SBC: Scalable Smartphone Barometer Calibration through Crowdsourcing

Haibo Ye[1], Tao Gu[2], Xianping Tao[1], Jian Lu[1]
[1]State Key Laboratory for Novel Software Technology, Nanjing University, China
[2]School of Computer Science and IT, RMIT University, Australia
yhb@smail.nju.edu.cn, tao.gu@rmit.edu.au, {txp, lj}@nju.edu.cn

ABSTRACT

We have seen increasingly popularity in embedding barometer into smartphone today. A barometer measures the barometric pressure, and it can be used for a variety of applications. For example, in localization techniques, it is used to detect the altitude or altitude change of a user. Unfortunately, the smartphone barometer measurement is not accurate, and it has to calibrate appropriately before use. In this paper, we present Scalable Barometer Calibration (SBC), a scalable, transitive calibration algorithm to automatically calibrate barometer for a large number of smartphone users. SBC requires neither any infrastructure nor any human intervention, it uses smartphone barometer and accelerometer only. SBC provides high accuracy of barometer calibration and minimum energy consumption, making it more realistic for real-world deployment. Our simulation and prototype system demonstrate the performance, scalability, and robustness of SBC.

Keywords

Smartphone Barometer, Calibration, Crowdsourcing.

1. INTRODUCTION

The advancement of embedded sensors in smartphones has motivated numerous sensor-assisted applications [11,19]. The barometer (a built-in sensor which measures atmospheric pressure and first appears in the Google Galaxy Nexus) now becomes more and more common in smartphones. Knowing the barometric pressure around mobile users is particularly useful for a variety of applications. For example, in weather forecasting, knowing a super-dense picture of barometric pressure from users in that area will provide timely and accurate weather variation forecast. In shopping mall or airport environments, a navigation service such as Google maps [3] can detect a user's current floor level by transforming the barometric pressure to altitude. Many outdoor applications [1, 5] make use of barometer to calculate the altitude or vertical displacement of users. Recent studies use smartphone barometer for indoor localization [16, 17]. These applications perform well only if the barometer measurement is accurate. However, in real-life scenarios, smartphone barometer is often not accurate if it is not calibrated carefully and constantly. Smartphone barometers need to be calibrated before they are used for building these applications. We name it the *barometer calibration* problem, which aims to calibrate barometers for different smartphone users.

Barometer on smartphone often has a drift (a.k.a. creep) between the reading and the real barometric pressure, and the drift varies from device to device. For calibration, it is important to get the drift of all smartphones. Unfortunately, in reality, it cannot be easily done. The barometric pressure varies by time and location [2, 18]. The barometric pressure at the same location may change in less then 10 minutes depending on the weather condition. The barometric pressure in different locations are different, even in the two adjacent rooms it maybe different because of different temperature and humidity in each room. In order to calibrate a smartphone barometer, we need to know the real barometric pressure at the smartphone's location, which may change over time. For a small number of smartphone barometers, they can be easily calibrated at a meteorological center where the barometric pressure can be measured accurately. However, it is a real challenge to calibrate barometers for thousands and millions of smartphone users in a scalable manner.

An intuitive solution is to use weather reports for smartphone barometer calibration. We can easily compare the barometer pressure values from a smartphone barometer with the one in a weather report, and obtain the drift by subtracting the two values, assuming that the value from the weather report is accurate. Unfortunately, it may not work in reality. In a weather report, the atmospheric pressure is typically measured at sea level, which does not match to smartphones' locations. Smartphone users may move around in different locations with different heights from the sea level. The barometric pressure decreases by 0.12 hPa for going up every 1 meter in the vertical direction. One may think of using a high-precision pressure gauge meter to calibrate smartphone barometers. To do this, we simply put these two devices together, and the drift can be easily obtained by comparing the two readings. But, in reality, it is difficult, if not both time-consuming and labor-intensive, to implement this method considering a large number of smartphones. In addition, barometer drift may change over time due to changes in the aneroid cell occurring slowly [7], and even the

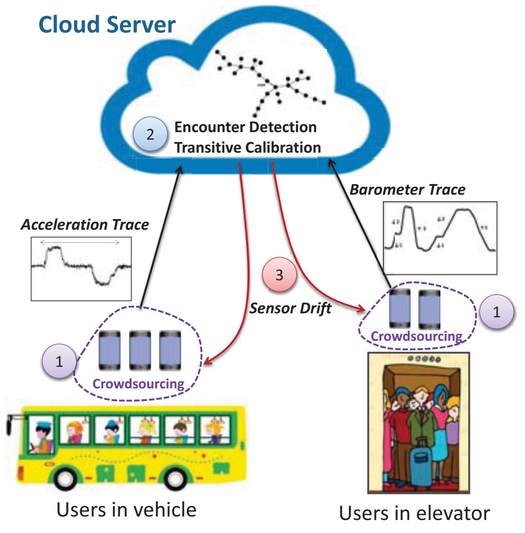


Figure 1: Overview of SBC

barometric pressure at the same location may keep changing in a day due to different weather conditions [2]. Hence, we may have to carry out calibration periodically, which it is obviously not scalable.

In this paper, we present a novel Scalable Barometer Calibration (SBC) approach to address the challenges of calibrating a large number of smartphone barometers. For salability, we propose a transitive calibration scheme through crowdsourcing. We observe that users often encounter with each other at places such as in elevators and on the public transport, the two most common places. In this case, their barometer measurements must be the same. For example, user A encounters with user B in an elevator. If user A's barometer is calibrated, we can calibrate user B's barometer using user A's reading. Later, when user B encounters with user C in a bus, and user C's barometer can be calibrated in the same way. Based on this observation, we design a scalable, transitive calibration algorithm to automatically calibrate different smartphones' barometers. We collect user data through crowdsourcing, calibrate smartphone barometers by user encounters and the calibration is done in a transitive way with more users involved. In more details, we first detect user encounters in elevators and vehicles and collect the data using crowdsourcing. The detection is done through both barometer and accelerometer signatures. We then calibrate groups of users when they encounter with each other, and apply a graph based algorithm to calibrate all smartphones in a transitive way.

In summary, we make the following contributions:

1. We propose an efficient and scalable approach for smartphone barometer calibration. SBC makes use of barometer and accelerometer on smartphone only, and does not require any infrastructure support.

2. We design several novel techniques to detect users when they appear in the same elevators and vehicles, and calibrate barometers for different smartphones in a transitive way.

3. We conduct both extensive simulations and field studies to analyze the performance of SBC. We deploy SBC in a real situation to demonstrate its superiority over existing solutions. Our evaluation shows that 100 of users in a 10-floor building can all be calibrated in a single day.

