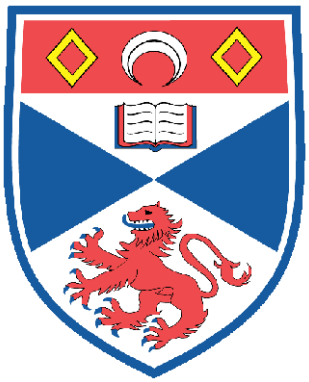
UNIVERSITY OF ST ANDREWS
SCHOOL OF COMPUTER SCIENCE



Locy: Energy-efficient sensing with Android smartphones.

Author:
Martin Maciej Kukla
mailto:mk74@st-andrews.ac.uk

Supervisor:
Dr Tristan Henderson
mailto:tnhh@st-andrews.ac.uk

April 4, 2014

Abstract

Phone sensing can be utilized by mobile applications to provide advanced services such as navigation systems. Phone sensing fetches raw sensor data (e.g. from an accelerometer) and tries to extract high-level information from it (e.g. a user is walking). Such a process may have high energy demands, which is crucially important to mobile phone users. To alleviate this problem, energy-efficient phone sensing methods must be investigated.

This project investigates the energy-efficiency of phone sensing. It establishes the energy efficiency of all sensors available across three different devices. The results are used to design Locy, an energy-efficient localisation library for Android, which is easy to plug into existing applications. When possible, Locy replaces the high-power GPS receiver with readings from the energy-efficient accelerometer. Its algorithm adapts to the remaining battery life of a device. Locy's energy efficiency is evaluated in two real-life scenarios. The library provably outperforms a standard Android localisation implementation.

Additionally, a new method for energy efficiency measurement is introduced and verified. This is believed to be the first study presenting complete results of sensors' energy efficiency across different devices. The results may be further leveraged for different research purposes. Finally, Locy provides a GPS localization service in an energy-efficient manner, which may be of great interest to developers wishing to increase the utility of their mobile applications.

Declaration

I declare that the material submitted for assessment is my own work except where credit is explicitly given to others by citation or acknowledgement. This work was performed during the current academic year except where otherwise stated.

The main text of this project report is 12,396 words long, including project specification and plan.

In submitting this project report to the University of St Andrews, I give permission for it to be made available for use in accordance with the regulations of the University Library. I also give permission for the title and abstract to be published and for copies of the report to be made and supplied at cost to any bona fide library or research worker, and to be made available on the World Wide Web.

Martin Maciej Kukla

Contents

List of Figures

1 Introduction

Smartphones are widely used devices, which provide **advanced services based on a user's context**. Localization is the most utilized context information: Google Maps [23] and Apple Maps [29] are popular navigation systems exploiting a user's geographical coordinates; Yelp [53] recommends a restaurant near a user; Facebook [14] tags pictures with a user's geolocation and Foursquare [17] is location-based social media. Beside localization, there are other relevant context information available on smartphones. Recently, the Apple's patent was granted for variable device graphical user interface [44]. The patent describes the interface that is adjusted depending on what kind of phone motion is detected e.g. icons are bigger while a user is running. Motion detection is also utilized in health mobile applications. Accupedo Pedometer [18] monitors a user's daily walking to estimate his calories burned. Furthermore, research community investigates other context information e.g. speech detection and Bluetooth proximity may help in quantifying social interactions [46]. Lastly, the context may redefine mobile operating systems in future. The functionality of a mobile phone may be adaptive e.g. a user may not be interested in bus timetable if he is driving a car. Google Now [24] tries to leverage that context information to provide a user with "the right information at just the right time".

The phone sensing process infers the context. The process consists of acquiring raw sensor data and extracting a relevant information out of them. For example, accelerometer data may indicate that the user is not moving. Modern mobile phones are rich in sensors e.g. Google Nexus 7 is equipped with standard sensors (i.e. microphone, camera, IEEE 802.11, GPS, Bluetooth or accelerometer), magnetic field, gyroscope and ambient light. Those sensors may be sampled and provide raw sensor data from which meaningful information could be obtained. This process usually involves feature extraction and classification. Different features may be used depending on a sensor and a type of information needs to be established e.g., the number of peaks in accelerometer data may be a good predictor to answer whether an user is walking or running. The classification is an inference, which assigns a high-level class to feature vector e.g., 2 peaks in one second may be interpreted as a user is walking.

Battery life is of key importance to mobile phone users. Modern mobile applications provide complex functionality, which results in high energy demands. Without any significant progress in battery hardware technology, energy consumption is one of the most crucial problems in mobile systems. Furthermore, there are global projects aspiring to provide mobile internet in developing countries e.g. Internet.org [15]. For those projects, the energy efficiency of a mobile phone is even more disturbing issue since a user's opportunity of charging a device is often limited.

Phone sensing may lead to fast battery depletion. Both phases of phone sensing incur additional energy costs. Simple physical sensor are energy-efficient themselves, but their careless sampling may result in high energy consumption. For example, while accelerometer is being sampled, high-power components including CPU are active which may significantly drain the battery life [45]. For other sensors like camera or GPS, the sampling itself is an expensive operation [6]. Also, the data extraction process is computationally intensive, which means high energy demand [41].

In mobile phone sensing, there is a specific energy-accuracy tradeoff. Energy efficiency of mobile sensing may be improved while decreasing the accuracy of the context information. For example, we could sense GPS less often. This would result in energy savings, but the location coordinates could be inaccurate during the periods when GPS is not being sampled.

This project investigates the energy efficiency of phone sensing. It proposes an unique energy measurement method, which is validated in the series of experiments. As a result of those experiments, the energy efficiency levels of all sensors available across three different devices are established. It is believed to be a first study of this type. Basing on that study, the energy-efficient sensing library is built, Locy. It leverages energy-efficient accelerometer to switch off high-power GPS, when it is not needed. The library's performance is evaluated in three real-life scenarios.

The dissertation is structured in the following way: First, the objectives are explicitly stated (Section 2). In Section 3, the context survey of the field is undertaken. Then, software requirements and software engineering processes are described (Sections 4 and 5) . The Section 6 presents the short description of ethical considerations. Subsequent sections are split into two threads: Sensor Energy Measurements and Locy. The former is focused on determining the energy efficiency levels of the sensors, whereas the latter describes the energy efficient sensing library. The design, implementation and evaluation of both are presented in Sections 7, 8 and 9. Lastly, the Section 10 demonstrates the project's conclusions.

2 Objectives

This project aims to investigate the energy efficiency of phone sensing. It sets out to review the current state of the art and identify possible enhancements. I intend to establish energy efficiency levels of different sensors in the experiments. The results of those experiments are planned to be leveraged for the design of an energy-efficient sensing library. This library should be implemented and evaluated in real-life scenarios. More specifically, the project has the following objectives:

1. **Primary objectives**
   (a) To review the current state of the art in sensing, energy measurement methods and energy-efficient sensing.
   (b) To determine the energy efficiency of different sensors.
   (c) To build an energy-efficient sensing library.
   (d) To evaluate the energy-efficient sensing library.
2. **Secondary objectives**
   (a) To identify possible enhancements in energy measurement methods and energy-efficient sensing.
   (b) To investigate the possibility of online energy measurement.
   (c) To compare the energy measurements across different devices.
   (d) To evaluate the energy-efficient sensing library across different devices.
   (e) To develop an energy-efficient strategy adaptive to a device's battery life.
   (f) To optimize energy efficiency in the case of applications contending for sensor usage.
3. **Tertiary objectives**
   (a) To develop a new technique for energy-efficient sensing.

3 Context survey

This section critically reviews the project's related work (Objective 1a). First, it studies the capabilities of phone sensing. Second, it explores different energy measurement methods and presents other research's energy measurement results. Finally, the approaches to energy-efficient phone sensing are studied. The section also tries to identify possible improvements in each of those areas (Objective 2a).

3.1 Phone sensing

Although physical sensors are noisy and often inaccurate, meaningful information may be extracted from them. Sensor-based **activity recognition** aims to determine a user's physical activity e.g. sitting, walking or driving. Most of this activity recognition is based on accelerometer data, though other sensors may be utilized (e.g. compass). As a part of Google Play Services, an activity recognition service is provided in Android: AcitivityRecognitionClient [1]. At the moment, the service is capable of detecting five different types of activities:

- a user is in vehicle.
- a user is on bike.
- a user is on foot.
- the user is still.
- the mobile phone is tilting.

ActivityRecognitionClient delivers the list of activities with the corresponding probabilities of them taking place at a given moment. The service requires Internet connectivity to work, since Google Play Services do. Furthermore, the research community has also established algorithms for detecting other activities:

- a user is taking an elevator up or down [51].
- a user is ascending or descending a staircase [51].
- a user is on escalator [51].
- a user is running [40].
- a different position of a phone (e.g., chest pocket or trousers pocket) [33].

Activity recognition algorithms are still in their early days. Some studies propose different algorithms for recognizing the same activity (e.g., walking [6] [46] [37]). Furthermore, implementations of the proposed algorithms arbitrarily select different parameters: threshold values (e.g., if a standard deviation is more than 30, then a user is walking) and window size (e.g., classify weather a user is moving over the period of two seconds). Lastly, the algorithms are usually evaluated on a limited amount of devices with few users, who hold their phones in constrained positions (e.g. trouser pocket).

Activity recognition may be leveraged for **navigation**. Detected activities may assist in localizing a user e.g., if a user is taking an elevator and there is one elevator nearby, he can be easily localized. Furthermore, physical sensors may provide useful

information for navigation: the speed of user's movement (accelerometer) and their direction (compass and gyroscope). Such information may be utilized to estimate a user's relative position. For example, take a user moving north in straight line with a speed of 5 meters per second. After a minute, his position is estimated to be 100 meters north in a straight line from the previous position. This process is called Pedestrian Dead Reckoning (PDR) and is often combined with absolute localization systems (GPS or Wi-Fi based localization). In GAC [54], infrequent GPS samples calibrate a user's relative position calculated by PDR (accelerometer and compass data). An absolute localization system may also be replaced by other mechanisms. Constandache et al.[8] propose an inertial navigation system which does not rely on GPS nor Wi-Fi based localization. It instead leverages electronic maps, which are equipped with additional information on how sensors readings should be changing while a user is taking a specific path. The system uses directional information (compass and gyroscope) to identify a path. For example, a user may be localized to be on a curved path if his compass' readings steadily change in the same direction. Directional information for maps may be automatically generated and pre-fetched to the mobile phone.

Inertial navigation is further leveraged for **indoor localization**. UnLoc [51] identifies certain locations based on their sensors' signatures (also SensLoc [34]). For example, the elevator has a specific accelerometer signature or a computer lab may create a characteristic magnetic field. After those landmarks with their fingerprints are learned by the system in an unsupervised manner, they enable a user to be localised if they return to those places. A similar idea has been previously applied in Wi-Fi based indoor localization systems e.g., Park et al.[43]. The hybrid solution, which combines Wi-Fi fingerprints and other sensor signatures has been also suggested by Radu et al.[48]. Lastly, SurroundSense [4] leverages the ambient sound as a localization fingerprint.

Another idea to improve the capabilities of phone sensing is placing **high resolution sensors** on a device, which may provide more detailed readings. Furthermore, more sensors are put on a mobile phone to infer the context more accurately. Google is currently working on Project Tango [20], which materializes both of those ideas. The prototype of that project can localize a user and build the map of the environment (simultaneous localization and mapping) based on many mobile phones' high resolution sensors and camera.

Although sensors provide much information, it is very difficult to extract meaningful information from them. There are many practical challenges for activity recognition (different algorithms for the same activity, their various parameters and their generality). They also affect other phone sensing applications e.g., inertial navigation. It is believed that those challenges may be addressed by comparing and evaluating algorithms against a big sensor data set of different activities. This idea has been proposed before by Hossmann et al.[27] and successfully executed in other fields e.g. Reality Mining [13]. In phone sensing, Kwapisz et al. [37] is believed to have collected the largest data set (29 different users). However, the all data collection was constrained to one phone position: a front trouser leg pocket.

3.2 Energy measurement

To improve the energy efficiency of phone sensing, an accurate method for measuring it needs to be established. This task is relatively easy for Nokia phones. Nokia Energy Profiler [50], an application delivered by Nokia, accurately measures power consumption and is widely used by research community [36] [39] [38]. However, Google does not provide any equivalent application for Android smartphones. This

makes the task of energy measurement difficult. The first subsection reviews the methods for measuring power consumption available on Android and the second subsection demonstrates energy measurement results of other studies.

