Energy Consumption in Personal Mobile Devices Sensing Applications

Cristiano G. Pendão[*], Adriano C. Moreira[†], Helena Rodrigues[‡]
Algoritmi Research Center – University of Minho, Guimarães, Portugal 4800–058
[*]cpendao@dsi.uminho.pt, [†]adriano.moreira@algoritmi.uminho.pt, [‡]helena@dsi.uminho.pt

***Abstract*—Personal mobile devices have a strong impact in the daily life of their users, making part of their daily routines. Most of these devices are equipped with several sensors and interfaces that may be used to study human mobility and its interaction with the elements present in physical spaces. Our goal was to develop an application for Android smartphones that could be used to collect data from several of the devices' interfaces (Wi-Fi, GPS, Bluetooth and GSM), and to send that data to a server for later processing and analysis. In order to maximise the autonomy of the devices, energy resources must be used efficiently. This paper focus on a power-consumption saving solution for mobile phone-based sensing systems in the context of human motion analysis. Experiments were conducted with the objective of comparing power-consumption in different situations using our solution. Results have shown that, considering current power consumption patterns, carefully designed solutions for mobile phone-based sensing for observing human motion may enhance energy efficiency satisfactorily. In this particular domain, we have explored periodic sampling of the sensors and the suspension of the sampling process in the Android operating system whenever the device is not moving and we report such results in this paper.**

I. Introduction

The analysis of human movement patterns in physical space, such as in a city, and the interaction of people with objects present in those spaces has high relevance in many domains. Examples of applications are urban and social planning, the study of resources distribution in a city such as public transports, divulgation of advertising and information, prevention of epidemics and the spread of diseases, the response in cases of emergency and terrorist attacks[1], and the monitoring of the urban environment, for example by measuring the levels of noise and pollution[2].

The use of sensor networks have been intensively explored for data collection sensing tasks in urban environments, the so called urban sensing. The first generation of sensor network is composed by static sensors, that allow collecting information about the environment. However, by using static sensors it becomes difficult to cover large urban areas. The costs, considering purchase, installation and maintenance of an adequate number of devices, are a major obstacle to the systems' scalability. It is not possible to place sensorial devices with enough spatial density to obtain samples that allow an overview of the space. Furthermore, sampling becomes restricted to fixed points, being required a complex network infrastructure to gather together collected data for analysis. Due to these limitations, the research focus on sensor networks evolved from static networks to networks that adapt to urban environment dynamics by using an approach centred on people and their mobility[3], leading to architectures based on mobile sensor networks.

A. Collaborative Sensing: Participatory and Opportunistic

Mobile sensing is based on the use of mobile sensors applied to urban environment elements, such as public transport, vehicles, taxi cabs, garbage trucks, or using sensors carried by people, exploiting their mobility in urban areas. The advantages of this approach are numerous, including the smaller number of sensors required to dynamically cover larger areas[1]. A people-centred approach is heavily dependent on cooperation, and this method of sensing is generally referred to as collaborative sensing.

The used sensors can be controlled or uncontrolled. The controlled sensors are devices which route can be easily controlled or predicted, being possible to cover a given area that will allow to complete a sensing task. Uncontrolled sensors are carried by people, applied in automobiles or in other mobile nodes, and their routes cannot be easily predicted. Despite the challenges, the advantages of using uncontrolled mobile sensors have proven to overcome its limitations[4]. The role played by people in collaborative sensing systems has, though, a major impact in this type of systems, both in terms of scalability and diversity of applications that can be supported[3].

In several published studies[3][5][6][7], the authors divide the collaborative sensing in two groups, depending on the methodology used: participatory and opportunistic. In participatory sensing, the user is an active part in collecting data, demanding high levels of explicit interaction. In opportunistic sensing, the participant plays a passive role and the interaction required for data collection is minimal or nonexistent.

B. Advantages of using Personal Mobile Devices

Personal mobile devices (mobile phones, smartphones and tablets) are the mobile sensors that offer the best opportunities for observing human motion. Thanks to rapid technological developments, integrating numerous sensors and communication interfaces, these devices can be exploited for data collecting in urban environment and about population motion. In addition, the high and growing number of users of these devices maximises the number of observed individuals, and may reach millions of devices distributed throughout the world. The participation of people becomes simpler, with no need of carrying specific equipment since participants already use their personal devices daily.

Due to its multiple communication technologies (GPRS, HSPA+, LTE, Bluetooth), these devices are able to send the collected data in a simple way, resulting in lower costs compared to other equipment. Projects developed previously, using mobile sensors applied in cars[8], use unsecured Wi-Fi access points for data transmission. Most of the services available on smartphones require an Internet connection, and most are massively used (e.g. social networking and email), creating communication opportunities for sending collected data. The approach based on personal mobile devices maximises the amount of collected data with no extra costs, neither need for maintenance of the devices. Moreover, sensing tasks are not restricted to a pre-defined time or to the lifetime of fixed sensors. For these reasons, the focus of research has been driven up to mobile sensing, using and exploiting these devices' capabilities.

C. Mobile Sensing Challenges

Consolvo et al.[9] found that users have some reservations considering the cession of their location, wondering why this information is needed. Smartphones are directly linked to the personal lives of its users and are constantly pointing their location, compromising their privacy. Kapadia et al.[6] addresses these problems by describing challenges and discussing possible solutions. Shin et al. presents the AnonySense[10] system, that authors describe as a privacy-aware system for creating applications based on opportunistic collaborative sensing in personal mobile devices. However, the truth is that the privacy of users is practically impossible to guarantee, as shown by recent results from some researches. Movement patterns of an individual can be identified directly through GPS samples, or indirectly, through the Wi-Fi access points and GSM base stations[11].