The rest of this paper is organized as follows. Section 2 gives an overview, followed by the detailed design. Section 3 gives the theoretical analysis. Our evaluation is reported in Section 4. Section 5 discusses the related work, and finally, section 6 concludes the paper.

2. SYSTEM DESIGN

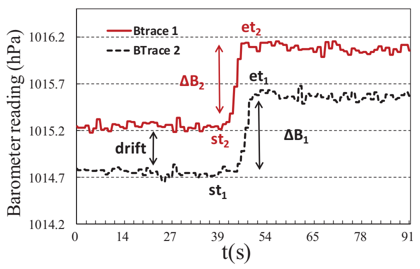
We give an overview of SBC in this section, as shown in Fig. 1. The system operates in three phases. In the first phase, SBC collects data from user smartphones through crowdsourcing. When a user travels up and down in the building (e.g., taking elevators), the mobile client software running on the phone collects barometer readings in real-time. The activities of taking elevators are detected and captured by our activity recognition algorithm. We design a robust technique to recognize such activities using barometer on smartphone. When a user is in the vehicle, the mobile client software collects the accelerometer trace in real-time. The recognized elevator activities and the accelerometer trace will be uploaded to the cloud server as a user trace. In the second phase, in the cloud server, we calibrate barometers on different smartphones based on user encounter in elevators and vehicles. The user encounter in elevators are detected based on user elevator activities. The user encounter in vehicles will be detected using our DTW matching algorithm which makes use of the accelerometer readings. After encounter detection, the calibration is done in a transitive way with more users involved, and eventually propagated to all possible users in a scalable way. In the last phase, a mobile user first downloads the barometer drift of his smartphone. When measuring the barometer pressure, the value is the sum of the row barometer reading and the drift.

Table 1: Barometer sensor parameters

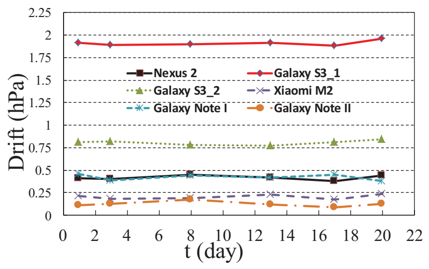
Property	BMP180/182	LPS331AP
Absolute accuracy (300-1100 hPa)(0-65°C)	-4.0 ... +2.0 hPa (-33...+17m)	- 3.2...+2.6hPa (-27...+22m)
Relative accuracy (950-1050 hPa)(@25°C)	$\pm$ 0.12 hPa ($\pm$ 1m)	$\pm$ 0.2 hPa ($\pm$ 1.7m)
Noise	0.06 hPa (0.5m)	0.06hPa (0.5m)
Used in smartphone	Galaxy Note 2/3, Xaiomi M2, Sony Ericsson Active, Nexus 3/4	Galaxy S3,S4

2.1 Barometer on Smartphone

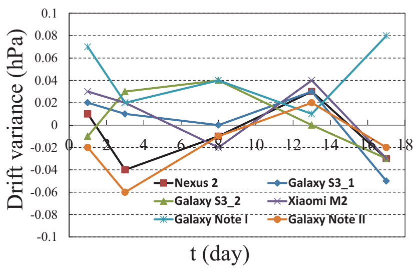
Barometric pressure is the force per unit area exerted on a surface by the weight of air above that surface in the atmosphere of Earth [2]. As altitude increases, barometric pressure decreases. Barometer sensor on smartphones can measure the barometric pressure. Barometer sensor has become increasedly popular on smartphones today. Most commonly used barometer sensors are BMP180/182 and LPS331AP. Table 1 gives their technical specifications. From the table,



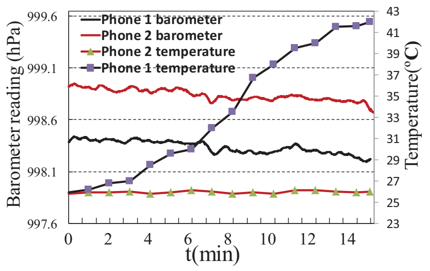
(a) The barometer reading drift between smartphones



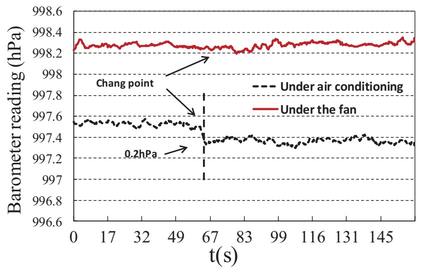
(b) The drift in different smartphones



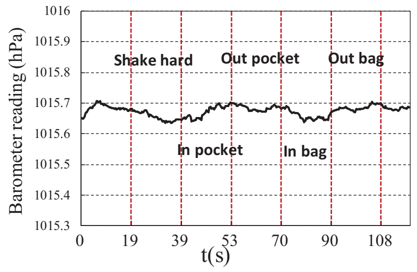
(c) The drift variation in different smartphones



(d) The relation between smartphone temperature and barometer reading



(e) The relation between wind and barometer reading



(f) The barometer readings in different usage scenarios

Figure 2: The properties of smartphone's barometer

we observe that while the absolute accuracy[1] is about ± 20 meters (which is low), the relative accuracy[2] is about ± 1.5 meters (which is high) and the noise is only about 0.5m. This implies that the barometer sensor has a high level of sensitivity, and it is good enough to detect the change of the barometric pressure when users go up or down in a building. There is a clear barometric pressure change of about 0.45hPa for each floor. Motivated by this observation, we use barometer to detect the activities when users change their floor levels by elevators.

We sampled the barometer readings of two smartphones of the same type at the same indoor location. Although the relative accuracy is high in the datasheet, but in reality, they are not accurate. There are considerable drifts between the barometer readings and the real barometric pressure. Figure 2(a) shows that there is a drift of sensor readings of two smartphones which are put together. The drift may result in an error ranging up to 1.8 hPa (15 meters). In another study, we compare the barometer readings from different smartphones with real barometric pressure, we are interested to know if the drift changes in a short time. We used 6 different smartphones, and recorded the drift to the real barometric pressure every 4 days. The results show that the sensor drift in Fig. 2(b) keeps stable and the variation in Fig. 2(c) is negligible. This means that the drift can be stable for several days(20 days).

