SurfaceSense: Smartphone Can Recognize Where It Is Kept

Rajkumar Darbar
Indian Institute of Technology, Kharagpur, India
rajdarbar.r@gmail.com

Debasis Samanta
Indian Institute of Technology, Kharagpur, India
dsamanta@iitkgp.ac.in

ABSTRACT

In recent times, researchers have proposed numerous approaches that allow smartphones to determine user current locations (e.g., home, office, railway station, restaurant, street, supermarket etc.) and their activities (such as sitting, walking, running, bicycling, driving, cutting bread, making coffee, watching television, working at laptop, taking lunch, using water tap, brushing teeth, flushing toilet etc.) in real-time. But, to infer much richer story of context-aware applications, it is necessary to recognize the smartphone surfaces - for example on the sofa, inside the backpack, on the plastic chair, in a drawer or in your pant-pocket. This paper presents SurfaceSense, a two-tier, simple, inexpensive placement-aware technique, that uses smartphone's embedded accelerometer, gyroscope, magnetometer, microphone and proximity sensor to infer where phone is placed. It does not require any external hardware and provides 91.75% recognition accuracy on 13 different surfaces.

CCS Concepts

• Human-centered computing ∼ Ubiquitous and mobile computing • Computing methodologies ∼ Machine learning algorithms.

Author Keywords

Context-aware computing; surface detection; smartphone sensors; machine learning and pattern recognition.

1. INTRODUCTION

Of late, smartphone usage trend is increasing at a rapid speed, and at the same time, mobile devices are becoming really smarter day-by-day. According to eMarketer statistics[1], there was almost 1 billion smartphone users all over the world in 2012 and by 2014, it'll be nearly 1.75 billion. It is expected that smartphone adoption will continue on a fast-paced trajectory through 2017. At the same time, smartphones are becoming really smarter day-by-day. With the proliferation of electronics technology, it is now possible to embed a larger number of sophisticated and complex sensors (such as accelerometer, gyroscope, magnetometer, ambient light sensor, GPS, proximity sensor, humidity sensor, temperature sensor, pressure sensor, camera, microphone etc.) in mobile-phones. By leveraging mobile sensing capabilities, researchers and engineers are trying to develop several effective and interesting context-aware applications.

Now-a-days, everyone carries their phone with them almost all time and in all places. Therefore, smartphone became ubiquitous in nature. Presently, mobile devices know little about the context in which they operate and so, user takes responsibility for managing their behavior to fit with the current environment. Performing these settings manually is an extra burden to users. For example, if user enters into a classroom, she has to manually set her cell-phone into silent mode. To provide intelligent supports to users, researchers are focusing on the development of context-awareness into mobile devices such that it should automatically adapt with user's changing environment or context[13][14]. There are two perspectives in context-aware applications, that is, user's context and phone's context. The user's context mainly focus on detection of user's current location (e.g., home, office, railway station, restaurant, street, supermarket etc.)[10][11] and their real-time activities (such as sitting, walking, running, bicycling, driving, cutting bread, making coffee, watching television, working at laptop, taking lunch, using water tap, brushing teeth etc.)[7][8][9] using smartphone.

So far, its all about user's context detection. But, to understand a detail analysis of context aware applications, it is also important to indentify phone's context, that is, on the sofa, inside the backpack, on the plastic chair, in a drawer or in your pant-pocket. Automatic detection of smartphone's context has several advantages. For example, (a) if phone is in pocket, then it automatically turns off the display and locks it to prevent pocket-dialing; (b) if cell-phone is in backpack, then phone should ring at the highest volume at the time of incoming call. Similar setting is desirable, if it is on bed or sofa; (c) in pant-pocket location, vibration is sufficient to draw user's attention; (d) if phone is on hard surface like wooden table or metal cabinet, then vibration may cause damage in phone. In this case, only ringing is enough; (e) if phone is put in drawer, it is reasonable to assume that in near future, phone'll not be used. So, it can go to the power saving mode; (f) if phone is in pocket, then don't activate pollution sensor; (g) it also helps to detect user's context. For instance, when a phone is on table, the microphone may be given the highest priority to estimate user's surrounding environment from ambient sound.

In this paper, we have two objectives. First objective is to understand where people normally keep their phones across various contexts such as at home, office-place, driving, sleep-

[1]http://www.emarketer.com/Article/Smartphone-Users-Worldwide-Will-Total-175-Billion-2014/1010536

IndiaHCI'15, December 17-19, 2015, Guwahati, India
 ISBN 978-1-4503-4053-3/15/12$15.00
http://dx.doi.org/10.1145/2835966.2835971

ing etc. For this purpose, we carried out a user study and identified 13 different phone placements like wooden table, soft-bed, glass table, backpack, plastic chair, cart-box, fabric chair, phone holder, metal chair, wooden-drawer, metal drawer, pant's pocket, and user's hand.

In our second objective, we mainly focus on how to identify these locations automatically using phone's inbuilt sensors. The automatic sensing of phone's context has several challenging issues: which are the most useful sensors to build low cost and robust context inference system? how to fuse multi-sensor data? which are the most effective feature-set? which machine learning algorithm provides reliable results? what is the trade-off between inference accuracy and power consumption? what about processing-time and memory requirement? how does system response in noisy environment?