In some cases, users of a collaborative sensing system allow their personal device to serve the purposes of the system without receiving any type of reward and abdicating some of their privacy. However, to maximise and maintain the number of participants over time, it becomes necessary to define a motivating reward model that will achieve and retain an adequate number of participants. The type of reward can vary from financial rewards to the access to a service. If the reward is attractive, the user may be willing to waive their privacy. Social networking and other services use this type of strategy.

Moreover, there is also a technical challenge that compromises the adherence of people to mobile sensing applications: the use of some interfaces of mobile devices in sensing tasks, such as the GPS, the Wi-Fi, and Bluetooth, involves high energy consumption, with a strong impact on the autonomy of the device. Although devices can be charged by users frequently, the high energy consumption of sensing applications for smartphones immediately condemns them to failure. Users can abdicate their privacy without major reservations as long as the reward is attractive, but not the autonomy of their devices. Users are mostly very addicted to their smartphones, so if an application dramatically reduces the autonomy of the device, making it unavailable or forcing the user to charge it constantly, this application will be uninstalled.

In order to overcome this challenge, we present a solution for reducing energy consumption in mobile sensing applications for smartphones, using algorithms that exploit data from inertial sensors of smartphones.

II. Related Work

LifeMap[12] is an application developed for Android devices, which uses smartphone sensors to generate context. This application uses smartphone's inertial sensors to provide location information in closed environments. This information is combined with GPS and Wi-Fi data to generate user daily life context. The system does not require an infrastructure or costly hardware and uses an event-based technique that reduces power consumption by using a minimal required set of sensors to define the context of a particular situation.

The LifeMap main feature is the ability to determine location accurately, indoors at compartments level of a house or building, creating a context map over the geographic map. The LifeMap application generates user life contexts categorised into four parts: Location: The user's geographic position is specified through the detected Wi-Fi access points information, in order to identify the position accurately at the compartments level; Activity: Defined by the user's movement and smartphone's use; Connectivity: Displays the current network connection status, GSM and Wi-Fi; Surroundings: It is a set of circumstances around the user. The key concept in this project is the ability to record the location in closed environments using inertial sensors and the identical locations aggregation (e.g. within the same compartment) using Wi-Fi hotspots. In the adopted approach, the aggregation process refines the location information based on historical data. To record indoors motion it uses the smartphone's accelerometer and digital compass. When a user is in motion, information is collected from both sensors and is used to determine the direction and approximate the user's current position, being complemented, when possible, by the GPS signal. Users can view detailed information about their points of interest on Google Maps and upload their contexts or download other users' contexts.

Moves[13] is an iPhone application developed by a company called ProtoGeo, which records the users' physical activity and identifies the type of motion, such as walking or running. It also shows a daily routine map and timeline, allowing people to realise their habits. The application does not require users to start or stop the tracking, it constantly runs in the background. The ProtoGeo's goal is that Moves becomes an everyday tool that makes the mainstream population conscientious about their physical activity.

Sampo Karjalainen, Designer CEO of ProtoGeo, said, "*Moves is an example of how smartphones are becoming increasingly context-aware. Today, the mobile phone can constantly learn from its owner's real-time situations and habits. This information can help build better, personalized, end-user apps [...]. As the app is so widely accessible, we are excited by what the technology can offer and how the information provided by Moves has a positive impact in the lives of thousands, or hopefully millions, of users globally.*"[13]

Users don't need to buy an additional device to use the application, also it uses adaptive techniques to minimise energy consumption, using, most of the time, data from the cellular network, activating the GPS when detecting a known type of movement by the accelerometer.

Despite the used techniques, on the application's website users are cautioned for the battery consumption, recommending that smartphones should be charged overnight. Tests on a normal day of using an iPhone 4S with the Moves application running constantly in the background, leads to only 14h of autonomy. As observed by Liz Gannes, a Moves' user, the application has an impact on battery life, but not as much as constant GPS tracking[13].

To address the problem of power consumption efficiency of mobile sensing systems, in particular sensing of positioning data from GPS, Wi-Fi, GSM or Bluetooth, several works explore user movement detection using the accelerometer in order to decrease sensors sampling rate[14][15][16][17]. None of these works directly addresses the problem of smartphone-based sensing in the context of human motion analysis. In particular, in [14], the authors describe, implement and evaluate a sensors' power management system for mobile devices in the context of human state recognition. The system receives as input a description of all the states to be automatically classified, as well as sensor management rules for each state. Device battery life is improved by powering only a minimum set of sensors depending on the user's state and using appropriate sensor duty cycles. A battery's lifetime test was conducted that proved the system to significantly reduce the device energy consumption. Although it presents very interesting results on power consumption, this system focus on human activity recognition which involves a different set of requirements on it concerns sensing parameterisation and on the used set of sensors and thus it is not directly comparable to sensing systems aiming to gather data about human mobility.

Alternatively, SensLoc [15] is a mobile location service that comprises a robust place detection algorithm, a sensitive movement detector and an on-demand path tracker. SensLoc was evaluated using different data sets collected from both real-life and scripted tours and has shown to consume significantly less energy than algorithms that periodically collect location data to provide the same information.