3.2.1 Energy measurement methods

The Power Monitor is a hardware power meter delivered by MoonSooon Power-Solutions [31]. It is designed for analyzing power of single lithium (Li) batteries (all Android phones). Although the tool provides accurate measurements, it can only be used in a lab environment i.e., it cannot measure energy consumption of GPS sampling while a user is moving outdoors. Since the device is a costly (771 dollars), it cannot be directly used by most mobile phone users. To alleviate this problem, a mobile phone vendor establishes the power profile of a device as part of their manufacturing process. The power profile is then placed onto a mobile device [3] and may be further utilized by any battery usage application.

Most battery usage applications on Android smartphones leverage a **power model** to determine the device's current energy consumption. Those applications continuously query the state of hardware components such as CPU, wireless card or display to estimate their energy demands. For example, the wireless card may be switched off/on or scanning. Each of those states has an assigned energy consumption e.g. if a device is switched off, its power demand is 2 mA power; if switched on, its power demand is 31 mA and while scanning, its power consumption is 100 mA. All such energy consumption values are available in the power profile provided by a mobile phone vendor (as described in the previous paragraph). The applications aggregate energy consumption values of current states of all components to estimate the total power consumption of a device. Those applications include PowerTutor [55] and Battery Monitor Widget [25]. Although they provide online energy measurements and can by used by users, their measurements are often inaccurate. With time, the power profile provided by a vendor can change, as the battery itself does. To ease this problem, PowerTutor adapts its algorithm based on device model. However, it only supports three mobile phones: HTC G1, HTC G2 and Nexus one. Finally, the battery usage applications based on a power model are not applicable to phone sensing: the power profiles provide no information on power consumption of physical sensors i.e., PowerTutor does not support accelerometer.

Battery life may be also measured to estimate the power consumption of a device. If a mobile device has high energy demands, its battery life should deplete quickly. BetterBatteryStats [26] and Battery Stats Plus [21] use battery life as a metric to monitor mobile applications' energy consumption. This idea has also been utilized by the research community: SenseLess [6] compares the energy efficiency of different sensors based on how long the full battery depletion takes while only one sensor is being sampled. This approach provides an accurate approximation of the power consumption of a device and does requires no additional hardware. Furthermore, the method may be applied by mobile phone users and also works for physical sensors. However, the measurements may be time-consuming. In SenseLess, it took 170 hours to fully deplete the battery life with all sensors switched off. Since 2009 (Senseless), the battery capacity has increased and the Android operating system has become more energy efficient. Full battery depletion is likely to take even longer for modern mobile phones.

TrepnProfiler [32] is a new method for measuring power consumption of a device. The tool, provided by Qualcomm, is a low level software, which runs only on SnapDragon chips, which are also provided by Qualcomm. The tool provides accurate

measurements, including physical sensors, and may be used by mobile phone users. However, the tool is still relatively new and works only on mobile phones with SnapDragon chips.

Method	Accurate	Duration	Physical Sensors	All mobile phones	Outside lab environment
The Power Monitor	✓	Short	✓	✓	-
Power model	-	Short	-	✓	✓
Battery life	**✓**	**Long**	**✓**	**✓**	**✓**
TrepnProfiler	✓	Short	✓	-	✓

Table 1: The comparison of energy measurement methods. Using battery life as metrics is the most suitable method for the project.

In this subsection, four different energy measurement methods were reviewed. The characteristics of them are summarized in the Table 1. The project requires accurate energy measurements outside of the lab environment (i.e. studying power consumption of a device while a user is walking outdoors). The method of **measuring battery life seems to be the most suitable** for that purpose. The method has one disadvantage: it is time-consuming. To alleviate this problem, the 1% percentage battery depletion is proposed instead of the full battery depletion. This approach is further explained in Section 7.1.2. TrepnProfiler would also satisfy the project's requirements, but the tool does not work on all Android devices. However, SnapDragon chips are becoming a standard in the industry and the tool itself may become one as well.

3.2.2 Energy measurement results

The energy efficiency of sensors has been examined before [6] [8] [52] [7]. Those studies provide similar results regardless of the energy measurement methods used; e.g. the Figure 1 shows the energy consumption of different sensors established in SenseLess [6]. All of the studies confirm that the accelerometer is more energy-efficient than the microphone, which consumes less power than GPS and IEEE 802.11 scanning, which are more energy-efficient than the video camera.

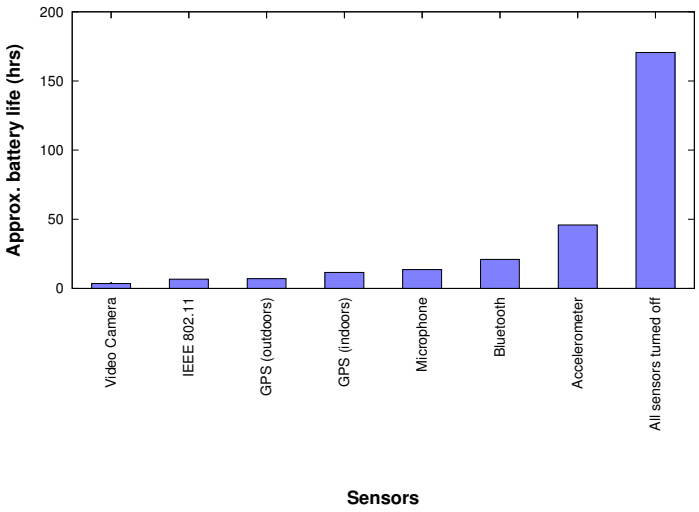


Figure 1: Energy consumption of different sensors established in SenseLess [6]. Other studies provide similar results. Accelerometer is more energy-efficient than GPS or IEEE 802.11 scanning.

Depending on the energy measurement method used in other studies, they provide more details on sensors' energy consumption. For example, GAC [54], which utilizes the Power Monitor device as its measurement method, establishes the GPS power consumption pattern over time (i.e. the peaks of energy consumption at the moments when samples are taking places i.e. every 20 seconds). Also, the energy consumption of sensors may differ depending on their parameters. SmartDC [7] studies how power consumption of GPS and Wi-Fi scanning changes depending on their sampling interval i.e. higher sampling frequency results in higher power consumption. SmartDC also investigates the energy demands of an accelerometer depending on its duty-cycling ratio (sampling time over sleeping interval). A smaller ratio (i.e. small sampling time relative to sleeping interval) results in less power consumption.

All of those studies either only focus on few sensors (e.g. accelerometer, GPS and IEEE 802.11 in SmartDC [7]) or measure sensors' power consumption on only one device [6]. To validate those results, this project aims to provide the complete set of all sensors' energy measurements across three different devices.

3.3 Energy-efficient phone sensing

Relations between sensors may be exploited for energy-efficient phone sensing. For example, if a user is stationary (it can be detected by accelerometer), their geographical position (GPS coordinates) will not change. This relation between accelerometer and GPS, allows the high-power sensor (GPS) to be switched off based on the results of the energy-efficient one (accelerometer). This observation is used by the Fused Location Provider [2], a smart location framework provided by Android. In the research community, SenseLess [6] leverages the same idea of substituting GPS with accelerometer. There are also frameworks that utilize more complex relations between sensors. EEMSS [52] recognizes high-level user state (e.g., working in a office) and optimizes by learning relations between motion information (accelerometer), location (GPS and Wi-Fi based) and background noise (microphone). If it is quiet and

an user is not moving, he is working in a office or resting at home (no need to sample GPS). ACE [42] is a similar system, which is also capable of making guesses about user state. Additionally, it also acts as a middleware, which may save energy among many applications.

Continuous sampling of physical sensors (e.g. accelerometer) is energy inefficient. Since all high-power hardware components need to stay awake (LittleRock [45]), it results in fast battery depletion. To reduce those energy demands, a sensor may be sampled only a fraction of the time e.g. two seconds of sampling and two seconds of not sampling (sleeping interval). Such an approach is called **duty-cycling**. Energy is saved, but interesting events (a user starts walking) may be missed if they happen over the sleeping intervals. There is a specific trade-off in choosing the duration of sleeping intervals. Longer sleeping intervals will save more energy, but increase the probability of missing interesting events. To resolve this, the duration of sleeping intervals may be adjusted depending on whether interesting events are being detected. For example, if a user starts walking, the accelerometer sampling should become more aggressive. This adaptive sampling methodology was proposed by Rachuri et al. [47]. SociableSense [46] further develops this methodology. It controls the duration of sleeping intervals depending on the history of detected events. If someone said one word, the duration of microphone sleeping intervals should be decreased, but not to the same extent as when continuous speech is detected over last half a minute. This idea is formalized as a learning technique based on the theory of learning automata. Similar ideas have also been previously studied in wireless communication [10] [12] [11].

In phone sensing, feature extraction and classification phases are computation-intensive. It has been proposed to **offload the computations to the cloud**. This could result in energy optimizations, but also poses additional challenges: it affects latency (network delay) and a user's data plan (network bandwidth). SociableSense [46] investigates the trade-off between energy consumption, latency and a user's data plan in more etails. Also, internet connectivity may not be always available or there may be poor wireless coverage. Musolesi et al. [41] studies the energy efficiency of different strategies depending on the type of Internet connectivity available.

There are **regular patterns in human behavior and mobility** [16] [5], which could be leveraged for energy optimization in phone sensing. Those patterns may be learned by machine learning algorithms and may be utilized to schedule sensing more efficiently. EnLoc [9] leverages the logical mobility tree to schedule GPS sampling. The tree holds information on when a user moves and where he goes e.g. he leaves his home at 8:00 am to get to the office. The walk takes him 20 minutes. EnLoc may learn this pattern and schedule GPS at 8:00 to confirm that a user left home as usual. Next, it will schedule GPS at 8:20 to verify that a user is in the office. Thanks to the user's regular pattern, GPS sampling conserves energy. SmartDC [7] takes this concept further: it is capable of learning a user's mobility tree in an unsupervised manner.

Other approaches to energy-efficient phone sensing also exist. For example, Srinivasan et al. [49] proposes two-tier activity classification. First, it performs short accelerometer sampling to determine whether a user is moving at all. If so, it samples accelerometer longer to establish exact user's physical activity. If a user is not moving, the second longer sampling is not performed. The total energy consumption of recognizing a user not moving is lower for two-tier activity recognition, therefore making the whole system more energy-efficient. Another interesting approach to energy-efficiency is Hsieh et al. [28]. It analyzes different Android Inter-Process Communication mechanisms for continuous sensing applications. Three mechanisms (Intent, Binder and Context Provider) are studied in terms of their latency, memory footprint and CPU us-

age. Although power consumption is not analyzed, memory footprint and CPU usage are likely to significantly impact energy efficiency. For payload sizes smaller than 4KB (as for all physical sensors), the Binder delivers the best performance with the smallest memory footprint and CPU usage.

Many different approaches to energy-efficient phone sensing have been proposed. As those approaches are focused on different parts of phone sensing, they could be combined to obtain higher energy efficiency. For example, regular patterns in human behavior could be incorporated into adaptive sampling. SociableSense [46] uses a simple learning technique to control the sleeping interval duration. This could be more efficient if the regular patterns of a user were taken into account. For example, take a user who walks for 20 minutes at 8am every Monday. The duration of sleeping interval in accelerometer sampling could be adjusted more significantly on Monday than other days. Another example is leveraging two-tier activity classification for adaptive sampling: If a system is unsure how long sleeping intervals should be, it can sample for shorter period of time to figure it out.

3.4 Conclusions

In this section, the capabilities of phone sensing have been reviewed. It was concluded that phone sensing may be further improved by collecting large data sets of labeled activities. In the subsequent subsection, energy measurements methods were studied and a suitable method for this project was chosen. Measuring time of 1% battery depletion was proposed as an improvement to full battery depletion. Furthermore, the energy efficiency of sensors established by other studies was analyzed. Finally, approaches to energy-efficient sensing were studied. It was suggested that further energy optimizations may be achieved by combining different approaches e.g. leveraging machine learning and two-tier classification for adaptive sensor sampling. The section has addressed two primary objectives 1a and 2a of the project.

4 Requirements specification

This section specifies the properties the software solution must satisfy to fulfill the objectives of the project. Analogically to the objectives, the requirements are characterized by priority: primary (priority 1), secondary (2) and tertiary requirements (3). Furthermore, the requirements are separated into two sections.