Considering all these challenges, in this paper, we present SurfaceSense, a scalable phone's context sensing framework, to recognize 13 different phone's placements using phone's embedded accelerometer, gyroscope, magnetometer, microphone and proximity sensor. Our proposed technique primarily works in two steps. First step identifies that there is a change in phone's placement, and second step detects phone's current location. To understand the change in phone's placement, we proposed a simple threshold based algorithm on tri-axial accelerometer signal and it is running continuously as a back-ground service. When change in phone's placement is detected, phone vibrates for four seconds and during the vibration, accelerometer and gyroscope record motion data, magnetometer records magnetic field strength, proximity sensor measures the presence of nearby objects at different distance levels, and microphone captures phone's vibration echoes. Once sensor data collection is completed, data are processed to identify those surfaces. The surface recognition procedure is basically a two-tier hierarchical classification approach. In the first and second level, a simple 'if-else' rule-based reasoning module is used on the basis of proximity and magnetometer sensor data pattern to categorized surfaces into four subgroups, that is, metal and non-metal inner surfaces, metal and non-metal outer surfaces. Then, extracted features from accelerometer and gyroscope sensor data and recorded vibration echoes are fed into four Random Forest (RF) classifiers to infer surfaces from each group. Note that, there is one RF classifier for each subgroup.

In particular, key contributions of this paper are summarized as follows:

1. We present the system architecture of SurfaceSense, which follows two-tier hierarchical classification approach, to recognize 13 different phone placements using smartphone's built-in sensors with 91.75% accuracy. The advantage of our proposed method is that it requires a less number of surface candidates to build the classifier module for each subgroup and reduces overall complexity.

2. We propose a simple threshold based algorithm using accelerometer signal to detect change in phone placement.

3. We have implemented SurfaceSense system architecture as an Android application on the Samsung Galaxy S4 and analyzed the resource consumption profile in terms of CPU and Memory usage.

The remainder of the paper is structured as follows: Section 2 describes related work on phone placement recognition. In Section 3, a brief phone placement study has been presented. The Section 4 describes SurfaceSense system architecture and Section 5 focuses on the experimental results. The Section 6 details the implementation of SurfaceSense as an Android App. Finally, Section 7 concludes the paper.

2. RELATED WORKS

Phone context sensing is an active research field. So far, various approaches have been proposed using phone's embedded sensors and sometimes, with the help of external hardware. Harrison et al. [2] identified proximity materials of mobile devices using multi-spectral optical sensor. In their approach, different LEDs (i.e. infrared, red, green, blue and ultraviolet) were used to artificially illuminate the target material and photo-register measured the reflected light properties (i.e. wavelength). The experimental result showed 86.9% placement-detection accuracy for 27 different test placements. This approach consumes less power ($\sim$20 mA) and faster enough (took 5 sec). In [5], the authors built *Phoneprioception* model with 85% accuracy using experience sampling method (ESM) to infer phone placements. They also demonstrated that reasonably accurate classification is possible using proximity sensor, light sensor, and multi-spectral sensor. F. Wahl et al.[15] presented RFID tag based phone placement inferring method. In their experiment, they placed RFID tags at pant, table, jacket and bag. Smartphone's built-in NFC reader automatically scans RFID tags when phone passes these sites and recognizes the places with an average accuracy of 80%.

The methods described in paper [2] [5] and [15], demand some external hardware setup. But, it would be much better and robust, if cell-phone's inbuilt sensors (such as accelerometer, gyroscope, microphone, proximity sensors, magnetometer etc.) can be used to infer phone surface. Keeping this in mind, Cho et al. [4] proposed *VibePhone* where extracted Jigsaw and time histogram features from vibration generated acceleration readings are used as input to the SVM classifier that recognized six contract surfaces (i.e. sofas, plastic tables, wooden tables, hands, backpacks, and pants pockets) with 85% accuracy. In [6], S. Hwang et al. proposed *VibroTactor*, an easy and inexpensive solution, by analyzing smartphone's microphone captured acoustic signal generated when the mobile device vibrates. They derived several characteristics (such as peak count, peak intensity, peak frequency, and skewness) from spectrograms of vibration echoes on different placements. These features are fed to the RBF classifier that achieves recognition rate of 91% in 12 different real-world placement sets. Kunze et al.[1] proposed a symbolic phone location method based on active sampling of vibration generated motion data captured by accelerometer and 'beep' sound signatures. They achieved recognition rates of up to 81% for 35 trained locations and 86% for 12 abstract locations. In paper [16], J. Yang et al. demonstrated a low cost solution using smartphone embedded proximity (IR) and

light sensor to detect 'in pocket', 'in bag', 'out of pocket' and 'out of bag'. The average accuracy of their demo prototype is above 98% and it consumes less than $\sim$6mW power for collecting sensor readings. Recently, I. Diaconita et al. [17] presented an acoustic probing based approach to detect 'in a backpack', 'on desk', 'in user's hand' and 'in user's pocket'. In their approach, mobile phone emits and records short bursts of inaudible audio signals while it is placed at above mentioned positions. The differences in signal attenuation reveal the nature of the material surrounding the mobile phones. They performed this experiment in various environments such as office, bus, train, outdoors etc. For identification purpose, they extracted MFCC, DMFCC and Band Energy (BE) features from the recorded audio signal. Finally, they achieved 97% and 96% classification accuracy using K-Nearest Neighbors and Random Forest respectively.