It is clear that the research community is well aware of the energy-efficiency problem in sensing activities using smartphones, but experimental results about the real-world impact on energy consumption of the proposed solutions are still limited. In this paper we address some of the variables affecting the energy consumption in sensing of human mobility and evaluate their impact through extensive real-world experiments.

III. The Power Consumption Problem

The battery's lifetime depends on its capacity, the smartphone usage level and the hardware consumption. The chart in Figure 1 presents the consumption[1], in mA, of the main interfaces on a smartphone.

It is noticeable through the table I and Figure 1 that among the interfaces used in sensing tasks, the GPS, Bluetooth and Wi-Fi, are those which represent the most significant impact in the devices' autonomy, increased by consumption of the CPU required to use these interfaces. In Wi-Fi's case, there is energy consumption even in idle mode.

[1]Values measured using a industrial power monitor with 5 kHz sampling rate, using the average power with lower standard deviation.

	Interface			
Mode	Wi-Fi	GPS	Bluetooth	Mobile Network
Idle	12 mA	0 mA	6 mA	Not considered
Full	275 mA	85 mA	50 mA	Not considered

Table I. Consumptions per interface

The values for Wi-Fi and GPS presented in table I were obtained from a presentation made by Jeff Sharkey, a Google's software engineer, in 2009 at Google IO conference[18]. Bluetooth's consumptions were obtained from Bluetooth v2.1 Class 2 modules' data sheets, similar to those used in most smartphones. Usage consumptions of mobile network interface (GSM, UMTS, EDGE, ...) are not considered in this study since the energy consumption of these interfaces is not affected by the sensing tasks.

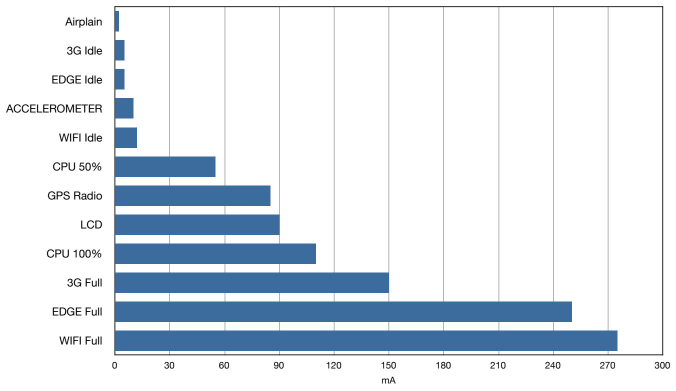

Figure 1. Electric charge used in a smartphone per unit of time[18]

A. *Interfaces' Sampling: Continuous and Periodic*

The interfaces' sampling can be performed continuously or periodically. In continuous mode the sampling tasks are executed again as soon as the previous task returned the result, while in periodic mode the execution of the sampling tasks are controlled, for example, by timers.

Most of sensing applications do not require a continuous sampling to achieve their purposes. In sensing tasks aiming to observe the motion of humans, data is useful if it allows analysing people motion over time in a certain physical space. Therefore, reducing the interfaces' use, through periodic sampling, it is possible to reduce the power consumption. This approach involves setting a sampling period for each interface, in order to maintain the efficiency of data collection.

Defining the sampling periods requires the understanding of which factors may influence the data collection efficiency. The interfaces differ in characteristics, such as energy consumption, access method and operation range.

The operation range of GSM is much higher than Wi-Fi's, therefore the sampling period for GSM should be longer, but the energy consumed by the Wi-Fi is much higher. Other aspect to take into account is the time that a scan takes to be completed. A Bluetooth scan, for example, takes about 13 seconds to be completed, therefore using a sampling period shorter than 13 seconds is inefficient.

Considering the interfaces' characteristics, sampling periods have been defined in order to estimate consumption of an approach based on periodic sampling.

Taking into account the energy consumption values shown in Figure 1 and table I, the impact of continuous and periodic sampling on the smartphone's battery life has been estimated. We obtained the values represented in the figure 2, which reflect the consumption after 60 minutes. It is considered a battery with 1200 mAh of capacity. The objective is to estimate only the interfaces consumption, therefore the CPU's consumption discarded, and the LCD has been considered to be turned off during the process.

The estimative suggests that continuous sampling completely drains the battery (1200 mAh) in about 3 hours, confirming that a continuous sampling is not practicable. Comparing with periodic sampling, we observe a significant energy consumption reduction, which is reflected in the battery lifetime of about ≈10 hours. However, despite the significant reduction, the energy consumption remains at unacceptable levels.

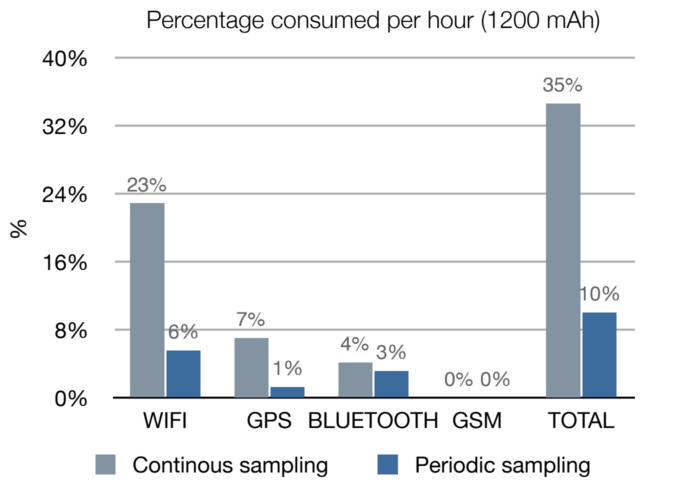

Figure 2. Estimation of the battery percentage consumed per hour in continuous and periodic sampling. Sampling Periods: Wi-Fi 10 seconds; GPS 60 seconds; Bluetooth 20 seconds; GSM 20 seconds;