We then conducted more experiments to further study barometer sensor properties under different usage scenarios—1) the smartphone gets hot, 2) the smartphone is under the wind, 3) the smartphone shakes, and 4) the smartphone is in pocket or bag. Figure 2(d) shows the barometer readings of two smartphones at the same location for 15 minutes—one with a constant temperature, the other with a growing temperature (e.g., when continuously running a computation intensive application). The result shows that the temperature does not affect the barometer reading. Figure 2(e) shows the barometer readings of the two smartphones with and without the wind (e.g., a fan and the air conditioner are used to generate the wind). The result shows that the barometer readings keep unchanged under the fan, and vary in a small range of 0.2hPa under air conditioning, it shows that the wind and temperature changes have limited impact on the reading. Figure 2(f) shows the barometer readings when shaking the phone, putting in and taking out from the pocket or bag. The result shows the barometer readings remain constant under these scenarios.

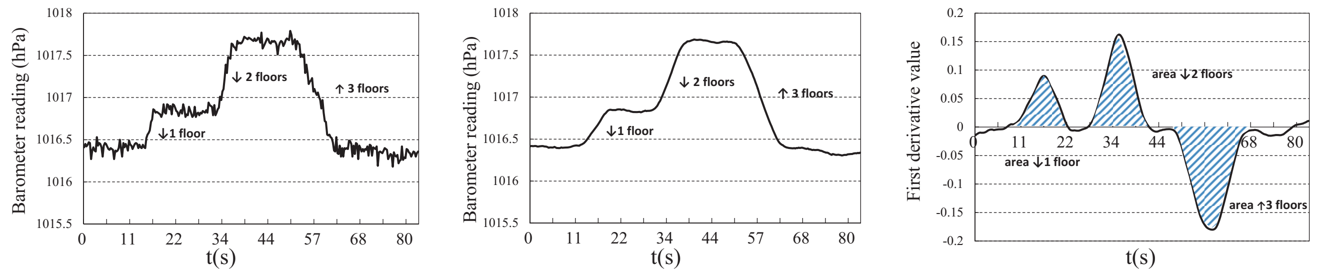
It is clear that appropriate calibration needs to be done, and it shows that if we carefully calibrate the barometer by getting the drift between the barometer reading and the real barometric pressure, the barometer can be very accurate.

2.2 Transitive Calibration Algorithm

The objective of barometer calibration is to calibrate each smartphone's barometer to the real barometric pressure. Our approach is to choose an accurate smartphone's barometer as a reference point and find the drift between each of other user's barometer and the reference. Then all calibrated barometer readings will be accurate. Before we introduce our calibration algorithm, we first introduce the property of the drift between sensors. We define $drift_{AB}$ as the drift between barometer A and B, and $drift_{AB} = Baro_A - Baro_B$, where $Baro_A$ and $Baro_B$ is the reading from barometer A and B, respectively, under the same barometric pres-

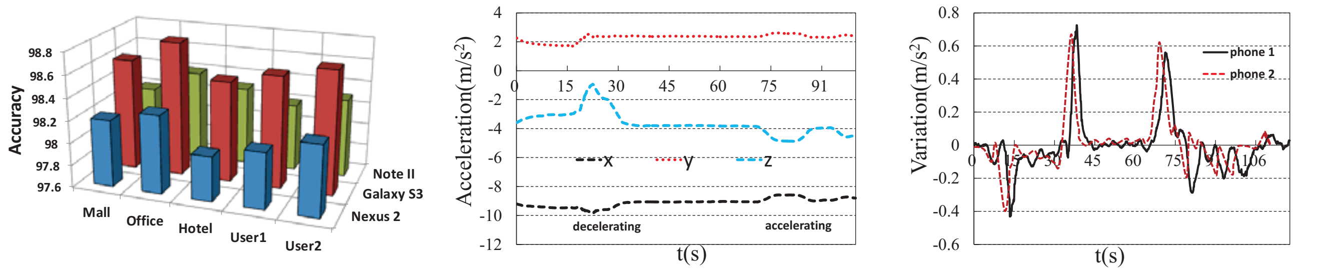
[1]The difference of the change of a sensor reading compares to the change of real barometric pressure.

[2]The accuracy of a sensor reading compares to the real barometric pressure.



(a) Raw barometer readings of taking elevator (b) Smoothed barometer readings of taking elevator (c) First derivative value of the readings

Figure 3: Floor-change activity detection by barometer readings



(a) Floor-change detection accuracy (b) The acceleration readings when the phone is randomly placed (c) The combination of the accelerometer readings variation in three axis

Figure 4: The acceleration readings signature

sure. The barometer drift holds the following properties: 1) $drift_{AB} = -drift_{BA}$, 2) $drift_{AB} + drift_{BC} = drift_{AC}$. These properties clearly demonstrate the transitive relationship. We define barometer calibration as follows: 1) For two smartphones, they are calibrated when the drift of the two barometers is known by the cloud server. 2) For more than two smartphones, they are calibrated when the drift between every two barometers is directly known by encounter in elevator/vechile or indirectly known by the transitive relationship. In order to calibrate all the barometers, we first propose two ways of calibrating two barometers. One is based on user encounters in elevators in an indoor environment, the other is based on user encounter in vehicles in an outdoor environment. We know that a user can not encounter with all other users in elevators or vehicles, in our approach, we then calibrate all users based on the transitive property of calibration. It is important to notice that, we only detect user encounters in elevators and vehicles because it is energy efficient and accurate, making our approach more practical and scalable for widely usage. Other encounter detection techniques based on bluetooth [15] and sound [20] are not practical because they cost more energy and have more restrictions.

2.3 Calibration by User Encounter in Elevators

Calibration is done by analyzing barometer traces. The idea is to calibrate users' barometers when they encounter each other. Users encounter often in elevators since elevator is very common in buildings and users often take elevators. So we first detect the activity of user taking elevators, than find out who encountered each other in the elevator.

2.3.1 *Elevator Activity Detection*

We first present a novel technique to recognize the activities of taking elevators using barometer. We represent a barometer sample P by $B = \{t, Baro\}$, where t is the time for sampling, and $Baro$ is the barometer reading at time t. The barometer samples arriving in time order form a barometer trace, which is represented by $BTrace = \{ID, B_1, B_2, ..\}$, where ID is the identity of the user. Users typically change their floor levels by taking elevators. The barometer sensor is inherent noisy. Figure 3(a) shows the raw barometer readings. In SBC, we smooth the values with a reasonable window size of 1000 ms (i.e., the value at time t is the average value from $t - 500$ to $t + 500$ ms), as shown in Fig. 3(b). In our previous study, we observe that barometer readings on smartphones don't change much in a short period of time unless users change their floor levels. Hence, the change of barometer readings can be used to recognize the floor-change activities. To do this, we extract the first derivative of the barometer readings and the resulting curve is shown in Fig. 3(c). We can see from the figure that the change of barometer readings is transformed to crest when going up and trough when going down. The crest and trough are sharp when taking elevators and smooth when taking escalators and stairs. This let us distinguish taking elevators from taking escalators and stairs. The start and end time of the activity is the time of the left and right edge of each crest or trough.