3. PHONE PLACEMENT STUDY

To understand where people commonly keep their phones in everyday life, we interviewed 92 participants (60 male and 32 female), all aged between 20-38 (Mean = 29). They were primarily graduate, post-graduate students in our university's information technology department. They are all well experienced with smartphones. All of them were given a compensation for their time. We asked them two questions: (a) where they normally keep their phones in five different contexts like, in home while awake, in home while sleeping, at office, driving and walking around; (b) how they decide where to put their phone. Table 1 shows where people put their phones across various activities. We were surprised that in this study none mentioned belt-pouch as a choice for carrying mobile phone while they are walking or driving. However, due to the increased size (width and height) of the latest smart phones, it is impractical to carry the phone in belt-pouch. In fact, belt-pouches are not commercially available for many latest smart phones. From this phone placement study, we got total 13 surfaces and decision factors for selecting these surfaces are easy to access notifications, phone's safety, physical comfort, minimize distraction, common habit, near by charger socket and so on.

Table 1. Phone placements during different activities.

Activities	Phone Placements
Awake at home	wooden table (62%), glass table (17%), in hand (13%),plastic chair (3%), fabric chair (2%), metal chair (2%), cardboard-box(1%)
Sleeping at home	bed (78%), wooden table (9%), glass table (6%), wooden drawer (3%), phone holder (2%), metal drawer (1%)
Working at office	wooden table (81%), pant's pocket (12%), in hand (7%)
Driving	pant's pocket (94%), backpack (6%)
Walking	pant's pocket (91%), backpack (9%)

4. SURFACESENSE FRAMEWORK

The SurfaceSense system architecture fundamentally comprises two parts: (1) change in phone placement detection and (2) phone's context sensing. The working of the said two modules are discussed in the following sub-sections.

4.1. Change in Phone Placement Detection

Detection of change in phone placement means how phone will automatically understand that user has changed its current position. For example, user is taking the phone placed on the table and putting it in his pant's pocket. In this case, phone position has changed from on the table to inside pant's pocket. In this paper, the proposed change in phone placement detection algorithm will run continuously as a background service. It is a simple threshold based algorithm relying on the captured data of the smartphone's tri-axial accelerometer sensor. Since this algorithm will run continuously, phone's battery consumption is an important concern. To optimize the power consumption of the device, we use only the accelerometer signal as it is the most informative sensor regarding the change in phone placement detection.

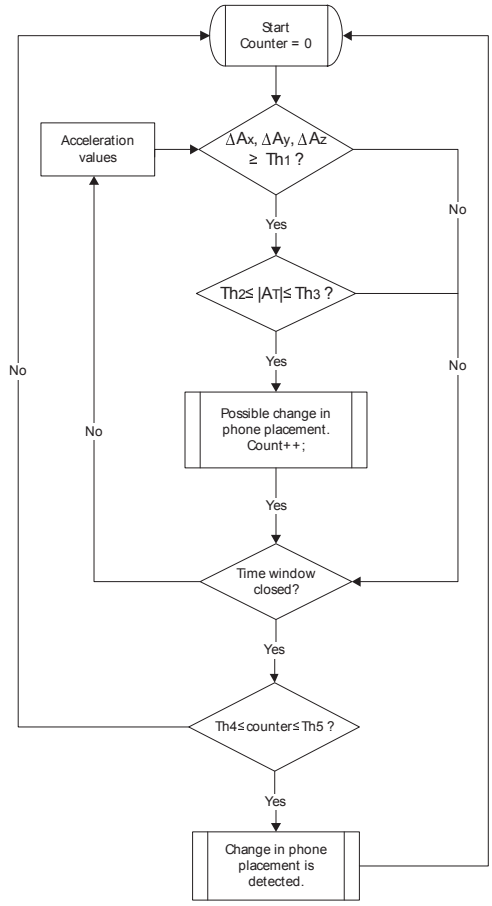


Figure 1. Flow diagram of the change in phone placement detection algorithm.

Figure 1 represents the flow-diagram of change in phone placement detection algorithm. To understand that there is a change in phone placement, we continuously analyze t sec of accelerometer signal (A_x, A_y and A_z) window in real-time. If change in each axis acceleration (ΔA_x, ΔA_y and ΔA_z) exceeds threshold Th_1, then we calculate the norm of the current accelerometer signal as described in Equation (1).