4.1 Requirements: mobile phones and energy measurements

1. All software components of the project will work on the Android mobile operating system.	
Type	non-functional
Priority	1
Description	The energy-efficient sensing library, Locy and all Sample Sensor Applications used for energy measurements should work on the Android mobile operating system.

2. Software components should work on a variety of Android devices.	
Type	non-functional
Priority	2
Description	The energy-efficient sensing library, Locy and all Sample Sensor Applications used for energy measurements should work on a variety of Android devices.

3. The project should empirically determine the energy costs of different sensors.	
Type	functional
Priority	1
Description	In the experiments, the order of different sensors' energy efficiency should be established. The Sample Sensor Applications and a measurement method are involved in this requirement.

4. The library will calculate energy costs of different sensors in real time.	
Type	functional
Priority	2
Description	The energy measurements of the sensors could be calculated online, once the library is installed on a user's device.

4.2 Requirements: the energy-efficient sensing library

5. Energy efficiency of the library should be tested against a baseline implementation.	
Type	functional
Priority	1
Description	In another series of experiments, the power consumption of the smart library needs to be compared against a naive implementation. This allows the library's utility to be validated.

6. The library will be integrated with Tristan's experience sampling method (ESM) mobile application.	
Type	functional
Priority	1
Description	The library should be easily plugged into existing mobile applications. Tristan's ESM is used as an example application for that purpose.

7. The energy saving algorithm should be adaptive according to current battery life.	
Type	functional
Priority	2
Description	The algorithm should be more energy-efficient and less accurate while battery level is low. This would allow to utilize better the remains of the battery life.

8. The library intends to manage mobile applications' contention for sensor usage.	
Type	functional
Priority	2
Description	Many application can use a library in energy-efficient manner. That would make a library more practical for industry purposes.

9. The energy saving algorithm should improve its performance by learning from user behaviour.	
Type	functional
Priority	3
Description	A library may improve its energy efficiency while regular patterns in a user behaviour are known. The utility if the library would be higher in users' eyes and it is considered as original idea in research community.

5 Software engineering processes

Besides Dr Tristan Henderson being my supervisor, he also acted as **a fictional customer** of my software product. The continuous client contact allowed us to apply an **Agile** approach. At the beginning of our cooperation, we established what we want to achieve (objectives) and how to measure whether we have achieved them (requirements). During the project, we used those requirements to indicate whether we are on track to finish the product on time.

We applied **Test-Driven Development**: every week, I tested and implemented a next block of the final product. Then, I documented my progress (all progress files are available in *progress/* directory) and met with Tristan. He provided me with continuous feedback on my work and assessed whether I met last week's aims. At the end of each of these meetings, we set goals for the next week. This approach was suitable for **a research project**, as it was often difficult to estimate the time I would need to complete each part of the project. Some of our objectives were new to the research community and so we simply did not even know whether they were possible. Energy-efficient sensing is a very complex subject, so a different software engineering process might have introduced a risk of not delivering the product on time. By making development cycles short and having continuous feedback, I could achieve the objectives set out at the beginning. For example, Locy went through a couple of phases. First, there were two applications: an application continuously sampling the GPS (baseline application) and another continuously sampling the accelerometer. These applications' energy efficiencies were compared. The latter had lower energy demands, and thus was evolved into its new form. The new form of the application switched on GPS when it inferred from motion detection that a user was not walking. After this was implemented, its energy efficiency was evaluated. The result was not so promising, and thus, duty-cycling was introduced. This was followed by another series of energy efficiency experiments. Finally, the application was refactored into a separate library, called Locy. Such development cycle shows how early testing (evaluating energy efficiency) has affected the project's design decisions.

For the development process, we also used a variety of support software. The code was hosted and shared between myself and Tristan by using Git, a popular source code management tool. Since the very beginning, the whole project (except for Tristan's ESM) was open-sourced and publicly available on GitHub: `https://github.com/mk74/locy`. We believed this would ensure higher code quality and attract other researchers who work on the same problem. All applications were created in Android Developer Tools [19], a special plugin for Eclipse. Google's new Android IDE, Android Studio [22] has interesting features specifically for Android development (e.g., code refactoring to run faster), but is still in beta. The testing of our library project involved energy measurement, which we couldn't automate, hence why no testing framework was used. Lastly, we used CiteULike for sharing and managing research papers and references for this document.

6 Ethics

The project did not raise any ethical considerations. Initially, it was believed that a study with real users needs to be conducted to verify the performance of our energy-efficient sensing library, Locy. If the algorithm were dependent on the specific behavior of a user, the performance of the library would need to be validated in the experiments with real users. As the project progressed, it turned out that the library's algorithm is independent of user behavior. Therefore, the experiments involving only myself were sufficient to validate the performance of the library. Regardless, the preliminary ethics considerations were analyzed in **Preliminary Ethics Self-Assessment Form, which is attached in Appendix C**.

7 Design

This section describes the design of the software components and experiments involved in this dissertation. The first subsection outlines how energy demands of different sensors are compared. This includes the design principles behind sample sensor applications and the details on how to measure their energy efficiency. The second subsection lays out the design of the energy-efficient sensing library, Locy. It summarizes its energy-efficient algorithm, how the algorithm can be differentiated depending on battery life and the library's architecture. The design presented in this section will be revisited in the next section.

7.1 Sensor energy measurements

Sensor energy measurements are a the part of the project, where the energy efficiencies of different sensors are determined. To achieve this objective, each sensor is represented in a series of simple Android applications, **Sample Sensor Applications**. Each of these applications continuously samples one sensor and switches off all others. Sample Sensor Applications are designed such that the only one difference between them is which sensor is being sampled. To compare sensors' energy demands, a novel method of measuring energy efficiency of the applications is proposed. The method calculates time each application needs to deplete one percent of battery life. The order of energy efficiency of different sensors may be established by comparing those time measurements: the most energy efficient sensor is the one for which one percentage battery depletion takes longest.

7.1.1 Sample sensor applications

To determine differences in sensors' energy efficiency, each sample applications samples only one sensor while all others sensors are switched off. However, **this triggers other differences between applications**: various sampling frequencies and sizes of sensor data. Since the aim is to compare the energy efficiency of the sampling itself, the the other differences' impact has been minimized where possible.

Sampling frequencies vary among sensors. Inertial sensors e.g., accelerometer or gyroscope could be sampled as often as every 20 milliseconds, whereas the minimum frequency of GPS is 20 seconds. The issue is more complex with wireless communication sensors such as Bluetooth or IEEE 802.11. A full Bluetooth scan takes around 12 secs, but could be stopped earlier and restarted. To account for those differences, **the least energy-efficient strategy of complete, correct sampling** was chosen for comparing sensors' energy efficiency. Sensors are sampled as often as possible, providing the previous sample delivered valid data. For example, the next IEEE 802.11 scan is started once the previous one has finished (IEEE 802.11 scan takes around 2.5 secs depending on the device). This provides complete information on available access points, while minimizing sampling frequency, which increases energy consumption. The strategy will look similar for GPS and Bluetooth. Because other sensors' (inertial sensors, camera and microphone) sampling is immediate, they are sampled as often as possible.

To check whether the Sample Sensor Applications work correctly, each application prints out its raw sensor data. The frequency of this output is 1 per second and is constant across applications. As the previous paragraph describes, sensors' data are delivered at different frequencies. Raw data being printed at different frequencies could add noise to our experiments. By using **the same raw data printing frequency for all**

applications, the differences in sampling frequency are mitigated. The GPS sample application prints out its raw sensor data as often as accelerometer sample application, even though they have different sensor sampling frequencies. It is worth noticing that as a side effect, printed values may be repeated or missing, when no new sensor data was delivered in the print interval.

Another issue concerning applications' output is that **sensors have different raw data sizes**. For example, the result of wireless communication scans (IEEE 802.11 and Bluetooth) consist of a list of records (e.g., 20 access points with their names and other parameters), while a light sensor's result is a single number. Furthermore, more complex data is delivered by the camera and microphone. As the raw data is printed on a regular basis, such differences could lead to significant changes in energy efficiency of the Sample Sensor Applications. To alleviate those differences, **the output is standardized** to the same form for all applications. One line of output, including between 1 and 3 numbers, is printed.

Those numbers may represent different information depending on the sensor:

- For wireless communications sensors, only the number of available access points is printed:

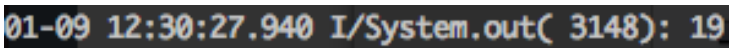


Figure 2: The output for IEEE802.111 sensor application.

- For the microphone, the maximum absolute amplitude for every second is printed:

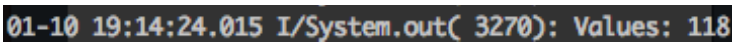


Figure 3: The output for microphone sensor application.

- For the camera, the total size of the recorded video is printed:

Figure 4: The output for camera sensor application.

- The light and proximity sensor provide only one value:

Figure 5: The output for light sensor application.

- Other inertial sensors provide all three coordinates:

Figure 6: The output for accelerometer sensor application.

There are still other issues that could add noise to the experiments e.g., various API among sensors or different capabilities of IEEE 802.11 wireless cards. However, all differences noticeable on the application level are eliminated. Such a design of the Sample Sensor Applications guarantees that the energy efficiencies of different sensors' sampling is compared.

7.1.2 1% battery depletion

Once the Sample Sensor Applications are prepared, they need to be ranked by energy efficiency. To compare their energy efficiencies, time measurements on how long 1% depletion of battery life are made. Those time measurements are then compared. For any two sensors, if a longer time is needed for 1% depletion by one of them, that sensor is more energy-efficient. It is worth mentioning that 1% is used, because it is the precision of the information on battery life that the Android API provides.

The method should be accurate if the comparison is made on the same percentage of battery life e.g., between 99% and 98%. Although the battery life is nonlinear, the battery should behave similarly on the same percentage across many runs. Repeated measurements should be similar. Also, time measurements should be brief, enabling accurate online energy measurement. All of these assumptions will be validated in the next section.

For the purpose of this dissertation, this measurement method was directly incorporated into Sample Sensor Applications. Each sample sensor application continuously queries the battery status and the time of 1% battery depletion is registered. The whole process is exactly the same for all sample sensor applications, hence adding no noise to the experiments.

7.1.3 The list of sample sensor applications

The Table 2 consists of the complete list of Sample Sensor Applications created for this project. Beside the Sample Sensor Applications, a couple of supporting applications were implemented:

- EnergyMeasurementListSensors - listing all physical sensors available in a mobile phone.
- EnergyMeasurementNetwork - determining energy efficiency of Android wireless-based localization.
- EnergyMeasurementGPSSatellites - determining energy efficiency of acquiring GPS Satellites' signal without solving navigation equations on a mobile phone.

All of these applications can be found in the *sensors_energy_measurement/sample_apps* directory.

Sensor	Application name
Camera	EnergyMeasurementCamera
Microphone	EnergyMeasurementMicrophone
IEEE 802.11	EnergyMeasurementWifi80211
GPS	EnergyMeasurementGPS
Bluetooth	EnergyMeasurementBluetooth
Bluetooth LTE	EnergyMeasurementBluetoothLE
Accelerometer	EnergyMeasurementAccelerometer
Gyroscope	EnergyMeasurementGyroscope
Magnetic Field	EnergyMeasurementMagneticField
Ambient Light	EnergyMeasurementLight
Proximity	EnergyMeasurementProximity

Table 2: The complete list of Sample Sensor Applications.

7.2 Locy

Locy is an energy-efficient sensing library for Android phones. It utilizes relations between sensors to provide energy-efficient localization for mobile applications. The relation is between the accelerometer and GPS sensors. The former may be user to check whether a user is moving or not. If a user is not moving, his geographical location (GPS coordinates) do not change. Therefore, GPS sampling may be switched off. This operation brings energy savings, since GPS sampling has higher energy demands than accelerometer sampling.

The library has three important features requiring further explanation. First, a movement detection algorithm needs to be carefully designed to accurately check whether a user is moving. Second, this algorithm needs to be energy-efficient itself. This goal is achieved by duty-cycling sampling with a frequency dependent on the device's battery life (the Requirement 4.2). Lastly, the software infrastructure of the library needs to be further discussed. Since the library is aimed at being plugged into existing applications (the Requirement 4.2), its API needs to satisfy specific needs.