To detect the taking elevator activities, we calculate the area size of each crest or trough. If it meets certain conditions, a taking elevator activity is detected. In detail, each area is defined as a continuous and closed region formed by the x axis and the curve. The region is located below or upon the x axis, which should meet the following conditions: 1) Lasted time between 3 and 120 seconds, 2) Area size bigger than 1.0. Figure 3(c) shows the areas of taking elevators. In SBC, we do not impose any constraint on the ways users carry or use their smartphones. A smartphone can be held on hand, placed into a pocket or bag, or used to make/receive a phone call, etc. We define an taking elevator activity as $A = \{STime, ETime, SBaro, EBaro\}$, where $STime$ is the start time of an activity, $ETime$ is the stop time of an activity, $SBaro$ and $EBaro$ is the barometer reading at $STime$ and $ETime$, respectively. The user's moving trace can be then defined as $MTrace = \langle ID, A_1, A_2, \ldots \rangle$, where ID is the identity of the user. The detection is done in the smartphone and the resulting $MTrace$ will be uploaded to the cloud server. We conducted experiments with two users using three different smartphones under real-life situations in three different buildings. Figure 4(a) shows the accuracy. The results show the average accuracy using barometer is about 98.3%.

2.3.2 Find encounter in elevators

We observe that if users encounter each other in an elevator, the time and value of their barometer change are the same. In an other word, if we detect two taking elevator activities from both users' barometer traces, these activities start and end at the same time, and the barometer readings change is the same, we conclude that the two users encounter in the same elevator, Fig. 2(a) is an example. This is formalized as follows. *1)* I_1: $A_i.STime = A_j.STime$; *2)* I_2: $A_i.ETime = A_j.ETime$; *3)* I_3: $A_i.SBaro - A_i.EBaro = A_j.SBaro - A_j.EBaro$; and *4)* I_4: $A_i = A_j$; where A_i and A_j is the floor-change activity for user i and j, respectively. The rule is then formulated as follows.

$$R_1 : I_1 \wedge I_2 \wedge I_3 \rightarrow I_4.$$

If we have $A_i = A_j$, the drift is then calculated by the following formula.

$$drift_{ij} = \frac{\sum_{t=A_i.STime}^{A_i.ETime}(B_i(t) - B_j(t))}{n} \tag{1}$$

where $B_i(t)$ and $B_j(t)$ is the barometer reading of user i and j, respectively, at time t, n is the total sample size.

We analyze a case that when two users enter into different elevators at different floors and experience a floor-change activity with the same barometer change at the same time. The above rules will wrongly conclude they are in the same elevator. To handle this case, we first observe that when users encounter in an elevator, they often experience more than one floor-change activity together. For example, user i and j encounter each other at the ground floor and go up to the 8th and 10th floor, respectively. Before arriving at level 8, the elevator stops at levels 3 and 5. In this scenario, user i and j experience 3 floor-change activities (i.e., from 1 to 3, 3 to 5 and 5 to 8). Based on this observation, we conclude users encounter in elevator when there are $n (n \geq 2)$ consecutive floor-change activities between them, where n is called the confidence of the encounter. A big n value will minimize the probability of the fault case. We formalize it as follows.

1) I_5: $\exists A_1, A_2, .., A_k \in MTrace_i$; *2)* I_6: $\exists A_1, A_2, .., A_k \in MTrace_j$; *3)* I_7: $A_{m+1}.STime - A_m.ETime < 30$ holds in $MTrace_i$ and $MTrace_j$; *4)* I_8: $MTrace_i.A_m = MTrace_j.A_m$; *5)* I_9: user i and j are in the same elevator and the confidence is k. The rule is then formulated as follows.

$$R_2 : I_5 \wedge I_6 \wedge I_7 \wedge I_8 \rightarrow I_9.$$

where $MTrace_i$ and $MTrace_j$ is the trace of user i and j, respectively, and k is the confidence that user i and j is in the same elevator.

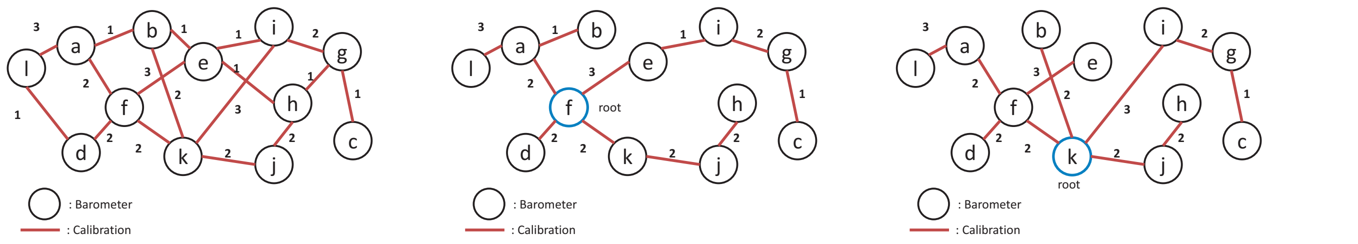
2.4 Calibration by User Encounter in Vehicles

We represent a accelerometer sample C by $C = \{t, x, y, z\}$, where t is the time for sampling, and x, y, z are the accelerometer reading at the three axes of the smartphone. The accelerometer samples arriving in time order form a accelerometer trace $CTrace = \{ID, C_1, C_2, ..\}$, where ID is the identity of the user. Assume two users are in the same vehicle, their smartphone accelerometers will have the same readings when the vehicle is accelerating or decelerating. We use this observation to detect users encounter in the same vehicle.

2.4.1 Feature extraction

In order to compare two accelerometer traces from two users. Using a reading from a signal axis is not feasible because the reading is affected by the placement of the phone. Another way is to combine the readings from all the three axes. Fig. 4(b) shows the result of combining three axes. As we can see, the crests and troughs in the trace almost disappeared. This is due to gravity, and the direction of the vehicle accelerating/decelerating and the direction of gravity are mutually perpendicular. A slight acceleration change in the vertical direction of gravity will cause very small change to the combination of the acceleration. For example, the gravity is $9.8m/s^2$, when the train acceleration is about $2m/s^2$, the combination is changed from $9.8m/s^2$ to $10m/s^2$, only $0.2m/s^2$ change appears. This means the acceleration signature when the user in the vehicle is weakened, which will cause error when comparing. When doing comparison, a simple approach is to use a absolute value as the feature and calculate the mean squared error (MSE) of the two waveforms. Since the users' phones have not been calibrated (i.e., the readings of the two phones have a constant drift under the same acceleration), there exists an unknown constant drift. This will cause error when calculating the MSE value.