$$|A_T| = \sqrt{|A_x|^2 + |A_y|^2 + |A_z|^2} \quad (1)$$

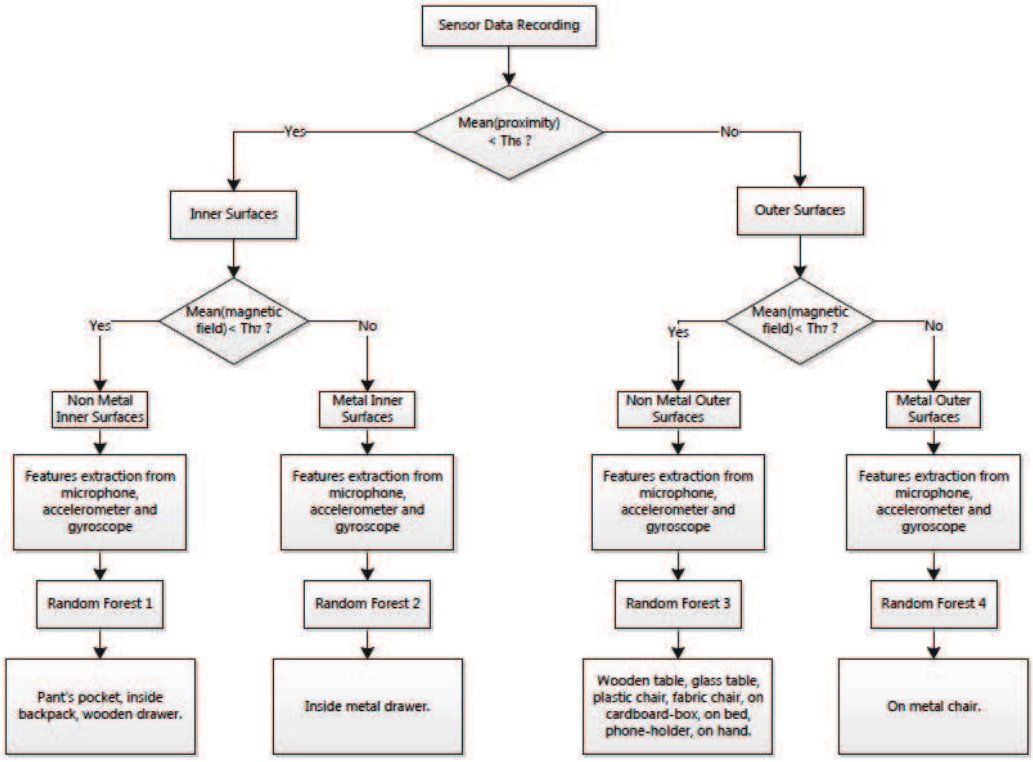
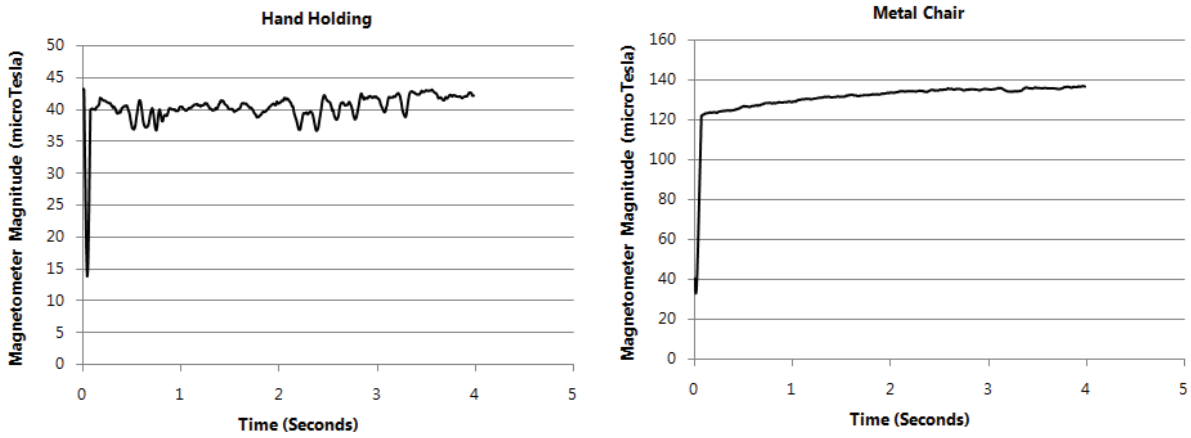


Figure 2. Flow diagram of phone's context sensing algorithm.



(a) (b)

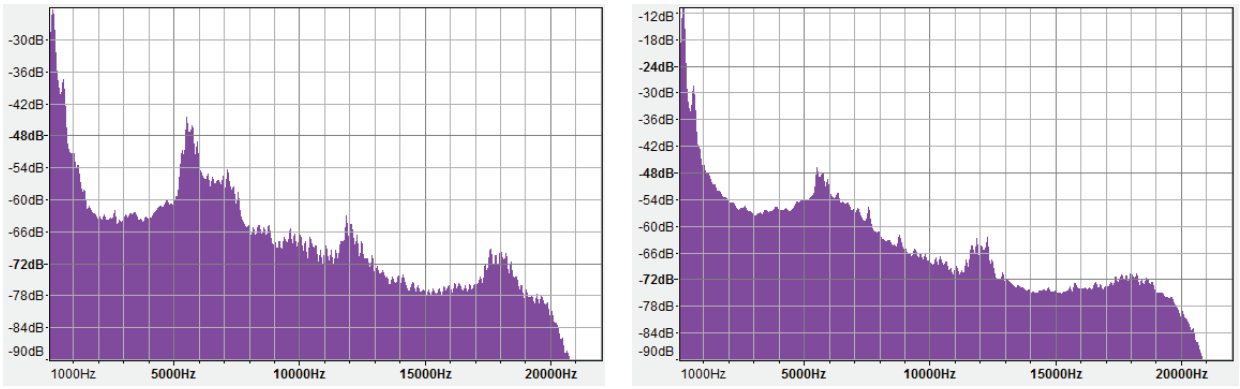
Figure 3. Magnetometer readings for (a) hand-holding position and (b) metal chair position.

Now, check that if $|A_T|$ is within the range of Th_2 and Th_3, that is, $Th_2 \leq |A_T| \leq Th_3$ and two consecutive $|A_T|$ satisfy this criteria within a given time interval of δ msec ($\delta \ll t$), then a counter increases every time and change in phone placement is suspected. In the final step, if counter status is $Th_4 \leq counter \leq Th_5$ after δ sec window, then there is a real change in phone placement.