7.2.1 Movement detection algorithm

The movement detection algorithm is based on accelerometer sensor readings. The sensor provides the raw data describing the acceleration of a device in three dimensions (x, y, z relative to the device's position). The data is difficult to analyze: the acceleration of a user may be split between the axes in different ways depending on the initial position of a mobile phone. While a user is walking, accelerometer data will be significantly different depending on where a user keeps their mobile phone (horizontally in a purse or vertically in trousers pocket). To simplify the analysis, **the total magnitude** is applied over accelerometer data (Equation 1). This reduces the dimensionality of the data to one while retaining the information of the user's acceleration.

$$total\ magnitude = x^2 + y^2 + z^2 \tag{1}$$

Once the total magnitude over accelerometer data is applied, the accelerometer readings of a user's walking and their staying in one place may be compared (the Figure 7). Walking may be characterized by **the series of peaks**. A user makes a step by lifting up his foot and putting it back on the ground. The other foot moves in a similar pattern.

A peak on a graph corresponds to a user's step. If a user is stationary, no peaks are observed.

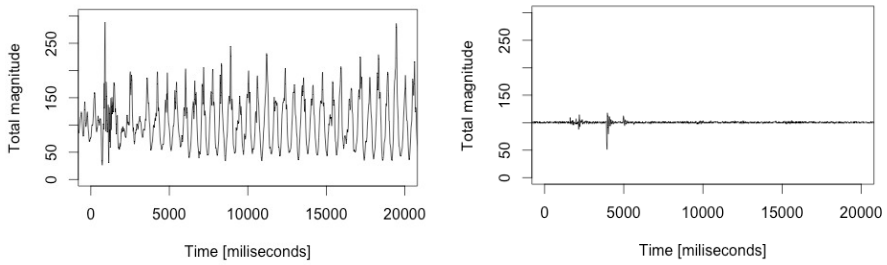


Figure 7: The difference in total magnitude over accelerometer data between walking and not walking. Walking may be characterized by the series of peaks. One peak corresponds to a step made by a user.

For the purpose of Locy, the difference between walking and being stationary needs to be formalized. While a user is walking, the series of peaks means high accelerometer data variation, which can be expressed by **standard deviation**. The Figure 8 contrasts the standard deviation while a user is walking and when they are not (a standard deviation was calculated over 2.5 seconds intervals). The standard deviation is much bigger when the user is walking.

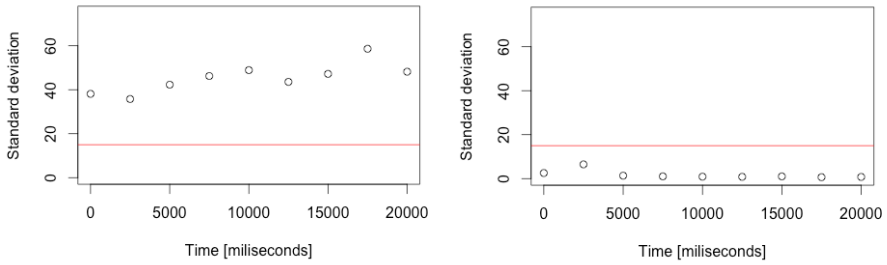


Figure 8: The difference in standard deviation over total magnitude of accelerometer data between walking and not walking. Walking is characterized by higher standard deviation. The threshold of 15 is chosen to classify a user's movement.

There are two challenges with this approach. First, the duration of sampling windows (the interval over which a standard deviation is calculated) needs to be chosen. The Figure 8 uses 2.5 seconds as the duration of sampling windows. Second, the threshold of the standard deviation for the movement classification needs to be determined. The Figure 8 defines the threshold as 15. Both parameters were determined by **trial-and-error**. Different values were chosen and evaluated against the small data set collected for the purpose of this project. The best accuracy was provided by the parameters used in the Figure 8 i.e. 2.5 seconds for sampling windows and 15 as the threshold

for movement classification. Details of this trial-and-error approach are presented in the section 8.2.1.

Locy provides accurate movement detection by undertaking the classification over three consecutive sampling windows. If at least two of them indicates the user is walking (the standard deviation is above the threshold), the user activity is classified as walking. Three consecutive sampling windows last in total 7.5 seconds (the duration of the sampling window is 2.5 seconds). It is assumed that such a movement detection algorithm is accurate and should be universal i.e. it does not dependent on mobile phone type, position or user.

7.2.2 Adaptive duty-cycling sampling

A user is unlikely to change their activity (moving or not) every 8 seconds. This observation could be leveraged for energy optimization. Instead of continuously sampling and classifying accelerometer data, the device could sleep for a certain period of time, conserving energy. After this sleeping interval, the device could start sampling again. This approach is called **duty-cycling** and is incorporated in Locy.

Duty-cycling sampling may be customized by changing the **duty-cycling ratio** between sampling and sleeping window durations. For example, a ratio of 0.5 means that the duration of sampling windows is half the duration of the sleeping intervals. There is an **energy-accuracy trade-off** in choosing the ratio: If the ratio is smaller, the important events are more likely to be missed as sampling windows are relatively shorter. On the other hand, smaller ratio leads to bigger energy optimization as the sleeping intervals are relatively longer. Locy utilizes this trade-off by changing its duty-cycling ratio depending on the device's **remaining battery life**. If the level of battery life is high, the duty-cycling ratio is also high. In that case, Locy does not miss any important events, but is less energy-efficient. If the level of battery level decreases, Locy also decreases its duty-cycling ratio, resulting in additional energy savings. Such an approach satisfies one of the project's secondary requirements (the Requirement 4.2). The details of how the ratio changes are presented in the Table 3.

Battery life	Duty-cycling ratio
80-100%	1
60-80%	0.5
40-60%	0.33
20-40%	0.25
0-20%	0.2

Table 3: The different duty-cycling ratios in Locy depending on the battery life of a device. Locy adopts its algorithm depending on a current battery life.

7.2.3 Software architecture

Locy is an Android library project intended to be plugged into existing applications. Most of those applications use localization services provided by Android's LocationManager. Developers wishing to replace LocalizationManager with Locy should have few difficulties: Locy's API is consistent with the LocationManager API. To use Locy, a developer only needs to add the Locy library to an existing project and change one line in his source code: Instead of getting LocationManager (Listing1), LocyManager needs to be created (Listing 2). All other parts of the code should work perfectly

without additional changes. This approach was tested by plugging Locy into Tristan's Experience Sampling Method (ESM) mobile application. This primary requirement (the Requirement 4.2) was achieved without difficulty. Its results may be viewed in the *ESM/* directory.

Listing 1: Localization services with LocationManager.

```
LocationManager locationManager =  (LocationManager)
   getSystemService(LOCATION_SERVICE);
```

Listing 2: Localization services with LocyManager.

```
LocyManager locationManager = new LocyManager(this);
```

8 Implementation

This section describes the implementation of the experiments and the software components involved in the project. The implementation raised practical challenges, resulting in a revisit of the design presented in the previous section. The first part of this section summarizes the challenges of establishing sensors' energy efficiency. After a description of software and hardware problems, the characteristics of experiments' results are investigated and new terms "wrong samples" and "energy efficiency levels" are defined. The second subsection presents the implementation details of the energy efficient sensing library, Locy. The derivation of the parameters for the movement detection algorithm are described and two extensions of Locy are summarized. Lastly, the work presented in this section reveals important knowledge for the project. This is concluded in the last section.

8.1 Sensor energy measurements

Implementing the experimental designs presented two groups of challenges. First, the designs were affected by the Android operating system and its API. A few of those issues (IEEE802.11 scanning, Bluetooth Low Energy API and privacy policy for video capturing) are described later. Second, the experiments involved many devices, with hardware differences that posed some challenges. The specifications of the involved mobile phones are presented in Table 4.

	Google Nexus 7	**HTC Flyer**	**HTC desire**
Android	4.3	3.3	2.3.3
CPU	Quad-core 1.2 GHz Cortex-A9	1.5 GHz Scorpion	1 GHz Scorpion
Memory	1 GB RAM	1 GB RAM	576 MB RAM
Battery	Li-Ion 4325 mAh	Li-Po 4000 mAh	Li-Ion 1400 mAh
Camera	✓	✓	✓
Microphone	✓	✓	✓
IEEE 802.11	✓	✓	✓
GPS	✓	✓	✓
Bluetooth	✓	✓	✓
Bluetooth LTE	✓	-	-
Accelerometer	✓	✓	✓
Gyroscope	✓	-	-
Magnetic Field	✓	✓	✓
Ambient Light	✓	✓	✓
Proximity	-	-	✓

Table 4: The differences between mobile phones involved in the experiments.

The experimental results raised two additional challenges: Some results were perceived as invalid (referred to here as "wrong samples") and the valid results tended to have specific properties, referred to as "energy efficiency levels".

8.1.1 Software problems

Preliminary experiments showed that **Wi-Fi scanning** was more energy-efficient than the accelerometer. This was unexpected, so the details of IEEE 802.11 scanning were further investigated. Unlike other sensors, IEEE 802.11 has many parameters: an active and passive mode, power saving mode, scan result caching, complete scanning etc. The parameters were not recorded in the Android documentation, but since they could affect the energy efficiency of the IEEE 802.11 observed in the preliminary experiments, their values had to be determined. The results of IEEE 802.11 scanning on a mobile phone were compared with the ones obtained on a laptop. Kismac [35] was used on the laptop for a detailed list of access points, with their signal strengths. Both scanning results were compared both with a mobile and a stationary user. The parameters of IEEE 802.11 scanning on a mobile phone turned out to be identical to ones on the laptop. No caching scheme, no other energy optimization was involved (table 5). It turned out that preliminary experiments of Wi-Fi scanning were correct: A fault with the accelerometer had caused the Wi-Fi scanning to seem more energy-efficient. After fixing the problem, Wi-Fi scanning was found to be less energy-efficient than accelerometer, as expected. It is worth noting that active IEEE 802.11 scanning is deprecated in the Android API. Passive scanning is more energy efficient than active scanning, since radio reception (passive scanning) usually requires 10 times less power than radio transmission (active scanning).

Characteristic	Value
Scanning mode	Passive
Power save mode	No
Caching	No
Complete scanning	Yes

Table 5: The parameters of IEEE 802.11 scanning. These parameters are defaults: no specific energy optimization is involved.

Energy efficiency may not be the highest priority in Android's design. To make accurate energy efficiency comparisons, Wi-Fi connectivity needed to be completely disabled (except for Wi-Fi sensor application). Disabling wireless connectivity in the main settings is not sufficient, as the Android system may switch Wi-Fi connectivity on at any point if prompted by the **Google Location Service**. To undertake valid experiments, Google Location Service had to be also disabled (Figure 9).

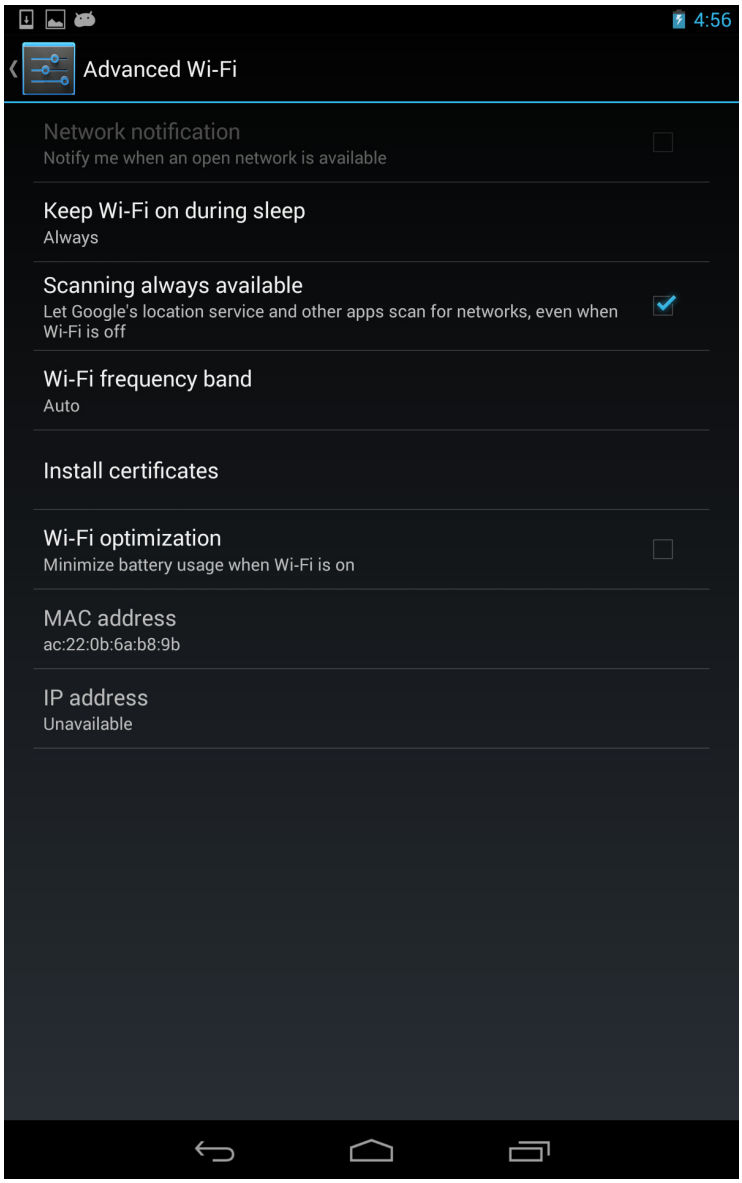


Figure 9: Android Advanced Wi-Fi Settings. The permission of Google Location Serive to perform scanning at any point needs to be disabled.