IV. Our Solution

Periodic sampling reduces the interfaces' use in a linear way over time. In our approach in addition to reduce the sampling periods we intend to stop data collection when unnecessary. According to the results obtained in other studies, on average people are in motion for less than 20% of the time, each day[12][19]. Therefore, by keeping the data collection at a minimum level during the remaining ≈80% of the day, a significant reduction on energy consumption may be achieved. Considering the users daily motion average time (20%) and the consumption rate of the periodic sampling (10.1% of the battery charge per hour) we estimate that the autonomy of the battery can be extended up to 50 hours[2].

The proposed energy saving solution exploits the two aspects described above: periodic sampling of the sensors accordingly to the type of phenomenon being observed (in our case is human mobility), and the suspension of the sampling process whenever the device is not moving. Figure 3 depicts the general architecture of the system proposed to minimise energy consumption. A prototype of this system has been implemented as an Android application.

[2]Not considering the consumption arising from the normal operation and use of the smartphone.

For controlling the sampling periods we use a set of timers associated to each interface, and set the sampling period accordingly to the characteristics of each interface. For motion detection, we use the accelerometer which requires lower energy consumption than any other interface, as shown in Figure 1, being therefore an excellent tool to control the data collection process.

Additionally it was necessary to address a challenge related to the fact that the Android operating system suspends all processes running when the smartphone enters standby mode in order to save energy when the device is not being used. However, for sensing applications this becomes a critical problem as it interrupts the sensing tasks making it impossible to perform any function or use the smartphone's interfaces and sensors for a long period of time.

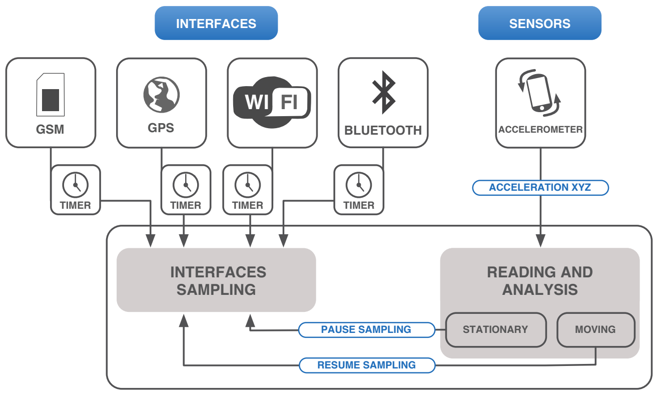

Figure 3. Data collection controlled by motion detection through the analysis of accelerometer data

A. *Maintaining the Sensing Tasks Active*

The solution for enabling long term sensing tasks involves the use of the WakeLock of the Android's PowerManager Class. The WakeLock allows an application to maintain control over the state of the device, and can keep resources active even when the phone is not being used by the user.

However, using the WakeLock, even in partial mode, involves high energy consumption. The Android's documentation itself warns the developers about this situation. The consumption depends on the type of WakeLock used and the time using it. Thus, it is recommended that it be acquired while the task is being executed and then released.

Sensing tasks are periodic and must be performed while motion exists. If the WakeLock is released after the first execution, the CPU becomes inactive and the following tasks can not be executed. If, on the other hand, the WakeLock is not released, the continuous CPU operation will have a tremendous impact on the device's autonomy, because it constantly consumes energy and other processes take advantage of the CPU activity to perform their own tasks.

To minimise this problem, we used the motion detection algorithm combined with an alarm system. If motion is not detected for a certain period of time, the sensing tasks are suspended and the WakeLock is released, allowing the CPU to enter into idle mode. However, when the CPU is idle, it is not possible to detect motion to restart the sensing tasks. Therefore, we implemented an additional mechanism that uses the Alarm Manager Class. This class provides access to the alarms system service and we can schedule a periodic execution of a particular method of our application. When the alarm goes off, the Intent[3] that has been registered for the event is broadcasted by the system and our service is able to restart the motion detection algorithm and the sensing tasks.

Through the use of the alarm, we can activate the CPU periodically (e.g. every 2 minutes), acquire a WakeLock to keep the CPU active, collect a few samples from the accelerometer and analyse them to evaluate the state of motion. If no motion is detected, the WakeLock is released immediately and the CPU goes back into idle mode. If motion is detected in the pre-analysis, the motion analysis algorithm enters in a continuous mode and, while motion is detected, the CPU is kept active as well as the regular sampling tasks. Later, if absence of motion is detected for a certain period of time (e.g. 2 minutes), the WakeLock is released and the CPU returns to idle mode until the alarm goes off again. The operation of the described algorithm is depicted in Figure 4.