If $counter < Th_4$, it means phone is in the same place. On the other hand, if $counter > Th_5$, then there is a chance that user is doing other activities like walking, running, going up or down the stairs.

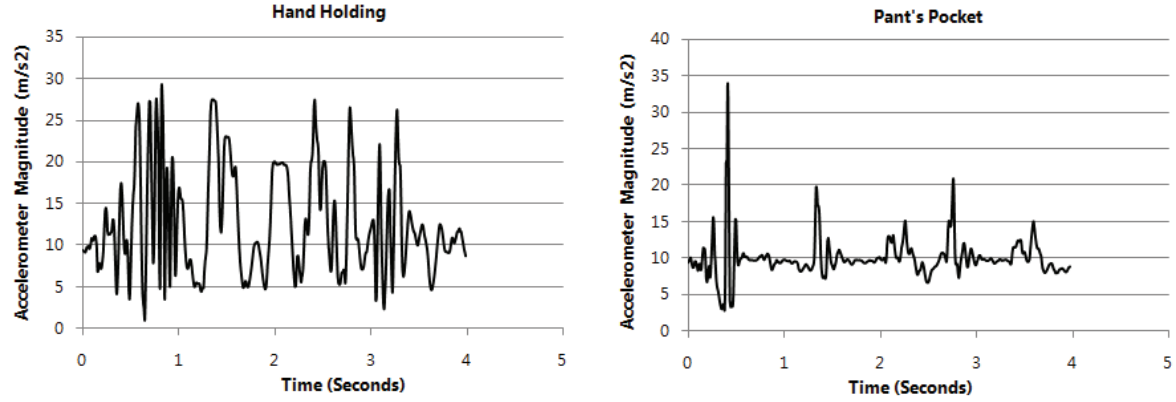
4.2. Phone's Context Sensing

If change in phone placement is detected, then phone's context sensing module starts working. This module fundamentally comprises three parts: (1) surface categorization using proximity and magnetometer sensors (2) feature extraction from microphone, accelerometer and gyroscope sensor data (3) Random Forest classifier to recognize phone placement.



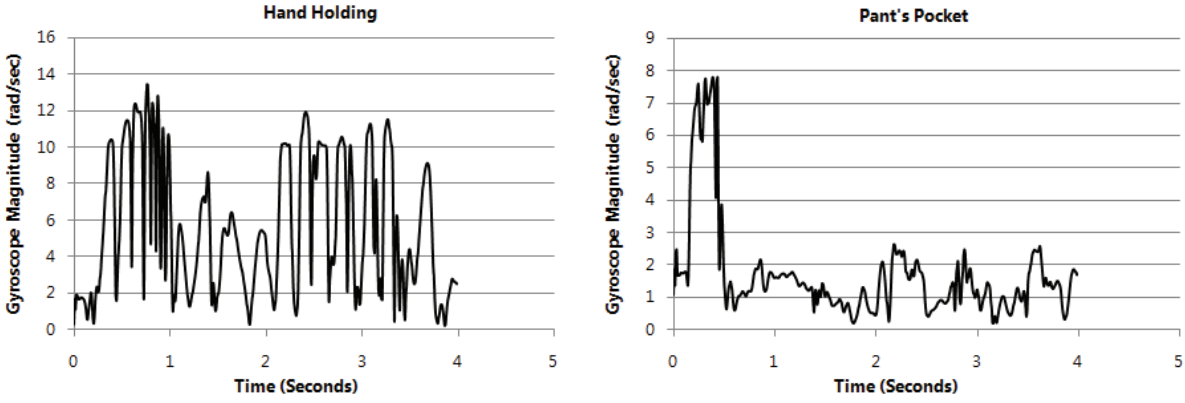
(a) (b)

Figure 4. Spectrogram of vibration echoes for (a) hand holding and (b) pant's pocket position.



(a) (b)

Figure 5. Accelerometer readings for (a) hand holding and (b) pant's pocket position.



(a) (b)

Figure 6. Gyroscope reading for (a) hand holding and (b) pant's pocket position.

Figure 2 represents the flow-diagram of phone's context sensing algorithm. The details of each part of this algorithm is discussed below.

4.2.1. Surface Categorization

This surface categorization works in two levels. In the first level, proximity sensor is employed and on the basis of proximity sensor data, phone placements are categorized into two broad groups: inner (i.e. pant's pocket, backpack etc.) and outer (i.e. plastic chair, wooden table etc.) surfaces. From the experiment, it is observed that proximity sensor of Samsung Galaxy S4 returns 0.00c.m. as the average value of proximity distance for inner surfaces and in case of outer surfaces, it is 8.00 c.m. Therefore, Th_6 = 5 c.m. (empirically) is con-

sidered as a threshold to distinguish between inner and outer surfaces.

In the second level, magnetometer sensor is used to further categorize each group into two subgroups : metal and non-metal surfaces. To be specific, non-metal inner surfaces are pant's pocket, backpack, and wooden drawer; metal inner surface is metal drawer; non-metal outer surfaces are wooden table, soft-bed, glass table, plastic chair, cart-box, fabric chair, phone holder, and user's hand; metal outer surface is metal chair. In the experiment, we logged magnetometer sensor data at the sampling rate of 100Hz. For non-metal surfaces, strength of magnetic field varies between 15 μT to 50 μT and in case of metal surfaces, it is approximately 65 μT to 200 μT. For example, Figure 3(a) and (b) represent magnetometer readings related to the positions of hand-holding and metal chair respectively. Here, we choose Th_7 = 62 μT as a threshold to distinguish metal surfaces from non-metal one.