Another software issue was the **Bluetooth Low Energy (LE) API**. Bluetooth LE was designed to reduce power demands while providing the same functionality as Classic Bluetooth. Classic Bluetooth scanning on Android was therefore expected to be less energy-efficient than the Bluetooth LE. However, the Bluetooth LE API is new (added in Android 18) and still under development. Its current version (Android 18) prints out lots of information to a log file if any device is found, leading to faster battery depletion than Classic Bluetooth. However, Bluetooth LE is more energy efficient if no device is found. This observation is further analyzed in the evaluation section Figure 15.

Lastly, Android imposes its **privacy policy on video capturing** applications. Video

applications are required to provide a camera preview to make customers aware of the recording. This privacy policy results in higher energy demands, as the rendering a camera preview requires additional computation. As mandated by the Android API, a camera preview was added to the Camera Sensor Application.

8.1.2 Hardware problems

Since modern mobile phones (e.g., Google Nexus 7) have energy-efficient hardware, **the time of 1% battery depletion may be significant**. A test application that only continuously queried the accelerometer without performing any other operations (screen was switched off, Wi-Fi connectivity was off) took around half an hour to deplete 1% of battery on a Google Nexus 7. Since online energy measurements 2b are one of the project's objectives, the measurement method should be faster. This could be achieved by adding additional energy demands to all Sample Sensor Applications, such as by running complex computations (e.g. dividing big numbers) and discarding their results. However, such an approach could invalidate out experiments, since different Sample Sensor Applications could make use of CPU power to a different extent. Sample Sensor Applications requiring more CPU (e.g., triangulation for GPS) may be more affected by those computation demands due to e.g. CPU scheduling. The simple solution to this problem was to **always keep the screen on** for the applications. This not only significantly increases all Sample Sensor Applications' energy demands, but can also be added uniformly to each of them without invalidating the experiments. On Google Nexus 7, this operation reduced the time of 1% battery depletion with continuous accelerometer sensing to around 13 minutes (56% shorter).

Another hardware issue was **choosing an initial battery percentage** for the experiments (Table 6). In the case of the Google Nexus 7, battery behavior was stable and repetitive for any chosen percentage. To reduce the time of a full experimental cycle sampling followed by charging the battery, a percentage between 98% and 99% was chosen. The same percentage could not be chosen for other devices, since their batteries were old and less-stable. For example, the 98% -99% percentage was tested for the HTC Flyer, but the results were not repetitive. The percentages for other devices were found by a **trial-and-error approach**. If the results of the experiments conducted on a given percentage were repetitive in a couple of trials, the chosen percentage was considered suitable. Also, some specific characteristics of batteries charging cycles were discovered: For the HTC Flyer, charging was much slower for the last battery percentage (99%-100%) than any other percentage. After being fully charged, the battery depleted much slower for the last percentage (99%-100%). Then, the depletion of the battery between 98% and 99% was immediate. This phenomenon was observed a few times. To avoid it, the percentage between 97% and 98% was tried and turned out more suitable for the experiments on the HTC Flyer. In the case of the HTC Desire, the charging was often not triggered when the battery level was more than 91%. To mitigate this issue, the range from 89% to 88% was used for the HTC Desire. The experiments' results were repetitive and the percentages suitable for the experiments.

Device	The percentage
Google Nexus 7	98%-99%
HTC Flyer	97%-98%
HTC Desire	88%-89%

Table 6: The percentages used for energy measurement method. These percentages are different for the devices and found by trial-and-error.

HTC Desire has less dynamic memory (576 MB RAM) than other devices (1 GB RAM) involved in the experiments. This affects the Camera Sensor Application. Initially, the experiment with video capturing could not be completed for HTC Desire, as there was no enough dynamic memory to encode and save the full video capture. To alleviate this problem, the codecs were changed to H.263 (which uses less dynamic memory when encoding video) and the video dimensions were reduced to reduce the amount of data needing to be encoded. Those parameters were standardized in the Camera Sensor Application across all devices. For consistency, the audio parameters were also standardized in the Microphone Sensor Applications for all mobile phones. Full details on the video and audio capture parameters are available in Appendices B.

8.1.3 Wrong samples

Despite mitigating many problems with the experimental execution, some wrong samples were still produced. Statistically, the wrong samples are defined as samples whose values (time measurements) are **outliers**. The proportion of the wrong samples varies among devices (Figure 10). The Google Nexus 7 had 100% successful samples, so the experiments were conducted up to **3 successful samples per sensor**. Other devices had a higher proportion of wrong samples: between 20 and 30 percent for the HTC Flyer and HTC Desire. Because of that, **5 successful samples per sensor** had to be collected to complete the experiments for those devices. The wrong samples also had different characteristics for the HTC Flyer and HTC Desire.

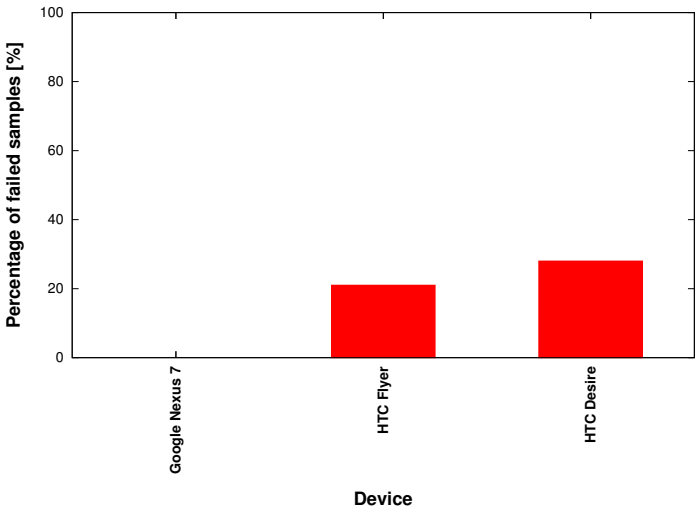


Figure 10: The percentage of the invalid samples among different devices. The method always delivers valid samples for Google Nexus 7.

The wrong samples from the HTC Flyer may be characterized as **bursty errors**. Figure 11 shows that most of the wrong samples (9 out of 11) are grouped in two periods: 9-13 samples and 24-33. The first period includes continuous series of 4 wrong samples, whereas the second period has 6 wrong samples within 11 samples made. Each group has different characteristics, and thus, the bursty errors are difficult to detect without complete knowledge of the experiments' results, as in online energy measurements.

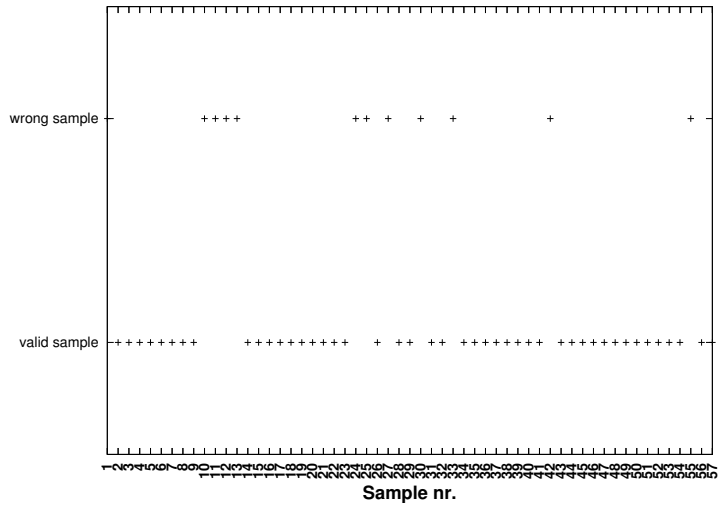


Figure 11: HTC Flyer: the wrong samples over all experiment runs. Those samples may be characterized as bursty errors.

For the HTC Desire, the proportion of wrong samples varies by sensor (Figure 12). Some sensors such as "plain run", Bluetooth, GPS or Camera always provided successful samples. On the other hand, the light and proximity sensors had high proportions of wrong samples (each around 60%).

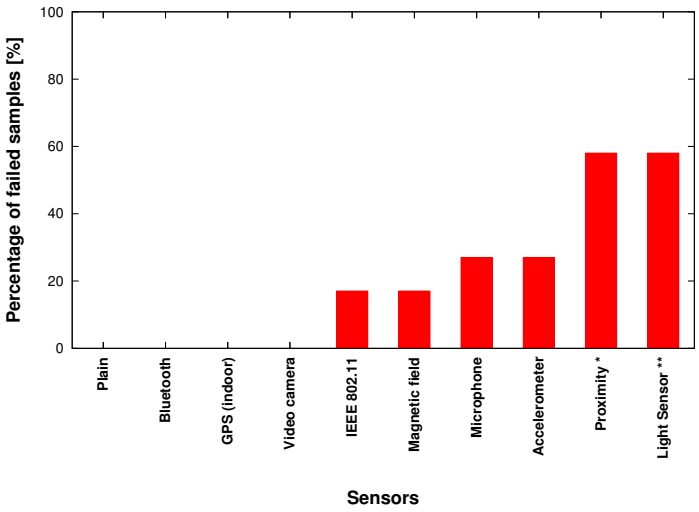


Figure 12: HTC Desire: the percentage of the invalid samples per sensor. Some sensors always provide successful samples, whereas light and proximity sensors deliver high proportion of the invalid samples.

8.1.4 Energy efficiency levels

After rounding valid samples' time measurements to 10 seconds, it was clear all measurements belonged to specific sets of discrete values. For the HTC Flyer, there were only three such values: 210, 240 and 270 seconds (Table 7). Those values are every 30 seconds. To simplify the valid samples' time measurements, they can be divided by this interval (30 seconds). The results of this operation were termed **energy efficiency levels**. For the HTC Flyer, there were three energy efficiency levels: 7, 8 and 9. A result of any experimental sample belongs to one of those levels.

Value	Samples
210	4
240	22
270	19
Others	0

Table 7: Energy measurement results for HTC Flyer. There are only 3 discrete values among all samples' results.

Similarly with the HTC Desire, all valid samples' results belong to a set of three numbers: 150, 200 and 250 seconds. As the interval between them equals to 50, the energy efficiency levels are 3, 4 and 5. Lastly, for the Google Nexus 7, there were 9 discrete values among the all experiments results. Each value was a multiple of 60 seconds. Each initial discrete values was divided by 60 to establish its energy efficiency level. The full list of all known energy efficiency levels among devices is shown in Table 8.

Energy efficiency level	Google Nexus 7	HTC Flyer	HTC desire
Level 1	-	-	-
Level 2	-	-	-
Level 3	-	-	150 secs
Level 4	240 secs	-	200 secs
Level 5	300 secs	-	250 secs
Level 6	-	-	-
Level 7	-	210 secs	-
Level 8	480 secs	240 secs	-
Level 9	540 secs	270 secs	-
Level 10	-	-	-
Level 11	660 secs	-	-
Level 12	720 secs	-	-
Level 13	780 secs	-	-
Level 14	840 secs	-	-
Level 15	900 secs	-	-

Table 8: The complete list of known energy efficiency levels across devices.

To compare the performance of two sensors for a single device, the energy efficiency levels may be used. The sensor with a higher energy efficiency level has lower energy consumption. The energy efficiency levels may also provide the information about **the energy distance** between two sensors. If there is a known energy efficiency level between two sensors' levels, it may be reasoned that the difference in energy consumption is "significant". For example, if a sensor belongs to level 6 and another to level 8, the existence of a level 7 (there is a sample with its corresponding value) may imply the sensors are not "closed enough". Finally, the energy efficiency levels are used as a normalization tool. The experiments' results, when represented as a level, could be shown on the same diagram (though they should not be compared).