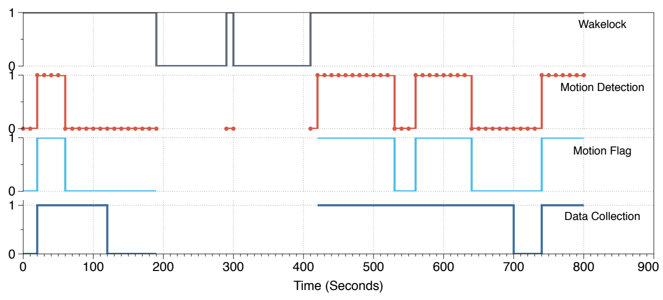

Figure 4. Motion detection algorithm states

B. Motion Detection

The proposed motion detection solution using the accelerometers' data is relatively simple, the challenge being how to distinguish between the merely manipulation of the device from the motion of its user. Our goal is not to classify or contextualise the type of user's motion, but only to detect whether the user is moving or not (walking, moving on a bicycle, car, bus, etc.). The accelerometers integrated into smartphones are very sensitive devices, detecting very small differences between the linear acceleration of the device and the gravitational field. Sampling the accelerometer's programming interface returns three values corresponding to the acceleration on the three axes (X, Y, and Z), and measured in m/s^2. These values are positive or negative, depending on the movement of the smartphone. Ideally, for a phone placed on a levelled surface, the readings would be zero in X and Y, and -9.81 in Z (acceleration of gravity). However, in practice, the readings are affected by noise and, due to the high sensitivity of the sensor, also by very light vibrations. Even the slightest movements results in significant variations in the sampled values.

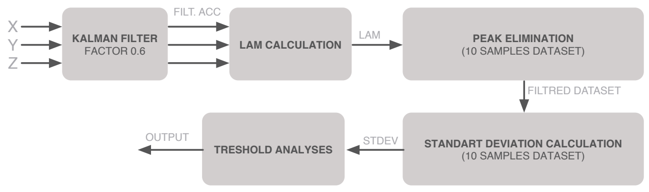

Figure 5. Motion detection algorithm block diagram

Our solution to distinguish between real motion of the user and other situations is depicted in Figure 5. First, readings obtained at 1 sample/second, from the three axes, are fed into a Kalman filter aiming to equalise the different sensibilities of the sensor on the three axes and to eliminate the their bias. This filter generates a sequence of filtered values, at 1 sample/second, calculated based on a set of previous samples. These sequences are then used to calculate the Linear Acceleration Magnitude (LAM). The LAM is represented by a single non-negative value that represents the magnitude of the acceleration vector. In our experiments we observed the temporal variation of the LAM (see Figure 6) and found that it exhibits large peaks and large variations while a user is in motion. We then resorted to an additional filter for removing extremely large values (outliers) before calculating the standard deviation of each sequence of 10 samples. Whenever the standard deviation is lower than a predefined threshold, absence of motion is declared. The output of this sub-system is a "motion indicator" value updated every 10 seconds (see Figure 6).

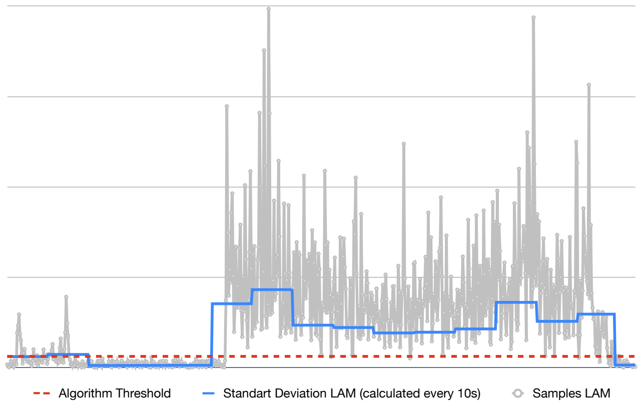

Figure 6. Magnitude Values of the accelerometer's linear acceleration of the walking motion

Setting the adequate threshold for distinguishing motion from other situations required the realisation of a few real-world experiments. Several test with a person walking with the smartphone on his pocket showed that the standard deviation of the LAM is, on average, around 0.43 m/s^2, in line with results obtained in other studies[12]. Therefore, setting the threshold to a value a little lower than this value would allow

[3]An intent is an abstract description of an operation to be performed. It can be used with startActivity to launch an Activity, broadcastIntent to send it to any interested BroadcastReceiver components, and startService(Intent) or bindService(Intent, ServiceConnection, int) to communicate with a background Service.

us to detect the presence/absence of walking motion. However, subsequent experiments with smartphone users traveling by car revealed that, on average, the standard deviation of the LAM is lower than while walking. We observed values around 0.35 m/s^2 for a smartphone placed somewhere inside the car, and values around 0.22 m/s^2 for a smartphone placed inside the driver's pocket. These experiments helped us in setting the threshold to a lower value (we used 0.20 m/s^2) in order to minimise the probability of false positives (to incorrectly detect the absence of motion). While lower values might increase the probability of false negatives, it minimises the probability that data collection is suspended while a user is actually in motion.

V. Results and Evaluation

We conducted a set of experiments to evaluate how the proposed solution impacts the energy consumption on a smartphone used in a collaborative sensing task. The experiments used two different Android smartphones running an application specially developed for this purpose, as part of our platform for collaborative sensing. This application runs as a service in the background and periodically collects data from 5 interfaces: Wi-Fi, GPS, GSM/UMTS, Bluetooth, and Battery level. The first four interfaces are sampled periodically with periods of 10, 60, 20, and 20 seconds, respectively. The battery level interface is sampled based on a native mechanism of the Android operating system (the operating system broadcasts a notification whenever the battery levels changes). Table II summarises the major characteristics of the used smartphones. These smartphones were restored to factory settings before the tests, and no other applications have been installed besides our own. Every test performed (18 in total) began at the same time (00h 00m 00s) of each day, in days with similar motion pattern of the same user.