4.2.2. Feature Extraction

During vibration phase, the device itself records the vibration echoes using microphone at 44.1 KHz sampling frequency and motion data with the help of accelerometer and gyroscope sensors at 100 Hz sampling frequency. These vibration sound and motion data vary depending on surfaces. For example, Figure 4(a) and (b) represent different sound signatures for hand-holding and pant's pocket positions respectively. Figure 5(a) and (b) depict the magnitude of accelerometer raw values for two smartphone positions, hand-holding and pant's pocket. Likewise, Figure 6(a) and (b) display two plots of gyroscope readings related to hand-holding and pant's pocket positions. To characterize different surfaces, we extract time and frequency domain features from recorded signals. Feature extraction step consists of three parts: (a) vibration sound features (b) accelerometer features and (c) gyroscope features.

Vibration Sound Features

This sound fingerprint is processed in frames with a 20 msec sliding window and 50% overlap. Each window is smoothed with a hamming filter and the following features are extracted. [1 - 2]: zero crossing rate, short-time energy (time domain); [3 - 6]: spectral flux, spectral rolloff, spectral centroid, and Spectral entropy (frequency domain). For each 6D feature vector, the standard deviation is calculated over all windows.

Accelerometer Features

The 3 axis accelerometer readings are divided into frames and we find global vertical (the direction of gravity) and horizontal (perpendicular to the direction of gravity) components from each frame to eliminate smartphone's different orientation effects in the feature set. To do this, we use a simple normalization scheme as described in paper [12]. The algorithm works as follows.

Let the raw accelerometer readings in a frame be a_i = $[a_x(i), a_y(i), a_z(i)]$, i = 1,...,n, where $a_x(i)$, $a_y(i)$, and $a_z(i)$ are the accelerometer readings along x, y and z axis respectively and n is the number of accelerometer readings in a frame. Note that, a non-overlapping rectangular window is used in the framing process and window size is 320 msec. We obtain vertical acceleration vector p corresponding to gravity as p = $[m_x, m_y, m_z]$, where m_x, m_y and m_z are the average values in each axis, that is, $m_x = \frac{1}{n}\sum_{i=1}^{n} a_x(i)$. The dynamic component of a_i, caused by the user's motion rather than gravity, is represented as d_i = $[a_x(i) - m_x, a_y(i) - m_y, a_z(i) - m_z]$. Then using vector dot product, vertical component v_i is computed as $v_i = (\frac{d_i \bullet p}{p \bullet p})p$. The horizontal component, h_i, is calculated as $h_i = (d_i - v_i)$. Finally, we use $\| h_i \|$ and $\| v_i \|$ as horizontal and vertical components to extract following features. [1 - 8]: mean, std, min and max of vertical and horizontal acceleration respectively (time domain); [9 - 16]: min, max, kurtosis and skewness of vertical and horizontal acceleration respectively (frequency domain). For each 16D feature vector, the standard deviation is calculated over all windows.

Gyroscope Features

We consider only time domain features from the magnitude of gyroscope readings, that is, [1 - 4]: mean, std, min and max. Here, we use a non-overlapping rectangular window of size 320 msec for framing purpose. Finally, to get 4D feature vector from recorded gyroscope readings, the standard deviation is computed over all windows.

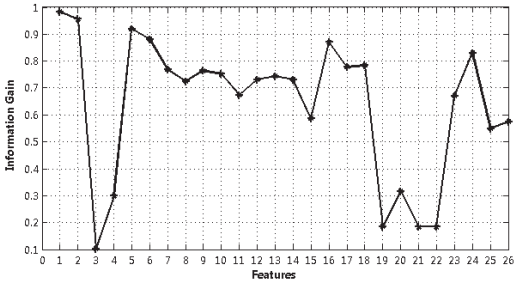


Figure 7. The measurement of information gain for 26 features.

Ultimately, we have total 26 features, that is, our current feature vector is of 26 dimensions (6D + 16D + 4D). In machine learning, the selection of an appropriate set of features is a key step in designing a robust classifier. Here, we applied 'information gain' based feature selection method, provided by WEKA tool, to reduce the dimensionality of feature space by eliminating redundant features. Mathematically, information gain is defined as

$$IG = H(X) - H(X|Y) \tag{2}$$

In equation (1), H(X) is the entropy of a feature X and $H(X|Y)$ is the entropy of X, after observing Y. They are defined as $H(X) = -\sum_i P(x_i)log(P(x_i))$ and $H(X|Y)$ = $-\sum_j P(y_j) \sum_i P(x_i|y_j)\log(P(x_i|y_j))$; where $P(x_i)$ and $P(x_i|y_j)$ are the prior and posterior probabilities of X, respectively. Information gain helps classifier to improve prediction performance, reduces over-fitting and minimizes the training time. Figure 7 shows the measure of information

gain (IG) for different features and to select the most effective features, we set IG = 8.00 as threshold. Hence, the most relevant features according to information gain are zero crossing rate, short-time energy, spectral centroid, spectral entropy, min. of FFT amplitude along horizontal accelerometer component, and min. of gyroscope amplitude, which are used to train and test our classification model.