8.1.5 Conclusions

This implementation of Sensor energy measurements shows how important a role the mobile operating system plays in energy efficiency. There is evidence that Android's design decisions may significantly influence the battery depletion rate. For example, the Android operating system prioritises privacy over energy efficiency by requiring a camera capture preview. It was also shown how a badly designed API may reduce energy efficiency: Bluetooth Low Energy is an energy-efficient sensor, but its API drains battery by extraneous logging. This also raises the question of how energy-aware mobile operating systems' APIs should be.

This energy measurement method was successfully applied for the Google Nexus 7. It delivered 100% valid samples, and may therefore be classified as a stable method. The method performed worse for the other two devices. The ideal initial battery percentage for the experiments needed to be manually customized and the method still delivered as many as 25% of wrong samples. Those two facts eliminate the possibility of online energy measurements. Furthermore, the wrong samples were determined based on the whole set of conducted samples. This set is not available in online energy measurement. Even if possible to mitigate for online measurement, the method would be impractical as it requires collecting many samples to get accurate results.

All samples delivered by the energy measurement method belonged to a small set of discrete values. Those values were further characterized by being multiples of some time interval. This observation was encapsulated by the idea of an energy efficiency level, which will be leveraged in further analysis of the results.

8.2 Locy

No significant challenges occurred when implementing the primary requirements of the energy-efficient library, Locy. The parameters of the movement detection algorithm were established using a trial-and-error approach. Two sample applications were used to achieve it. Furthermore, inLocy was proposed as the extension of Locy. It also leverages movement detection for Wi-Fi based localization. Lastly, a prototype was implemented as a separate service. This approach may be leveraged to handle mobile applications contention. However, it also increases energy demands, and thus was not incorporated in Locy. The final version of the library is available in the *locy/* directory.

8.2.1 Movement detection algorithm

The Section 7.2.1 described the movement detection algorithm. The parameters were chosen by trial-and-error. An R script and the data set used for the trial-and-error approach can be found in the *machine_learning/movement_detection* directory. The data set was collected over two activities: standing in one activity for 150 seconds (*inPlace.csv* file) and walking also for 150 seconds (*moving.csv* file). For most of the former activity, a mobile phone was either put on a desk in front of the user or kept in his front trousers pocket. Although the user occasionally moved his mobile phone or manually unlocked it, he did not move during the whole period (his geographical coordinates did not change). During the walking activity, the user kept his mobile phone in a trouser pocket or held it in front of him while walking constantly. The accelerometer data was collected by a sample application which can be found in the *helpers/SensorsDataProducer* directory.

Once the parameters for the movement detection algorithm were established (the duration of the sampling window was 2.5 seconds and the threshold of movement classification 15), they were also tested on other mobile phones. The parameters were tested online by using the SensorsGraphPlotter application, which can be found in the *helpers/SensorsGraphPlotter* directory. The application draws the graph of total magnitude of accelerometer data over time (Figure 13). It updates its graph live over time.

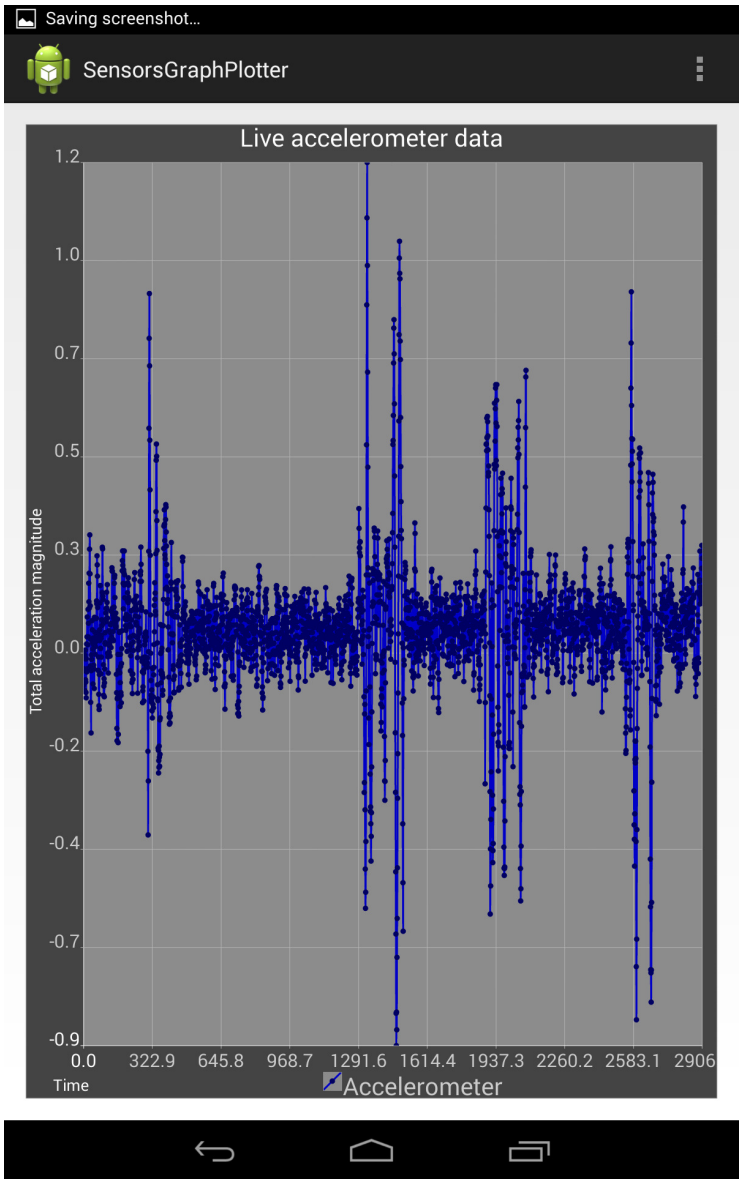


Figure 13: SensorGraphPlotter draws the graph of the total magnitude of acclereometer data over time. It updates the graph live. The application was used to evaluate the parameters of the moving detection algorithm.

8.2.2 inLocy

inLocy is a variation of Locy servicing Wi-Fi based localization. Unlike Locy, there was no existing localization system available for testing. For inLocy, a simple Wi-Fi based localization system was mocked. The mock system receives the results of IEEE 802.11 scanning made by a device and checks whether the results contain a specific BSSID (*9C:4E:20:C8:9E:71* - one of the access points available in the John Honey Building). If so, the user is localized as being in the John Honey building. The

mock system is available to inLocy under http://mk74.host.cs.st-andrews.ac.uk/locy/location.php. The system's source code is available in the *inLocyBackend/* directory. inLocy itself uses the same movement detection algorithm as Locy and its source code is available in the *inLocy/* directory. However, inLocy is only a prototype, and thus serves as a single application and does not use duty-cycling.

8.2.3 Library as a service

One of the project's primary requirements (the Requirement 4.2) was to manage mobile applications' contention for sensor usage (i.e. many applications concurrently using Locy in an energy-efficient manner). To fulfill this requirement, the state of the library must be shared across applications. The state is not shared if Locy is added to the project as a library (as in the current version of Locy). The state can be shared by defining Locy as a separate application, running as a background service. The application should be set to automatically start with the Android operating system (i.e. *BOOT_COMPLETED* intent action) and use Inter Process Communication (IPC) mechanisms to provide localization services to mobile applications. Although this should result in energy savings, the preliminary evaluation showed unpromising results. Implementation of IPC mechanisms in Locy would likely not be energy-efficient, but more work needs to be conducted to verify the result. Because of its high energy consumption, the feature is not in the final version of the library. The feature is nevertheless available in the *locy_as_a_service* branch of the repository.

8.3 Conclusions

The implementation of sensor energy measurements illustrates the crucial importance of the software layer in the energy efficiency of mobile phones. It was proven that some of Android design decisions result in rapid battery depletion. Additionally, the energy measurement method indicates specific characteristics of a device's battery life. Some samples were wrong due to bursty errors and error-prone sensors. Similarly, the valid samples' values seemed to cluster around a small set of discrete values (energy efficiency levels). Those levels are utilized for further analysis in the next section. Although the method is stable, it cannot be applied as an online energy measurement tool (the Requirement 4.1). For older batteries, the method requires many samples, making the method impractical.

Since most of the challenges were discovered during the implementation of the Sample Sensor Applications, there were no significant difficulties with the implementation of Locy. The details of how the parameters of the movement detection algorithm were derived were explained. inLocy, a variation of the library, was also proposed. It showed that it is possible to use Wi-Fi based localization instead of GPS. Lastly, Locy was also prototyped to work as a separate background service. Although it deals with application's contention for sensor usage the prototype had a higher energy consumption.

9 Evaluation and critical appraisal

This section mainly focuses on the analysis of energy efficiency of the components described in this dissertation. In the first subsection, the sensor energy measurements' results are evaluated. Basing on those results, I reach general conclusion and also inspect the sensors used for localization in more details. In the second subsection, the energy efficiency of Locy is investigated. The energy-efficient sensing library is compared against the baseline GPS localization service in two real-life scenarios. At the end, the conclusions are presented.

9.1 Sensor energy measurements

The Figure 14 shows the complete set of the experiments results across three different devices. For all of those phones, the time of 1% battery life depletion is the biggest when all sensors are switched off i.e. "plain" run. Next, all of the physical sensors belongs to the group of less energy-efficient sensors. On the boundary of this group, microphone may be noticed. Finally, IEEE 802.11, GPS and Video Camera have relatively high energy demands on all of the three devices.

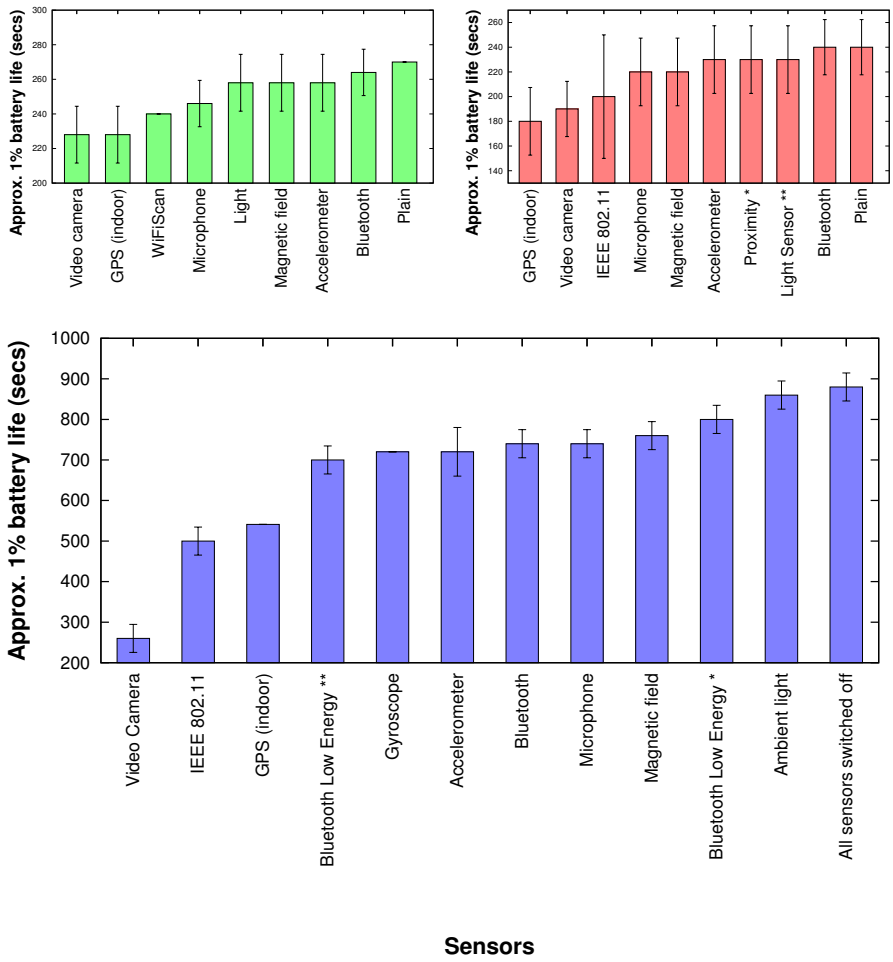


Figure 14: The complete results of energy measurements for all devices. The energy efficiency of different sensors is as expected and overlaps with the results of other research.

