ACM, 2014, pp. 147–158.
[10] J. Chon, H. Cha, Lifemap: A smartphone-based context provider for location-based services, IEEE Pervasive Comput. (2) (2011) 58–67.
[11] W. Kang, Y. Han, SmartPDR: Smartphone-based pedestrian dead reckoning for indoor localization, IEEE Sens. J. 15 (5) (2015) 2906–2916.
[12] UM6 sensor, available: https://www.chrobotics.com.
[13] J. Seitz, T. Vaupel, J. Jahn, S. Meyer, J.G. Boronat, J. Thielecke, A hidden markov model for urban navigation based on fingerprinting and pedestrian dead reckoning, in: FUSION, IEEE, 2010, pp. 1–8.
[14] C.W. Gardiner, et al., Handbook of Stochastic Methods, Vol. 4, Springer, Berlin, 1985.
[15] M.M. Samson, I. Meeuwsen, A. Crowe, J. Dessens, S.A. Duursma, H. Verhaar, Relationships between physical performance measures, age, height and body weight in healthy adults, Age Ageing 29 (3) (2000) 235–242.
[16] K. Chintalapudi, A. Padmanabha Iyer, V.N. Padmanabhan, Indoor localization without the pain, in: MobiCom, ACM, 2010, pp. 173–184.
[17] P.V. Viswanadham, Motion detection with bluetooth low energy scan, Google Patents, US Patent 9,179,254, 2015.
[18] F. Adib, Z. Kabelac, D. Katabi, Multi-person localization via RF body reflections, in: NSDI, USENIX Association, 2015, pp. 279–292.
[19] J. Yang, E. Munguia-Tapia, S. Gibbs, Efficient in-pocket detection with mobile phones, in: UbiComp, ACM, 2013, pp. 31–34.
[20] Noom Walk, available: https://www.noom.com.
[21] S Health, available: http://shealth.samsung.com.
[22] M. Dissanayake, P. Newman, S. Clark, H.F. Durrant-Whyte, M. Csorba, A solution to the simultaneous localization and map building (SLAM) problem, IEEE Trans. Robot. Autom. 17 (3) (2001) 229–241.
[23] B. Suger, G. Diego Tipaldi, L. Spinello, W. Burgard, An approach to solving large-scale slam problems with a small memory footprint, in: ICRA, IEEE, 2014, pp. 3632–3637.
[24] Google Tango, available: http://www.google.com/atap/project-tango.
[25] T. Sattler, B. Leibe, L. Kobbelt, Fast image-based localization using direct 2D-to-3D matching, in: ICCV, IEEE, 2011, pp. 667–674.
[26] Yang, e.a. Tian, Towards ubiquitous indoor localization service leveraging environmental physical features, in: INFOCOM, IEEE, 2014, pp. 55–63.
[27] Y. Jiang, Y. Xiang, X. Pan, K. Li, Q. Lv, R.P. Dick, L. Shang, M. Hannigan, Hallway based automatic indoor floorplan construction using room fingerprints, in: UbiComp, ACM, 2013, pp. 315–324.
[28] G. Shen, Z. Chen, P. Zhang, T. Moscibroda, Y. Zhang, Walkie-markie: Indoor pathway mapping made easy, in: NSDI, USENIX Association, 2013, pp. 85–98.

Please cite this article in press as: C. Qiu, M.W. Mutka, iFrame: Dynamic indoor map construction through automatic mobile sensing, Pervasive and Mobile Computing (2017), http://dx.doi.org/10.1016/j.pmcj.2016.12.008.