To solve these issues, we use the variation of acceleration as the feature. Since the gravity keeps unchanged, the variation of acceleration is only affected by vehicle acceleration. The combined variation of the three axes is the variation of vehicle acceleration. The way to get the variation of each axis and the combination are shown in Equations 2 and 3. The direction of the combination is set as the direction of the axis with the max mean variation. In Fig. 4(c), the solid black line shows the combination of the variation of the three axes. By this transformation, the variation will not be affected by the phone orientation and the gravity, and the signature of vehicle acceleration is clearly. More important, the variation keeps the same even the two smartphones are not calibrated. Figure 4(c) shows an example variation traces of two smartphones in the same vehicle. We can see



(a) The calibration graph (b) An example calibration tree with an arbitrary root (c) Computed calibration tree

Figure 5: Calibration for all barometers

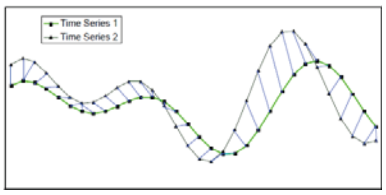


Figure 6: Dynamic Time Warping

that the fluctuation of the waveforms show the same patterns.

Before comparing two variation traces. As shown in Fig. 4(c). For every two accelerometer traces from users A and B, we get their minimum overlapped time t_0. From t_0, we get a series pairs of variation traces from users A and B, each trace contains variation of 60 seconds. For a special case, when two uses remains stationary, the acceleration keeps unchanged, the two variation trace will have the same value. This case must be removed. So, in our approach if the mean variation of a trace is smaller than 0.1, the pair is removed.

$$VarX_a(t) = X_a(t + \Delta t) - X_a(t) \tag{2}$$

$$VarAll_a(t) = \sqrt{VarX_a(t)^2 + VarY_a(t)^2 + VarZ_a(t)^2} \tag{3}$$

2.4.2 *Mathing with DTW*

Since the sampling rate of different smartphone may have a little different, a pair of variation traces we collect in the same time interval (60 seconds) will have different lengths of data, which cannot be handled by MSE. So the MSE based approach can not be used in this scenario. In our approach, we apply the Dynamic Time Warping Distance Measure (DTW) [10] which is less sensitive to the time shift. To calculate the DTW, we first align the two variation traces. For example, as shown in Fig. 6. for two time series of acceleration variance $VarM_s$ and $VarM_l$, where

$$VarM_s = s_1, s_2, s_3, s_4, ...s_n$$

$$VarM_l = l_1, l_2, l_3, l_4, ...l_m$$

the sequences $VarM_s$ and $VarM_l$ can be arranged to form a n-by-m plane or grid, where each grid $point(i, j)$ corresponds to an alignment between elements s_i and l_j. A warping path, W, maps or aligns the elements of $VarM_s$ and $VarM_l$.

$$W = w_1, w_2, w_3, w_4, ...w_k$$

The Dynamic Time Warping distance between two time series $VarM_s$ and $VarM_l$ is then:

$$DTW(VarM_s, VarM_l) = \partial(First(VarM_s), First(VarM_l)) + min\begin{cases} DTW(VarM_s, Rest(VarM_l)) \\ DTW(VarM_l, Rest(VarM_s)) \\ DTW(Rest(VarM_s), Rest(VarM_l)) \end{cases} \tag{4}$$

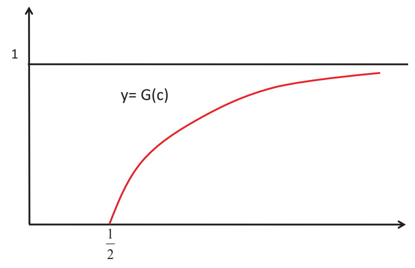
where $First(x)$ is the first element of x, and $Rest(x)$ is the remainder of the time series after the $First(x)$ has been removed, and $\partial(i, j) = (s_i - l_j)^2$. From the DTW value, we get numeric measure of the similarity between two user variation traces. In our approach, if the average DTW distance is smaller than 0.05 we conclude that the users are under the same barometric pressure, and the drift is get by

$$drift_{ij} = \frac{\sum_{t=t'}^{t'+60}(B_i(t) - B_j(t))}{n} \tag{5}$$

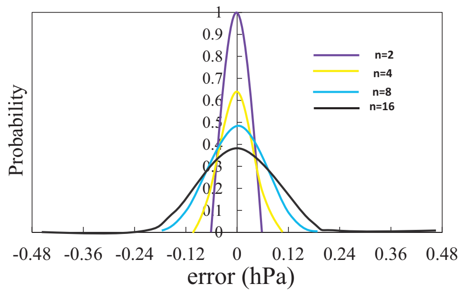
For better accuracy, we conclude users encounter in vehicle when there are $n(n \geq 2)$ pair of consecutive variation traces with DTW distance smaller than 0.05, and n is the confidence of the encounter.

2.5 Calibration for All Barometers

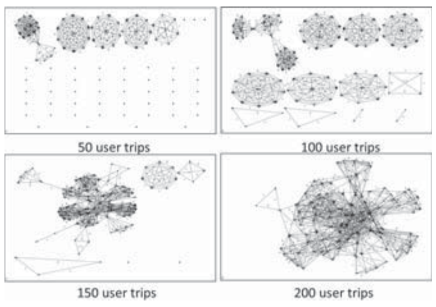
In the previous section, we present barometer calibration for two smartphones by detecting user encounters. To calibrate all user smartphones' barometers, while the same principle will be applied, the calibration propagates from phone to phone in a transitive way. We model this process using a graph shown in Fig. 5(a). In this graph, each barometer is represented by a node. If two barometers are calibrated by an encounter in elevator or vehicle, we draw an edge between the two nodes, and the weight of the edge represents the confidence value of the encounter (calibration). Since there may be more than one calibration done between two users (e.g., two users may encounter each other multiple times), we choose the calibration with the highest confidence value. Since barometer calibration is transitive, in theory, any two barometers can be calibrated if this graph is connected. To



(a) The curve $y = G(c)$



(b) Probability density function



(c) The calibration graph of 100 users in a 10-floor building

Figure 7: Theoretical Analysis

select a root barometer, a trivial approach is to randomly choose a node as root and find a spanning tree from the graph as shown in Fig.5(b). Any node in the spanning tree can be calibrated following the path from the node to the root. For example, to get the drift of barometer j, we find a path between node f and j, $f - k - j$, and obtain the drift by $drift_{fj} = drift_{fk} + drift_{kj}$.