Bluetooth Low Energy (LE) may have higher energy demands than Classic Bluetooth. The Figure 15 compares their energy efficiency depending on the situation. In there is no device found, it takes longer for Bluetooth LE scanning to deplete the battery life by 1% than for Classic Bluetooth scanning i.e. Bluetooth LE is more energy-efficient. However, if there is any device on the network, Bluetooth LE scanning depletes battery faster than Classic Bluetooth. The reason behind this phenomenon was analyzed in the previous section (the high output of Bluetooth LE API when any device is found) 14.

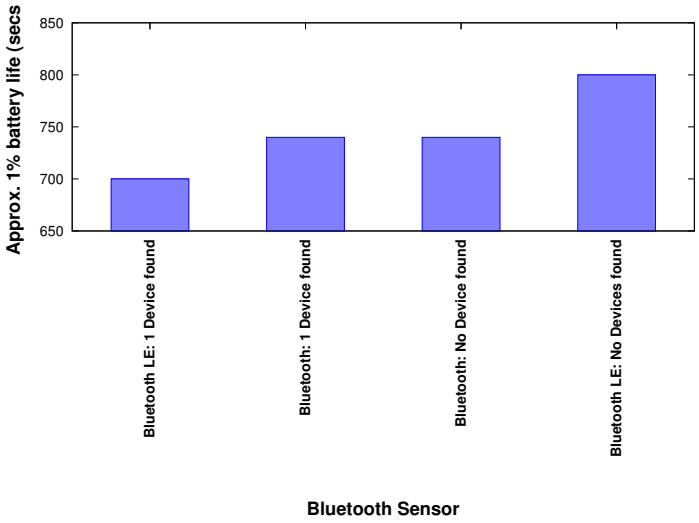


Figure 15: Energy efficiency of different Bluetooth sensors. Bluetooth Low Energy has higher energy consumption than Classic Bluetooth when any device is found.

The energy efficiency order of the sensors differs among the devices. The Figure 16 contrasts the energy efficiency results for the shared sensors between the devices.For each of its diagrams, the order of the sensors on x axis is the same. The diagram on the left (Google Nexus 7) is characterized by monotonically increasing energy efficiency of the sensors, whereas the other two diagrams are not. For example, Bluetooth is the most energy efficient sensor HTC Desire and HTC Flyer, but is average for Google Nexus 7.

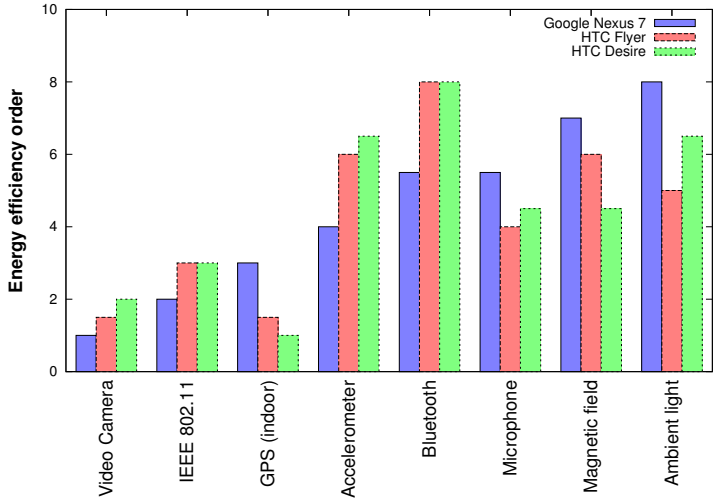


Figure 16: Energy efficiency of shared sensors across different devices. The order of energy efficiency differs depending on a device e.g. Bluetooth is the most energy efficient sensor for HTC Desire and HTC Flyer, but not for Google Nexus 7.

9.1.1 Localization

Different sensors may be used for localization purposes. The Figure 17 studies the energy efficiency of those sensors. For the comparison, the energy efficiency levels described in the previous section were used. For each of the devices, Bluetooth has the highest energy efficiency level i.e. it is the most energy-efficient. Also, microphone is always more energy efficient than traditional localization sensors (GPS and IEEE 802.11). On the other hand, the camera has worse energy performance than GPS and IEEE 802.11 on the two devices. Finally, the energy efficiency of GPS and IEEE 802.11 are similar, though the latter is more energy-efficient for two out of three devices.

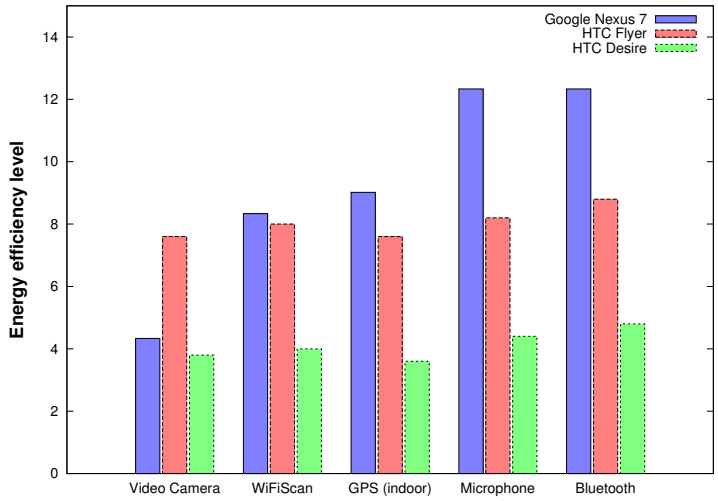


Figure 17: Energy efficiency levels of the sensors used for localization (Camera, IEEE802.11, GPS, Microphone, Bluetooth). The diagram explains how localization technologies evolve e.g. iBeacon, which is based on Bluetooth is widely used by industry because of Bluetooth's low energy demands.

Physical sensors are more energy efficient than traditional localization sensors. The energy efficiency of accelerometer is compared with GPS and IEEE 802.11 scanning on the Figure 18. The accelerometer is more energy efficient, but the difference is not always substantial. For HTC Flyer and HTC Desire, the difference between the energy efficiency of accelerometer and traditional localization sensors is less than one energy efficiency level. It is worth reminding here that Accelerometer Sensor Application applies continuous sampling (i.e. sample as often as possible without sleeping intervals).

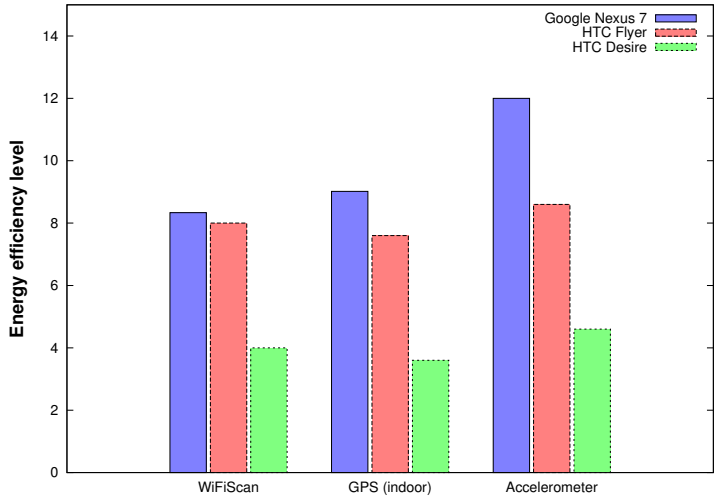


Figure 18: Energy efficiency levels of IEEE 802.11, GPS and accelerometer sensors across different devices. Acceleromter is more energy-efficient, but the difference is not substantial, and thus, efficient accelerometer sampling strategies needs to be introduced.

9.1.2 Conclusions

The complete set of energy measurements' results are intuitive and overlap with other research, which uses different measurement methods [8] [52] [7]. This gives validation to the energy measurement method used in this dissertation. For Google Nexus 7, the results confirm the energy-efficiency problems with Bluetooth Low Energy API, which were noticed during the implementation phase. Finally, the energy measurement results vary among different devices. This justifies why the universal power model cannot be used. It also validates the necessity of online energy measurements.

The analysis of the localization sensors' efficiency explains how localization technologies evolve. Bluetooth, the most energy efficient sensor, is already widely adopted in the industry. iBeacon [30], provided by Apple Inc., leverages Bluetooth LE for the indoor positioning system. Also, promising energy-efficiency results of microphone attracted research community. SurroundSenses [4] utilizes microphone for localization through ambient sound fingerprinting. Lastly, camera is the most expensive sensor, and thus, computer vision is not yet widely used for localization.

Accelerometer is more energy efficient than traditional localization sensors. Opposite to what previous research[6] stated, the difference in energy consumption is not substantial. The nature of accelerometer usage has been changed. Currently, the values of physical sensors may be pulled more often e.g., three accelerometer values are delivered per second on Google Nexus 7. Higher frequency of raw data may have better accuracy, but also results in faster battery depletion. Because of that, the smarter strategies for sampling needs to be designed. For example, accelerometer may sample for short period of time and if unsure what happens, it could sample longer. As accelerometer has lower energy demands than traditional localization sensors, it could be leveraged for "cheaper" replacement for those "expensive" sensors. However, its sampling needs to be conducted in energy-efficient manner.

The energy measurement method does not provide the details on the nature of en-

ergy consumption. It is unknown which exact operations consume the most energy during sensing. For example, significant part of energy may be consumed by CPU which needs to be active while sensing accelerometer [45]. As those characteristics are not recognized, it cannot be reason how energy efficient the applications with combined sensors (IEEE 802.11 scanning and accelerometer) will be. Also, it is unfamiliar how energy efficiency of physical sensors will change depending on their parameters (size of sleeping and sampling windows).

9.2 Locy

To evaluate the energy efficiency of Locy against the baseline implementation (the Requirement 4.2), two example applications were created. First of the applications uses naive Android's LocalizationManager to obtain GPS position, while another uses smart Locy's LocyManager. Beside that, there are no other differences between the applications. The former application may be found in the *LocyGPSLocalization/* directory, whereas the latter is available in the *NaiveGPSLocalization/* directory. The energy efficiency of the applications were compared in two real-life scenarios across three different devices. For each pair of a scenario and a device, three samples of energy performance were collected (all traces of the sample are available in the *user_experiments/inLocy* directory). The energy efficiency levels for each scenario were established by applying the process described earlier (the Section 8.1.4).

In the **first scenario**, a user is staying in one place for the whole period of sample collection (i.e. time of 1% battery depletion). A user was staying in front of the John Honey building and did not move. The comparison of Locy and the naive GPS implementation are presented in the Figure 19. Locy has lower energy consumption than the naive GPS implementation for all three devices. Locy detects a user not moving and switches off GPS. Locy's Accelerometer sampling is more energy efficient than GPS, and thus, its energy consumption is lower.

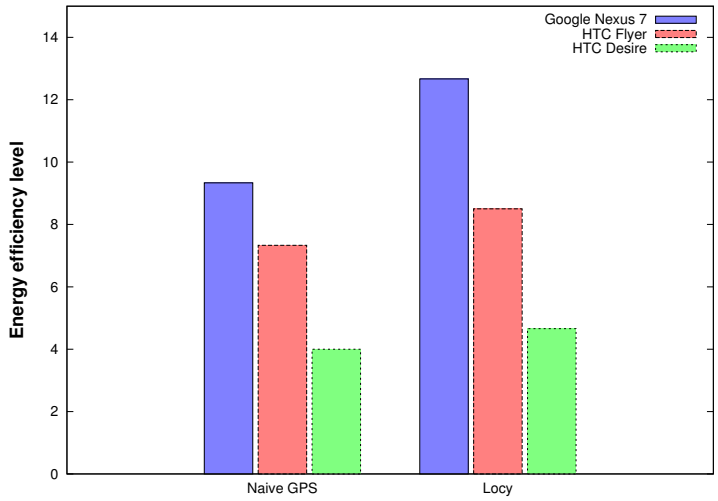


Figure 19: Energy efficiency levels of Locy library and Naive GPS Localization while a user is in place. Locy is more energy-efficient than baseline implementation for every phone.

The **second scenario** checked how Locy performs in more complex situation. A user was walking for 110 seconds and then he stayed in one place for the rest of the time (till the 1% of battery is depleted). Walking was undertaken along the Jack Cole building with stopping and standing in one place in front of the John Honey building later. The Figure 20 compares the energy efficiency levels of Locy and the naive GPS implementation. Locy is more energy-efficient for all three phones.