Characteristics	Samsung Galaxy Mini S5570	Samsung Galaxy Ace S5830
Factory Battery Capacity	Li-Ion 1200 mAh	Li-Ion 1350 mAh
CPU	600 MHz ARMv6	800 MHz ARM 11
Bluetooth	v2.1 with A2DP	v2.1 with A2DP
Wi-Fi	Wi-Fi 802.11 b/g/n	Wi-Fi 802.11 b/g/n
Android OS Version	2.2.1	2.3.6

Table II. Characteristics of the used smartphones

For each one of the smartphones, three sets of tests were performed. First, the energy consumption of each smartphone was measured without any sensing task running. We note that these baseline tests were conducted with Wi-Fi, GSM/UMTS and Bluetooth interfaces being managed by the Android OS and with the smartphones being carried by its user as usually. However, the smartphones have not been used to place or receive phone calls, send or receive short messages, or to access the Internet. This represents the best case in terms of the smartphone's autonomy since no energy has been used for collecting data or other tasks, except for logging the battery level. We name these tests "baseline".

The second set of tests was performed while running our data collection application, but without using the motion detection algorithm. Note that the data collection task includes the upload of the collected data to a remote server whenever there is access to the Internet through a WiFi network. All other parameters were the same as for the baseline tests. We name these tests "without motion algorithm".

The third set of tests was performed while running the motion detection algorithm, all other conditions being the same as for the "without motion algorithm" tests. We name these tests "with motion algorithm". Figures 7 and 8 show the results obtained for each one of the smartphones.

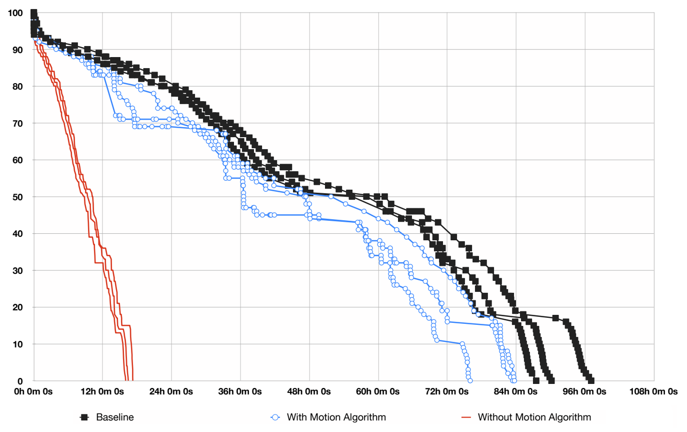

Figure 7. Battery drain with and without motion algorithm (Galaxy Ace)

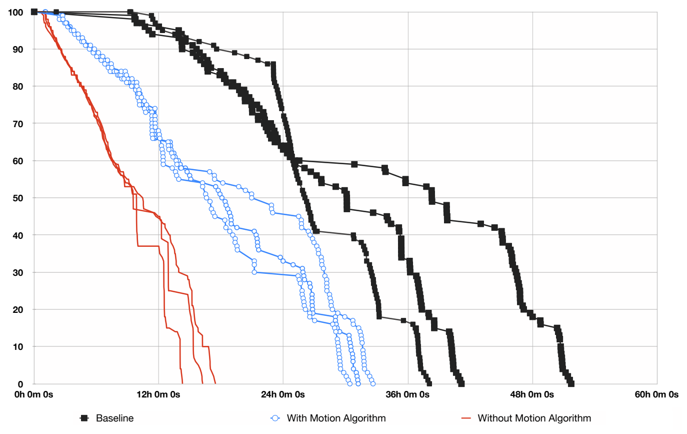

Figure 8. Battery drain with and without motion algorithm (Galaxy Mini)

As expected, collecting data without running the motion detection algorithm results in a strong penalty on the devices' autonomy when compared with the "baseline" tests. The results obtained during the "with motion algorithm" tests in both smartphones indicate a significant gain on battery autonomy when compared with the results obtained during the "without motion algorithm" tests.

Based on results obtained we calculated for both smartphones the average autonomy time (in hours) for each set of tests (values (a), (b) and (c) in table III). This values were calculated for time up to which the battery charge dropped to 10%, because when battery charge is low the draining patterns are highly irregular.

	Galaxy Ace	Galaxy Mini
Average autonomy - “Baseline” **(a)**	89,2 h	42,6 h
Average autonomy - “With motion algorithm” **(b)**	79 h	30,5 h
Average autonomy - “Without motion algorithm” **(c)**	16,3 h	15,4 h
Autonomy reduction compared with “Baseline” **(a) – (b)**	10,25 h	12,1 h
Autonomy gain compared with “Without motion algorithm” **(b) – (c)**	62,7 h	15,1 h
Baseline test average power consumption **(d)**	55,97 mW	104,23 mW
With motion algorithm test average power consumption **(e)**	63,23 mW	145,57 mW
Power consumption of our solution **(e)-(d)**	7,26 mW	41,34 mW

Table III. AVERAGE AUTONOMY AND AVERAGE POWER CONSUMPTION

As table III shows, using the motion detection algorithm for smartphone A (Galaxy Ace), we have obtained a battery autonomy reduction of approximately 10,2 hours comparing to the result obtained on “baseline” tests and a battery autonomy gain of approximately 62,7 hours comparing to the results obtained during the “without motion algorithm” test. In the same situation, for smartphone B (Galaxy Mini), the battery autonomy reduction was of 12,1 hours and the battery autonomy gain was of 15,1 hours.