There are two factors affecting the accuracy of our calibration algorithm. The first one is the confidence values of the edges in the path. The other one is the length of the path. The confidence determines the probability of correct calibration. The length of the path determines the calibration accuracy because errors may be accumulated along the path. This turns out to be an optimization problem—find a spanning tree, choose a root to minimize the sum of nodes' depths, and maximize the weight of the edges in the path. It has been proven to be a NP-complete problem [14], hence, finding this spanning tree is not realistic. In our approach, we propose a heuristic solution based on the observations that the confidences of edges in the path are important and the calibration errors accumulate slowly (which will be further analyzed in Section 3.2). We first find a maximum spanning tree which maximizing the edge confidences in the tree, we then choose a node as the root which minimizes the average depth of all other nodes. As an example, we run the algorithm on the graph shown in Fig. 5(a) and the resulting graph is shown in Fig. 5(c). The complexity of the algorithm is $O(NlogN)$.

3. THEORETICAL ANALYSIS

3.1 Modeling Calibration

We model the calibration process using a random graph $\Gamma_{n,N}$ with n labeled vertices and N edges, where a vertex represents a barometer (n is the total number of barometer sensors), and an edge may exist between two vertices if they are calibrated. A user sensor can be viewed as a set of connected vertices (i.e., connected component [13]) in $\Gamma_{n,N}$, All sensors can be calibrated if graph $\Gamma_{n,N}$ is connected. We denote p as the probability that there exists an edge between any two vertices in $\Gamma_{n,N}$, which is equal to the probability that two users encounter in an elevator. We denote p as the probability that there exists an edge between any two vertices in $\Gamma_{n,N}$, which is equal to the probability that two users encounter in an elevator. We assume the average number of users in elevator is k, the frequency that a user takes an elevator is f. We have

$$p = \frac{k * f * t}{n - 1} \tag{6}$$

where t is the time. It is shown in [13] that the random graph $\Gamma_{n,N}$ is almost surely be connected if

$$p > \frac{1}{n}(1 + \epsilon) \ln n \tag{7}$$

With a large n, the value of $(1 + \epsilon) \ln n/n$ is small, therefore a smaller p can meet the above equation.

It is shown in [13] that, in the random graph $\Gamma_{n,N}$, the size of the greatest component of $\Gamma_{n,N}$ is, for $c = \frac{N}{n}$ with $c > \frac{1}{2}$ with probability tending to 1, approximately $G(c)n$, where

$$G(c) > 1 - \frac{x(c)}{2c} \tag{8}$$

where

$$x(c) > \sum_{k=1}^{\infty} \frac{k^{k-1}}{k!} (2ce^{-2c})^k \tag{9}$$

The curve $y = G(c)$ is shown on Fig. 7(a). This means almost all points of $\Gamma_{n,N}$ belong to either some small component which is a tree or the single "giant" component of the size $G(c)n$. From this, we imply that, if $p > \frac{1}{n-1}$, most of the nodes will belong to the same "giant" component and all can be calibrated.

3.2 Calibration Error

We model the calibration accuracy as follows.

$$drift_{ij} = drift'_{ij} + \sum_{k=1}^{n} U_k \tag{10}$$

where n is the length of the path from node i to j, $drift'_{ij}$ is the real drift, and U is the noise function with a random value between -0.06 to 0.06 hPa. The accumulated noise function is $X = \sum_{k=1}^{n} U_k$. Since U has a uniform distribution, in probability and statistics, X is a Irwin-Hall distribution [4] function. The probability density function is

$$f_X(x:n) = \frac{1}{(n-1)!} \sum_{k=0}^{\lceil x \rceil} (-1)^k \binom{x}{y} (x-k)^{n-1}; x \in [0, n] \tag{11}$$

The curve of $f_X(x:n)$ is shown in Fig. 7(b). It is shown that the error is less than 0.24 hPa with high probability when the path length is less than 16. Since the minimum barometer reading distance of any adjacent floor level is about 0.45 hPa, the error is tolerable.

4. EVALUATION

We now move to evaluate our approach using both simulation and field studies.

4.1 Simulation Methodology

We design a simulator to evaluate the efficiency and scalability of SBC. The simulator models the calibration of a group of users who work in an multi-floor office building, it simulates the process of user taking the elevator up and down and arrive or leave the building by subway, bus or cars. For taking elevators, the simulation process is divided into cycles of elevator going up or down (which occurs with an equal probability). For elevator going up, each cycle simulates the process that the elevator goes up from the ground floor, with people entering and leaving the elevator from or to any levels, until the elevator is empty. It works as follows. The simulation process is divided into cycles of elevator going up or down (which occurs with an equal probability). For elevator going up, each cycle simulates the process that the elevator goes up from the ground floor, with people entering and leaving the elevator from or to any levels, until the elevator is empty. We model the process of people entering the elevator from the ground floor as the Poisson distribution. The expected number of the Poisson distribution is set to 1/4 of the maximum load of a typical elevator (i.e., four persons). People on the ground floor may go up to any floor with a probability of $1/(n-1)$, where n is the number of floors of the building. From any other floor f_i, some people may enter the elevator, and go to the rest $(n-i)$ floors with an equal probability $1/(n-i)$. Each cycle starts from the ground floor, we first compute the number of people entering the elevator and which floors they are going to, the elevator goes up from the ground floor, and stops when people exiting or entering, until there are no users in the elevator. For elevator going down, every time the elevator starts from the top floor, users in every floor may enter the elevator, and will go to the rest n' floors with an equal probability of $1/2(n'-1)$, except to the ground floor which is 1/2. When people enter or exit the elevator from a floor, the number of people on that floor gets updated, and the trace of every user is recorded. Based on our observation from real-life situations, in our simulation model, we assume that when an elevator passing a floor, the probability of a user in that floor entering the elevator is p (1% in our setting).

For arriving or leaving the building, we simulate the users will encounter in subway, bus or cars with a constant probability p. In the simulation of barometer readings of the building, the barometer reading of the ground floor is simulated with function $F(t) = F(t-1) + random() * 0.1$, where $F(0) = 1000hPa$ and t is the time, one unit of time is set to the time the elevator travels 1 floor, for simplicity, the barometer change of every floor is the same which is set to $0.45hPa$. Given a number of floors n and a number of users u, we simulate cycles of elevator going up and down until a certain number of user-elevator trips m is reached (a user-elevator trip is defined as the process of a user entering and leaving the elevator). And we will simulate user arrive and leave after $4 * u$ user-elevator trips are simulated, which means users averagely take the elevator 4 times everyday. At the end of each simulation cycle, we combine the ground truth from the floor-change activity detection to get the $MTrace$ of every user, and evaluate how well the barometer sensor of the users can be calibrated.