4.2.3. Classifier

We use Random Forest (RF) classifier [3], provided by WEKA tool[2], to recognize phone placements. RF is an ensemble learning method for classification that operates by constructing a multitude of decision trees at training time and outputting the class that is determined by a majority vote of the trees. We preferred Random Forest because it is fast and efficient for training, and more importantly, it is computationally much lighter than other classifiers. Note that, computational time of RF is $O(T(MN \log(N))$, where T is the number of trees in the ensemble, M is the number of features and N is the number of samples in the data set.

To detect phone's current context, we used four Random Forest classifiers. As per Fig.2, Random Forest$_1$ for non-metal inner surfaces, Random Forest$_2$ for metal inner surfaces, Random Forest$_3$ for non-metal outer surfaces, and Random Forest$_4$ for metal outer surfaces. The basic difference between these classifiers is that they are trained with different datasets. For example, Random Forest$_1$ is trained with a dataset which belongs to non-metal inner surface group, whereas, dataset of Random Forest$_2$ belongs to the metal inner surface group.

5. EXPERIMENTS

5.1. Performance of Change in Phone Placement Detection Module

We implemented change in phone placement detection module using Samsung Galaxy S4 Android smartphone and we capture its accelerometer data at 100 Hz sampling frequency. Time window (t), data interval (δ) and threshold setting are an essential part of our algorithm as shown in Fig.1. We analyzed accelerometer pattern from 10 participants in different phone placement changing scenarios and set t = 2 sec, Th_1 = 2.5 m/sec^2, Th_2 = 11 m/sec^2, Th_3 = 16 m/sec^2, δ = 250 msec, Th_4 = 1 and Th_5 = 8 after conducting experiments. Ultimately, we achieved 71.24% accuracy to detect the change in phone placement.

5.2. Performance of Phone's Context Sensing Module

5.2.1. Data Collection

For the experiments, we use Samsung Galaxy S4 GT-I9500 Android smartphone which has inbuilt microphone, accelerometer, gyroscope, magnetometer and proximity sensors. To collect training and test dataset, we tried for three days in lab environment with Galaxy S4 phone on 13 different surfaces. For each surface, we vibrated the phone for 4 seconds and recorded vibration sound using microphone at 44.1 KHz sampling rate, motion data using accelerometer and gyroscope at 100 Hz sampling frequency, magnetic field strength using magnetometer at 100 Hz sampling rate and proximity distance using proximity sensor at a sampling frequency of 10 Hz. We repeated this procedure 80 times for each surface with different phone orientations. In this way, we collected total 1040 samples (80×13) and divided it into four datasets, that is, dataset$_1$ for Random Forest$_1$, dataset$_2$ for Random Forest$_2$, dataset$_3$ for Random Forest$_3$ and dataset$_4$ for Random Forest$_4$. The dataset$_1$ contains 240 samples from pant's pocket, backpack, and wooden drawer; dataset$_2$ contains 80 samples from metal drawer; dataset$_3$ contains 640 samples from wooden table, soft-bed, glass table, plastic chair, cart-box, fabric chair, phone holder, and user's hand; dataset$_4$ contains 80 samples from metal chair.

5.2.2. Classification Results

We split each dataset into three subsets for conducting three fold cross validation test. Table 2 represents the overall performance of SurfaceSense system. The average accuracy stands at 91.75%, with more than half of the phone placements achieving accuracies over 90%. From our result, it is observable that soft and hard surfaces can be distinguished with good accuracy, but it is quite difficult to recognize several similarly hard surfaces (e.g. wooden table and glass table). However, this is an excellent outcome with respect to the results of few previous works [1],[4],[6] and [16].

Table 2. Performance evaluation of 13 phone placement detection. The average accuracy is 91.75%

Placements	Accuracy	Confused Surfaces
Pant's pocket	98.82%	backpack (1.18%)
Backpack	97.21%	pant's pocket (2.79%)
Wooden drawer	100%	-
Metal drawer	100%	-
Wooden table	78.43%	glass table (16.61%), card-board box (4.96%)
Glass table	69.41%	wooden table (25.12%), plastic chair (5.47%)
Card-board box	78.88%	glass table (13.52%), plastic chair (3.9%), wooden table (3.7%)
Plastic chair	95.2%	glass table (3.2%), card-board box (1.6%)
Soft-bed	86.37%	fabric chair (13.63%)
Fabric chair	88.54%	soft-bed (11.46%)
Phone holder	100%	-
User's hand	100%	-
Metal Chair	100%	-

6. SURFACESENSE ANDROID APP

We have developed SurfaceSense system as an Android app in Samsung Galaxy S4. The main components, as described in 'SurfaceSense Framework' section, were fully implemented in Java SE 7 and successfully running on the Android smartphone. It took almost 22.35 seconds to correctly detect a phone-placement. We also measured the CPU, memory, and power consumption footprints of SurfaceSense app on Galaxy S4 with the help of 'Power Tutor' and 'OS Monitor' applications, available at Google Play Store. The CPU usage is less than 4% during idle state and on average of 14%

[2]http://www.cs.waikato.ac.nz/ml/weka/

at the processing time. The memory consumption is ∼7.38 MB during silence and reaches ∼12.26 MB during running period. The average power consumption for sensor reading is less than 18.76mW.