As noted before, the motion patterns along the different days were very similar. In respect to that, we can observe, for each set of “with motion algorithm” tests, very similar battery consumption curves. We may perceive some small variations due to normal variations on movement patterns during the day (such as changing the place to have lunch or having an extra visit to the library). We may then observe that motion patterns have a direct influence on the variation of the consumptions over time, which corroborates our initial assumptions.

Using the data collected during the experimental tests, we also estimated the overall consumption of our sensing solution. We estimated the overall average power consumption in the "baseline" and "with motion algorithm" tests. Those values ((d) and (e) in table III) were estimated using the average autonomy of each set of tests (values (a) and (b) in table III) and the total capacity of the batteries present in table II. The values in the last line in table III correspond to the average power consumption of our sensing solution.

It is important to notice that although our solution’s average power consumption values differ greatly from mobile phone Galaxy Ace to mobile phone Galaxy Mini, the average autonomy loss is similar. The average power consumption difference may be due to the method for estimating power consumption as it depends, in our case, on the capacity of the battery, which may not be at the maximum level due to battery aging. Hardware characteristics in general, and, possibly, operating system versions, may also influence battery consumption curves. Additional tests must be therefor executed to uncover this correlation.

We have also compared our results against SensLoc system presented in [15] (see section II). In this work, the authors have estimated the overall energy used by SensLoc in two configurations: with path tracking disabled, and place and movement detection enabled; and with path tracking enabled. In the first configuration, SensLoc uses a sampling interval of 10 seconds for Wi-Fi and a duty cycle of the accelerometer of 50% (over a 10 seconds period) that provided information for detecting movements to trigger Wi-Fi scans. In the second configuration, SensLoc also samples GPS coordinates every 10 seconds when the user is traveling between places. SensLoc used 32,8mW and 54,8mW on average without path tracking and with path tracking, respectively.

The comparison between SensLoc and our system cannot be comprehensive, as there exist some differences between SensLoc’s and our solution’s implementations. However, SensLoc resembles our system insofar as its central sensing tasks are focused on location information. In particular, we may evidence that our solution makes additional samples of Bluetooth and GSM interfaces. On the other hand, SensLoc’s global sampling period is 10s, both for Wi-Fi and GPS interfaces, while our solution adopts a GPS sampling period of 60s while still keeping a Wi-Fi sampling period of 10s. Moreover, our solution uploads the sample raw data to a server in the Internet. In its turn, SensLoc locally computes the raw data to estimate track patching and place detection.

To estimate the power consumption, the authors logged the time each sensor was activated and used the average power consumption of each sensor (obtained from power measurements of an HTC G1 phone), and thus, the data provided does not corresponds to data effectively obtained from experimental usage in real settings which also would include consumption from different hardware components, such as CPU. The overall consumption of our solution (for mobile phone Galaxy Ace) is 7,26 mW on average which is nearly 22% of what SensLoc used when path tracking is disable (32,8mW) and 13% of what SensLoc used when path tracking is enable (54,8mW) (the percentages for Galaxy Mini are $\approx$ 126% and $\approx$ 75%). These results point out for a better energy-efficient behaviour from our solution, which we consider to be explained by a careful motion detection algorithm design as well as careful engineering process, in this particular situation, within the Android operating system. The power consumption gains are apparently more limited for the Galaxy Mini smartphone. This could indicate that a comparison based in observed smartphone’s autonomy time could be more significant, but SensLoc evaluation does not provide sufficient data to estimate the real autonomy of SensLoc system.

Generally, we should state that periodic sampling of smartphone’s sensors, using long sample periods as longer as it does not disrupted application quality and performance, using adequate algorithms to detect movement in order to suspend or to soften the sampling process, exploring conveniently operating systems services are promising guidelines for designing mobile sensing solutions.

VI. CONCLUSION AND FUTURE WORK

In this paper, we describe a power-consumption saving solution for mobile phone-based sensing systems in the context of human-motion analysis. We have explored periodic sampling of the sensors and the suspension of the sampling process whenever the device is not moving. We have conducted a set of experiments that have shown a significant improvement in the devices' autonomy during the execution of sensing tasks in the context of human-motion analysis. Our experiments involved a user performing his normal daily tasks and motion patterns.

The balancing between sensing tasks effectiveness and the energy consumption is possible in the context of human-motion analysis as suspending the sampling process during periods of immobility results in energy saving without compromise the collected data. Daily application consumption is however still dependent on the users' movement patterns during the day.

The Operating System model and APIs may significantly influence energy saving solutions and the efficiency of sensing tasks. In particular, pausing the CPU's running processes during the periods in which a smartphone is not being used has been shown that it constrains the implementation of sensing applications.

Recently (on September 2013), Apple unveiled the new M7 chip that integrates the company's latest smartphone, the iPhone 5S. This coprocessor has the function of constantly measure motion data (accelerometer, gyroscope and compass), that can be used by fitness and health applications, without waking up the main processor. This coprocessor reduces the load of tracking motion using the main processor, and requires significantly less battery. It also offers ways for the system to make power management more efficiently, making the iPhone more intelligent in terms of when to activate or to deactivate certain features to conserve battery life. The M7 also uses a new API named "CoreMotion" to identify the type of user's movement and makes optimisations based on it[20]. In the particular case of Android, and at least until most of the smartphones have a solution similar to the M7 Chip on iPhone 5S, mechanisms that uses a combination of the Android's Alarm Manager and Wakelock can be used to overcome this situation, as we have shown in this paper.