The parameters of the simulator are listed as follows.

1) user number at each floor: the number of users at each floor. *2) total trip number* : the number of user-elevator trips for all the users. *3) average trip*: the average number of user-elevator trips for each user. *4) average number of users in elevator*: the average number of users in elevator when it is moving.

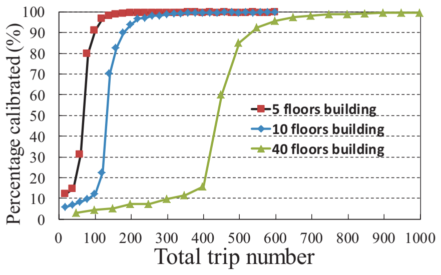
The performance metrics used in the paper are summarized as follows. *1) average hop*: The average hop of all nodes to the root in the calibration tree. *2) percentage calibrated*: The percentage of users who have been calibrated.

4.2 Simulation Results

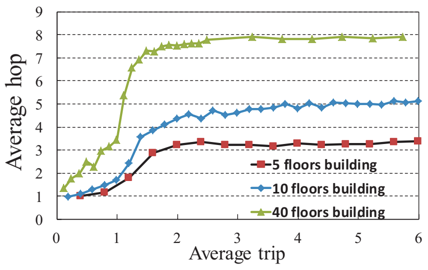
Figures 8(a) and 8(b) show the simulation results when the *user number at each floor* is 10 and the *average number of users in elevator* is 4. Figure 8(a) shows the *percentage calibrated* in three different buildings under different *total trip number*. It shows that all users can be calibrated after about 150/300/1000 trips in a 5/10/40-floor building. That means all users can be calibrated in a single day. Figure 8(b) show the *average hop* in the calibration tree. In Fig. 8(b), the hops is about 3/4.5/8 in the three buildings when all users are calibrated, which do not affect the calibration accuracy based on our analysis in Section 3.2. Figure 8(c) plots the calibration result with different *average number of users in elevator*. The result shows that the calibration process is faster with more average user in elevator. Figure 7(c) shows the calibration graph of 100 users in a 10-floor building

4.3 Field Study

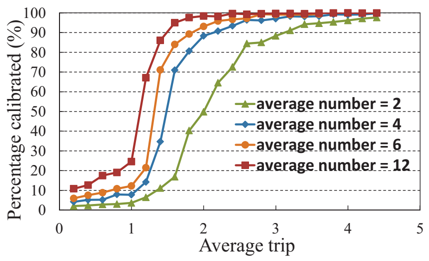
To evaluate SBC under the real-world situations, we implemented a prototype system and publish it on a website [6]. We encourage users to download and try SBC in our 10-floor computer science building. A total number of 67 users downloaded our application to their mobile phones (e.g., Samsung, Google Nexus, Sony Ericsson, etc). Out of 67 mobile phones, only 28 have both barometer sensors and a mobile network data connection (i.e., GPRS or 3G). We developed a mobile application named ”Talking to Strangers (up/down stairs)” which is built on top of SBC. The application finds users from other floors of a building for message chatting. This is similar to other chat applications such as find strangers around, but we incorporate SBC into our application to detect the floor of a user. When the application runs, it continuously collects barometer and accelermeter readings at a rate of 2 samples per second, and all the samples will be logged in a data file which will be uploaded to the cloud server every 2 hours. The taking elevator activity detection is done in real-time and the $MTrace$ will also be uploaded to the cloud server. The client also performs time synchronization with the sever by computing the round-trip delay time and the offset. The indoor/outdoor detection is



(a) The *percentage calibrated* with different *total trip number*

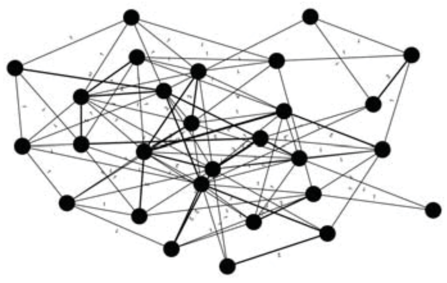


(b) The *average hop*

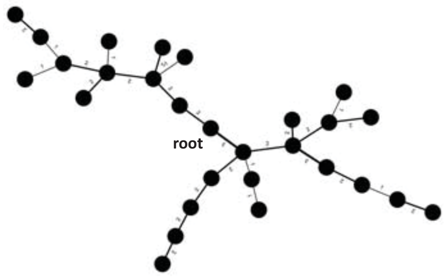


(c) Calibration result when *average number of users in elevator* is different

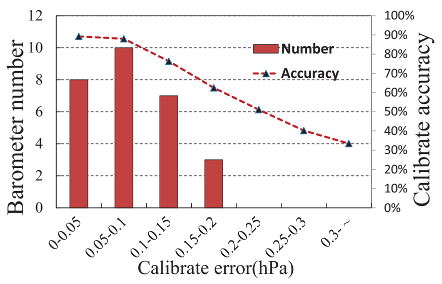
Figure 8: Simulation results



(a) The calibration graph of the users



(b) The calibration tree of the users



(c) The calibration error and accuracy

Figure 9: Field study results

done by scanning GPS signals. To get the ground truth, we first manually get the drift to the real barometric pressure for all the 28 smartphones, then placed a barometer logger at each floor to get the ground truth by comparing their readings with the ground truth. The field study ran for three days. In the cloud server, all calibration results and user accelermeter and barometer reading traces are logged. We analyze the performance based on the logged data in both the client and the cloud server.

4.4 Field Study Results

Figure 9(a) shows the calibration graph of the users at the end of the first day. The thickness of the lines represents the confidence (i.e., weight) of the calibration between two users. The graph is connected and users are directly calibrated (i.e., 2-8 users). The users' average trips are 6 (when coming in and going out for lunch and dinner). The calibration tree is extracted and shown in Fig. 9(b). The average hop of calibration is about 3, the max hop of calibration is 6, and the average calibration weight is about 1.8. The calibration error is shown in Fig. 9(c). The left vertical axis shows the number of barometers in different error regions. The right vertical axis shows the accuracy of the calibration in each region. The accuracy is defined as the ratio between error and the barometer reading distance of one floor.

4.5 Energy Consumption

We evaluate the energy consumption of SBC using a Samsung Galaxy Nexus smartphone running Android 4.1 OS. The power consumption is computed based on PowerTutor, a diagnostic tool for analyzing system and application power usage from the Android Market. The experiment ran for 12 hours continuously. The average power consumption of SBC is 26 mW. It shows that SBC is energy efficient.