7. CONCLUSIONS

We presented SurfaceSense, which is inexpensive, calibration free, and does not demand any external hardware. It only requires phone's build-in accelerometer, gyroscope, magnetometer, microphone and proximity sensors. This placement aware technique can be applied easily to develop various context-aware applications. Our proposed method provides almost 91.75% accuracy to recognize 13 different phone placements. One technical issue is that we did our experiment with plastic covered phone, but recognition rate may be affected by the presence of rubber cover or a leather cover. In future, we'll experiment our algorithm (1) with different mobile devices except Samsung Galaxy S4 and (2) in noisy environment to test its feasibility at high scale.

REFERENCES

1. K. Kunze and P. Lukowicz. 2007. Symbolic object localization through active sampling of acceleration and sound signatures. In *Proc. of the 9th International Conference on Ubiquitous Computing* (UbiComp '07), 163-180. http://dx.doi.org/10.1007/978-3-540-74853-3_10
2. Harrison, C. and Hudson S. 2008. Lightweight material detection for placement-aware mobile computing. In *Proc. of the 21st Annual ACM Symposium on User Interface Software and Technology* (UIST '08), 279-282. http://doi.acm.org/10.1145/1449715.1449761
3. Breiman, Leo. 2001. Random Forests. In Machine Learning, 45(1), 5-32.
4. J. Cho, I. Hwang, S. Oh. 2012. Vibration-Based Surface Recognition for Smartphones. In *Proc. of IEEE International Conference on Embedded and Real-Time Computing Systems and Applications* (RTCSA '12), 459-464. http://dx.doi.org/10.1109/RTCSA.2012.42
5. J. Wiese, T. S. Saponas, A. Brush. 2013. Phoneprioception: enabling mobile phones to infer where they are kept. In *Proc. of the SIGCHI Conference on Human Factors in Computing Systems* (CHI '13), 2157-2166. http://doi.acm.org/10.1145/2470654.2481296
6. S. Hwang and K. Y. Wohn. 2013. VibroTactor: low-cost placement-aware technique using vibration echoes on mobile devices, In *Proc. of the International Conference on Intelligent User Interfaces* (IUI '13), 73-74. http://doi.acm.org/10.1145/2451176.2451206
7. Samuli Hemminki, Petteri Nurmi, and Sasu Tarkoma. 2013. Accelerometer-Based Transportation Mode Detection on Smartphones. In *Proc. of the 11th ACM Conference on Embedded Networked Sensor Systems* (SenSys '13). http://doi.acm.org/10.1145/2517351.2517367
8. Jun Yang. 2009. Toward Physical Activity Diary: Motion Recognition Using Simple Acceleration Features with Mobile Phones. In *Proc. of the 1st ACM International Workshop on Interactive Multimedia for Consumer Electronics* (IMCE '09), 1-10. http://doi.acm.org/10.1145/1631040.1631042
9. M. Rossi, S. Feese, O. Amft, N. Braune, S. Martis, and G. Troster. 2013. Ambientsense: A real-time ambient sound recognition system for smartphones. In *Proc. of the International Workshop on the Impact of Human Mobility in Pervasive Systems and Applications* (PerMoby '13), 230-235. http://dx.doi.org/10.1109/PerComW.2013.6529487
10. M. Azizyan, I. Constandache, and R. Roy Choudhury. 2009. SurroundSense: mobile phone localization via ambience fingerprinting. In *Proc. of the 15th Annual International Conference on Mobile Computing and Networking* (MobiCom '09), 261-272. http://doi.acm.org/10.1145/1614320.1614350
11. H. Lu, W. Pan, N. D. Lane, T. Choudhury, and A. T. Campbell. 2009. SoundSense: scalable sound sensing for people-centric applications on mobile phones. In *Proc. of the 7th International Conference on Mobile Systems, Applications, and Services* (MobiSys '09), 165-178. http://doi.acm.org/10.1145/1555816.1555834
12. Mizell, D. 2003. Using gravity to estimate accelerometer orientation. In *Proc. of IEEE International Symposium on Wearable Computers* (ISWC '03) 252-253. http://dx.doi.org/10.1109/ISWC.2003.1241424
13. Hinckley K., and Horvitz E. 2001. Toward more sensitive mobile phones. In *Proc. of the 14st Annual ACM Symposium on User Interface Software and Technology* (UIST '01), 191-192. http://doi.acm.org/10.1145/502348.502382
14. Ho J., and Intille S. S., 2005. Using context-aware computing to reduce the perceived burden of interruptions from mobile devices. In *Proc. of the SIGCHI Conference on Human Factors in Computing Systems* (CHI '05), 909-918. http://doi.acm.org/10.1145/1054972.1055100
15. F. Wahl, O. Amft. 2014. Personalised phone placement recognition in daily life using RFID tagging. In *Proc. of the 1st IEEE Symposium on Activity and Context Modeling and Recognition*, 19-26. http://dx.doi.org/10.1109/PerComW.2014.6815159
16. Jun Yang, Emmanuel Munguia-Tapia, Simon Gibbs. 2013. Efficient In-Pocket Detection with Mobile Phones. In *Proc. of the ACM Conference on Pervasive and Ubiquitous Computing* (UbiComp '13), 31-34. http://doi.acm.org/10.1145/2494091.2494099
17. I. Diaconita, A. Reinhardt, F. Englert, D. Christin and R. Steinmetz. 2014. Do You Hear What I Hear? Using Acoustic Probing to Detect Smartphone Locations. In *Proc. of the IEEE Symposium on Activity and Context Modeling and Recognition*, 1-9. http://dx.doi.org/10.1109/PerComW.2014.6815157