As future work, we should perform a more exhaustive study about the sampling periods for each interface, as these sampling periods greatly influence the correlation between the energy consumption and the sensing tasks efficiency. Further research should be done on motion detection algorithm in order to detect user's motion more accurately, for example, by detecting different types of motion, and using the environment analysis to dynamically adjust the threshold value. Our next step is to perform tests to extract data which represents in a more explicit manner the current algorithm's motion detection accuracy, despite being perceptible through the battery consumption patterns presented.

Acknowledgment

Research group supported by FEDER Funds through the COMPETE and National Funds through FCT Fundação para a Ciência e a Tecnologia under the projects n. 13843 and PEst-OE/EEI/UI0319/2014.

References

[1] T. Zhang, S. Madhani, and E. van den Berg, "Sensors on patrol (sop): using mobile sensors to detect potential airborne nuclear, biological, and chemical attacks," in *Military Communications Conference, 2005. MILCOM 2005. IEEE*, pp. 2924 –2929 Vol. 5, oct. 2005.

[2] T. Abdelzaher, Y. Anokwa, P. Boda, J. Burke, D. Estrin, L. Guibas, A. Kansal, S. Madden, and J. Reich, "Mobiscopes for human spaces," *IEEE Pervasive Computing*, vol. 6, pp. 20–29, 2007.

[3] N. D. Lane, S. B. Eisenman, M. Musolesi, E. Miluzzo, and A. T. Campbell, "Urban sensing systems: opportunistic or participatory?," in *Proceedings of the 9th workshop on Mobile computing systems and applications*, HotMobile '08, (New York, NY, USA), pp. 11–16, ACM, 2008.

[4] S. Madhani, M. Tauil, and T. Zhang, "Collaborative sensing using uncontrolled mobile devices," in *Collaborative Computing: Networking, Applications and Worksharing, 2005 International Conference on*, p. 8 pp., 0-0 2005.

[5] J.-S. Lee and B. Hoh, "Sell your experiences: a market mechanism based incentive for participatory sensing.," in *PerCom*, pp. 60–68, IEEE Computer Society, 2010.

[6] A. Kapadia, D. Kotz, and N. Triandopoulos, "Opportunistic sensing: Security challenges for the new paradigm," in *Communication Systems and Networks and Workshops, 2009. COMSNETS 2009. First International*, pp. 1 –10, jan. 2009.

[7] "Participatory urbanism participatory urbanism: Empowering citizens to collectively author, share, and remix measurments from their environment." http://www.urban-atmospheres.net/ParticipatoryUrbanism/index.html, 2010. Last check at April 10 2012.

[8] B. Hull, V. Bychkovsky, Y. Zhang, K. Chen, M. Goraczko, A. Miu, E. Shih, H. Balakrishnan, and S. Madden, "Cartel: a distributed mobile sensor computing system," in *In 4th ACM SenSys*, pp. 125–138, 2006.

[9] S. Consolvo, I. E. Smith, T. Matthews, A. LaMarca, J. Tabert, and P. Powledge, "Location disclosure to social relations: why, when, & what people want to share," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '05, (New York, NY, USA), pp. 81–90, ACM, 2005.

[10] M. Shin, C. Cornelius, D. Peebles, A. Kapadia, D. Kotz, and N. Triandopoulos, "AnonySense: A system for anonymous opportunistic sensing," *Journal of Pervasive and Mobile Computing*, vol. 7, pp. 16–30, February 2011.

[11] M. C. Gonzalez, C. A. Hidalgo, and A.-L. Barabasi, "Understanding individual human mobility patterns," *Nature*, vol. 453, pp. 779–782, June 2008.

[12] J. Chon and H. Cha, "Lifemap: A smartphone-based context provider for location-based services," *Pervasive Computing, IEEE*, vol. 10, pp. 58 –67, feb. 2011.

[13] ProtoGeo, "Moves for iPhone." http://protogeo.com/, 2013. Last Check at May 1 2013.

[14] Y. Wang, J. Lin, M. Annavaram, Q. A. Jacobson, J. Hong, B. Krishnamachari, and N. Sadeh, "A framework of energy efficient mobile sensing for automatic user state recognition," in *Proceedings of the 7th International Conference on Mobile Systems, Applications, and Services*, MobiSys '09, (New York, NY, USA), pp. 179–192, ACM, 2009.

[15] D. H. Kim, Y. Kim, D. Estrin, and M. B. Srivastava, "Sensloc: Sensing everyday places and paths using less energy," in *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*, SenSys '10, (New York, NY, USA), pp. 43–56, ACM, 2010.

[16] J. Paek, J. Kim, and R. Govindan, "Energy-efficient rate-adaptive gps-based positioning for smartphones," in *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services*, MobiSys '10, (New York, NY, USA), pp. 299–314, ACM, 2010.

[17] Z. Zhuang, K.-H. Kim, and J. P. Singh, "Improving energy efficiency of location sensing on smartphones," in *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services*, MobiSys '10, (New York, NY, USA), pp. 315–330, ACM, 2010.

[18] J. Sharkey, "Google IO 09: Coding for Life – Battery Life, That Is." http://google.com/events/io/2009/sessions/CodingLifeBatteryLife.html, 2009. Last check at September 4 2012.

[19] I. Constandache, R. Choudhury, and I. Rhee, "Towards mobile phone localization without war-driving," in *INFOCOM, 2010 Proceedings IEEE*, pp. 1 –9, march 2010.

[20] "iPhone 5S Features." http://www.apple.com/iphone-5s/features/, 2013. Last check at September 25 2013.