5. RELATED WORK

Since barometer is a new sensor appears in the smartphones recently, to our best knowledge, this is the first time using the crowdsourcing based approach to do smartphone barometer calibration. The traditional barometers appeared many years, but the calibration of a large group of smartphone barometers is different from calibrating a traditional barometer, they are not comparable. In order to show the importance of calibrate the barometers, in this section, we talk about the smartphone barometer calibration used in floor localization, and compare it with other approaches not using smartphone barometer. Using a calibrated smartphone barometer, a smartphone can easily get the user's floor by transforming the barometer reading to altitude. When not using smartphone barometers, some fingerprint based techniques have been proposed such as [9,21]. They mainly rely on Wi-Fi signal strength. However, like RADAR [9] has to war-drive the building in order to obtain the map. War-driving is very time-consuming and labor-intensive, and it may have to be done periodically since the Wi-Fi signature at the same location may be changed over time. Hence, this solution is not scalable. The fingerprint based technique has been used in floor localization. SkyLoc [21] uses GSM fingerprints to locate a user's floor level in a multi-floor building. But the GSM signals vary significantly in indoor environments, and the training process in SkyLoc is time-consuming. It has a poor scalability since war-driving and training are required for every building. Different from these systems, SBC calibrate and makes use of the new barometer sensor appears in recent smartphones. It does not require war-driving to build the fingerprint database, SBC relies on crowdsourcing and intelligently calibrate the smartphone barometers, it can be used to locate users' floor level.

Sensor-assisted localization methods [11,12,15,19] have been proposed, making use of embedded sensors available on s-martphones. These systems typically use accelerometer and electronic compass. However, careful calibration is needed from time to time due to the limitations of the sensing technology. Crowdsourcing has been also used to reduce the training effort [8, 22]. These works rely on detecting user activities using sensors such as accelerometer. However, to ensure reliable detection, they typically require user-specific training which is costly, and the high sampling frequency which may drain the battery power quickly. FTrack [15] using accelerometer for floor localization. The main problem of this approach is that they cannot handle some practical issues such as different user walking patterns and a variety of ways to carry/use mobile phones, which may affect the accuracy and limit the feasibility. While SBC follows the basic principle of FTrack, it detects user activities of taking elevator by a novel barometer based technique, and it has no assumption of users walking pattern or the ways to carry/use mobile phones.

The increasing availability of barometer embedded in smartphones (e.g., Nexus 4) has motivated us to go beyond the existing work by building a simple, sensor-based, battery efficient solution for floor localization. Muralidharan's most recent paper [18] study on the properties of mobile-embedded barometers across a number of buildings. He concludes that it is difficult to use the barometer to determine the actual floor that a user is on. The main problem is that barometers are not calibrated, if all users' barometers are calibrated, it will be much easier to solve this problem.

6. CONCLUSION AND FUTURE WORK

This paper presents a novel, scalable barometer calibration scheme. Leveraging on mobile phone sensing and crowdsourcing, SBC requires neither any infrastructure nor any human intervention. Different from similar approaches, SBC relies on barometer and accelerometer only. SBC provides high accuracy of user encounter detection and minimum energy consumption, making it more realistic for real-world deployment. Our simulation and prototype system demonstrate the performance, scalability, and robustness of SBC. For our future work, we will further improve SBC by enhancing the calibration algorithm for larger number of users. We also plan to offer SBC as a free service for public use, and test SBC under real-life situations.

7. ACKNOWLEDGMENT

This work was supported by the China National High-tech R&D 863 program (2013AA01A213), the NSFC under Grants 91318301, 61373011, 61321491.

8. REFERENCES

[1] Altitude measuring, from google play.

[2] Barometric pressure. http://en.wikipedia.org/wiki/atmospheric_pressure.

[3] Google maps.http://maps.google.com/.

[4] Irwin-hall distribution. [online]. aviable at: http://en.wikipedia.org/wiki/irwin-hall_distribution.

[5] Runtastic, from google play.

[6] Sbc application and code. [online]. aviable at : http://moon.nju.edu.cn/twiki/bin/view/icsatnju/haiboye.

[7] C. D. Ahrens. Meteorology today: An introduction to weather, climate, and the environment hardcover. May 31čň2012.

[8] M. Alzantot and M. Youssef. Crowdinside: automatic construction of indoor floorplans. In *Proceedings of the 20th International Conference on Advances in Geographic Information Systems*. ACM, 2012.

[9] P. Bahl and V. Padmanabhan. Radar: an in-building rf-based user location and tracking system. In *INFOCOM 2000. Israel.*

[10] D. J. Berndt and J. Clifford. Using dynamic time warping to find patterns in time series. In *KDD workshop*. Seattle, WA, 1994.

[11] I. Constandache, X. Bao, and Azizyan. Did you see bob?: Human localization using mobile phones. In *MobiCom*. ACM, 2010.

[12] I. Constandache, R. Choudhury, and I. Rhee. Towards mobile phone localization without war-driving. In *INFOCOM2010*. IEEE.

[13] P. Erdos and A. Rényi. On the evolution of random graphs. 1960.

[14] M. Franceschet and e. Gubiani. From entity relationship to xml schema: a graph-theoretic approach. In *Database and XML Technologies*. 2009.

[15] G. X. Z. e. Haibo, Y. Tao. Ftrack: Infrastructure-free floor localization via mobile phone sensing. *PerCom*, 2012.

[16] X. T. e. Haibo, Y. Tao. B-loc: Scalable floor localization using barometer on smartphone. *MASS*, 2014.

[17] X. T. e. Haibo, Y. Tao. Crowdsourced smartphone sensing for localization in metro trains. *WoWMoM*, 2014.

[18] K. Muralidharan, A. J. Khan, A. Misra, R. K. Balan, and S. Agarwal. Barometric phone sensors–more hype than hope! ACM HotMobile 2014.

[19] A. Ofstad, E. Nicholas, R. Szcodronski, and R. Choudhury. Aampl: Accelerometer augmented mobile phone localization. In *Proceedings of the first ACM international workshop on mobile entity localization and tracking in GPS-less environments*, pages 13–18. ACM, 2008.

[20] C. Peng, G. Shen, and Zhang. Beepbeep: a high accuracy acoustic ranging system using cots mobile devices. In *Proceedings of the 5th international conference on Embedded networked sensor systems*, 2007.

[21] A. Varshavsky, A. LaMarca, J. Hightower, and E. de Lara. The skyloc floor localization system. in PerCom 2007.

[22] H. Wang, S. Sen, A. Elgohary, M. Farid, M. Youssef, and R. R. Choudhury. No need to war-drive: Unsupervised indoor localization. In *MobiSys*, pages 197–210. ACM, 2012.