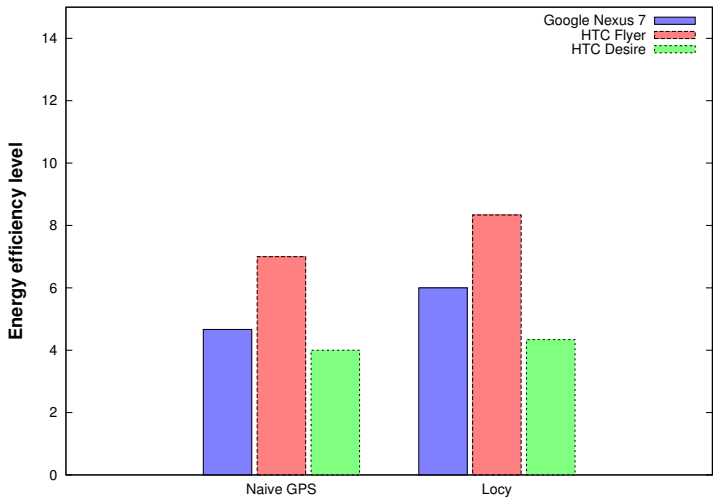


Figure 20: Energy efficiency levels of Locy library and Naive GPS Localization while a user is half of the time moving and the rest he is staying in one place. Locy is more energy-efficient than baseline implementation for every phone.

9.3 Conclusions

The energy measurement method delivered the complete set of accurate energy efficiency results, and thus, the method was validated. The results were gathered across three different mobile phones and include all sensors. It is believed that it is the first study of this type.

Physical sensors could be leveraged for localization purposes. The energy efficiency results gave a basis for how an energy-efficient library could be designed. The energy measurement method provided useful information and identified the challenges for Locy. The energy efficiency of combined sensors and different energy efficiency of physical sensors depending on their parameters needed to be checked.

Those challenges were met by Locy. The energy-efficient sensing library was evaluated against naive GPS implementation in two real life scenarios. While a user is staying in one place, Locy is more energy-efficient on three different devices. While a user is walking for half of the time and staying in one place for the rest, Locy outperforms naive GPS implementation. However, the difference is not as significant as in the first scenario.

10 Conclusions

Three areas relevant to this project have been reviewed (phone sensing, energy measurements methods and energy-efficient sensing). A possible enhancement has been identified in each area. It was concluded that further progress in phone sensing could be made by building a large sensor data set of different user activities. Furthermore, measuring 1% of battery life was proposed as a suitable method of energy measurements for the project. Lastly, hybrid approaches of different energy-efficient techniques were suggested to achieve additional energy savings. Measuring 1% of battery life is a novel way of investigating the energy efficiency of a mobile phone. The method was tested in a series of the experiments and found to be valid. It also revealed specific characteristics of battery life (wrong samples and energy efficiency levels). However, the method is unstable for older batteries, and thus unsuitable for online energy measurements.

The series of experiments yielded the energy efficiency of all sensors available across three different devices. It is believed to be the first study of this type. Different sensors had different energy demands depending on device, pointing out the importance of online energy measurement for energy-efficient sensing. The energy efficiency of the sensors utilized for localization were analyzed in more detail. The results explain how localization technologies evolve. Finally, physical sensors turned out be more energy-efficient than standard localization sensors (GPS and IEEE 802.11 scanning). This observation was the basis for the design of the energy-efficient sensing library, Locy.

Locy provides a GPS localization service in an energy-efficient manner, by leverating the energy-efficient accelerometer to infer whether a user is moving and hence whether it's necessary to invoke the high-power GPS at all. Further energy optimization is achieved by utilizing duty-cycling when samplingthe accelerometer. The duty-cycling ratio (sampling time over sleeping interval) is changed depending on the level of battery life, using less energy if the battery is low. Locy can easily be plugged into existing applications since it exposes the same API as the LocalizationManager provided by Android. Plugging Locy into Tristan's ESM was a one-liner.

Locy's energy efficiency was evaluated in a separate series of experiments. Its performance was compared against the naive GPS implementation. The experiments involved two real-life scenarios conducted on three different devices. Locy turned out to have lower energy demands than the naive GPS implementation.

Two extensions for Locy were proposed: inLocy (Wi-Fi based localization) and Locy working as a service. In future work, these extensions could be further developed and evaluated in real-life scenarios. Locy could also leverage other ideas for energy-efficient sensing: The regular patterns in human behaviour may help optimize duty-cycling when sampling (by changing duty-cycling ratio accordingly). A next research project is planned to further investigate that idea.

This project extended knowledge in energy-efficient phone sensing. It revealed novel characteristics of battery life and provided the complete set of sensor energy efficiencies across different phones. An energy-efficient sensing library was created, outperforming the standard Android implementation.

References

[1] Android. ActivityRecognitionClient — Android Developers, 24 Feb. 2014. Online at http://developer.android.com/training/location/activity-recognition.html.

[2] Android. Location APIs — Android Developers, 24 Feb. 2014. Online at http://developer.android.com/google/play-services/location.html.

[3] Android. *Power Profiles for Android*, 22 Feb. 2014. Online at http://source.android.com/devices/tech/power.html.

[4] M. Azizyan, I. Constandache, and R. R. Choudhury. SurroundSense: mobile phone localization via ambience fingerprinting. In *MobiCom '09: Proceedings of the 15th annual international conference on Mobile computing and networking*, pages 261–272, Beijing, China, 2009. doi:10.1145/1614320.1614350.

[5] N. Banerjee, A. Rahmati, M. D. Corner, S. Rollins, and L. Zhong. Users and Batteries: Interactions and Adaptive Energy Management in Mobile Systems. In *Proceedings of the 9th International Conference on Ubiquitous Computing*, UbiComp '07, pages 217–234, Innsbruck, Austria, 2007. Online at http://portal.acm.org/citation.cfm?id=1771592.1771605.

[6] F. Ben Abdesslem, A. Phillips, and T. Henderson. Less is more: energy-efficient mobile sensing with SenseLess. In *MobiHeld '09: Proceedings of the 1st ACM workshop on Networking, systems, and applications for mobile handhelds*, pages 61–62, Barcelona, Spain, Aug. 2009. doi:10.1145/1592606.1592621.

[7] Y. Chon, E. Talipov, H. Shin, and H. Cha. Mobility prediction-based smartphone energy optimization for everyday location monitoring. In *Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems*, SenSys '11, pages 82–95, Seattle, WA, USA, Nov. 2011. doi:10.1145/2070942.2070952.

[8] I. Constandache, R. R. Choudhury, and I. Rhee. Towards Mobile Phone Localization without War-Driving. In *Proceedings of IEEE INFOCOM 2010*, pages 1–9, San Diego, CA, USA, Mar. 2010. doi:10.1109/infcom.2010.5462058.

[9] I. Constandache, S. Gaonkar, M. Sayler, R. R. Choudhury, and L. Cox. EnLoc: Energy-efficient localization for mobile phones. In *Proceedings of INFOCOM 2009 Mini Conference*, pages 2716–2720, Rio De Janeiro, Brazil, Apr. 2009. doi:10.1109/infcom.2009.5062218.

[10] U. Deshpande, T. Henderson, and D. Kotz. Channel Sampling Strategies for Monitoring Wireless Networks. In *2006 4th International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks*, pages 1–7, Boston, MA, USA, 2006. doi:10.1109/wiopt.2006.1666486.

[11] U. Deshpande, C. McDonald, and D. Kotz. Coordinated Sampling to Improve the Efficiency of Wireless Network Monitoring. In *Proceedings of the Fifteenth IEEE International Conference on Networks (ICON)*, pages 353–358. IEEE Computer Society Press, Nov. 2007. doi:10.1109/ICON.2007.4444112.

[12] U. Deshpande, C. McDonald, and D. Kotz. Refocusing in 802.11 Wireless Measurement. *Passive and Active Network Measurement*, pages 142–151, 2008. doi:10.1007/978-3-540-79232-1_15.

[13] N. Eagle and A. S. Pentland. Reality Mining: Sensing Complex Social Systems. *Personal Ubiquitous Comput.*, 10(4):255–268, 1 Mar. 2006. doi:10.1007/s00779-005-0046-3.

[14] Facebook. Facebook, 28 Mar. 2014.

[15] Facebook. Internet.org, 28 Mar. 2014. Online at http://internet.org/.

[16] H. Falaki, R. Mahajan, S. Kandula, D. Lymberopoulos, R. Govindan, and D. Estrin. Diversity in smartphone usage. In *Proceedings of the 8th international conference on Mobile systems, applications, and services*, MobiSys '10, pages 179–194, San Francisco, California, USA, 2010. doi:10.1145/1814433.1814453.

[17] Foursquare. Foursquare, 27 Mar. 2014. Online at http://foursquare.com.

[18] Google. Google Play: Accupedo Pedometer. Online at http://play.google.com/store/apps/details?id=com.corusen.accupedo.te.

[19] Google. Android Developer Tools — Android Developers, 2 Mar. 2014. Online at http://developer.android.com/tools/help/adt.html.

[20] Google. ATAP Project Tango Google, 24 Feb. 2014. Online at http://www.google.com/atap/projecttango/.

[21] Google. Battery Stats Plus - Android Apps on Google Play, 28 Mar. 2014. Online at http://play.google.com/store/apps/details?id=com.rootuninstaller.bstats.

[22] Google. Getting Started with Android Studio — Android Developers, 2 Mar. 2014. Online at http://developer.android.com/sdk/installing/studio.html.

[23] Google. Google Maps, 28 Mar. 2014. Online at http://www.google.com/maps/preview.

[24] Google. Google Now, 27 Mar. 2014. Online at http://www.google.co.uk/landing/now/.

[25] c. T. Google Play. Battery Monitor Widget - Android Apps on Google Play, 22 Feb. 2014. Online at http://play.google.com/store/apps/details?id=ccc71.bmw&hl=en.

[26] S. K. Google Play. BetterBatteryStats - Android Apps on Google Play, 22 Feb. 2014. Online at http://play.google.com/store/apps/details?id=com.asksven.betterbatterystats&hl=en_GB.

[27] T. Hossmann, C. Efstratiou, and C. Mascolo. Collecting Big Datasets of Human Activity One Checkin at a Time. In *Proceedings of the 4th ACM International Workshop on Hot Topics in Planet-scale Measurement*, HotPlanet '12, pages 15–20, Low Wood Bay, Lake District, UK, 2012. doi:10.1145/2307836.2307842.

[28] C. K. Hsieh, H. Falaki, N. Ramanathan, H. Tangmunarunkit, and D. Estrin. Performance Evaluation of Android IPC for Continuous Sensing Applications. *SIGMOBILE Mob. Comput. Commun. Rev.*, 16(4):6–7, Feb. 2013. doi:10.1145/2436196.2436200.

[29] A. Inc. Apple - iOS 7 - Maps, 28 Mar. 2014. Online at http://www.apple.com/ios/maps/.

[30] A. Inc. iOS: Understanding iBeacon, 22 Mar. 2014. Online at http://support.apple.com/kb/HT6048.

[31] M. S. Inc. Monsoon Solutions, 22 Feb. 2014. Online at http://www.msoon.com/LabEquipment/PowerMonitor/.

[32] Q. Inc. Trepn Profiler - Qualcomm Developer Network, 22 Feb. 2014. Online at http://developer.qualcomm.com/mobile-development/increase-app-performance/trepn-profiler.

[33] Y. Kawahara, H. Kurasawa, and H. Morikawa. Recognizing User Context Using Mobile Handsets with Acceleration Sensors. In *Portable Information Devices, 2007. PORTABLE07. IEEE International Conference on*, pages 1–5. IEEE, May 2007. doi:10.1109/portable.2007.12.

[34] 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, pages 43–56, Zürich, Switzerland, Nov. 2010. doi:10.1145/1869983.1869989.

[35] C. KisMA. KisMAC - Welcome, 22 Mar. 2014. Online at http://kismac-ng.org/.

[36] M. B. Kjaergaard, J. Langdal, T. Godsk, and T. Toftkjaer. EnTracked: energy-efficient robust position tracking for mobile devices. In *Proceedings of the 7th international conference on Mobile systems, applications, and services*, MobiSys '09, pages 221–234, Kraków, Poland, 2009. doi:10.1145/1555816.1555839.

[37] J. R. Kwapisz, G. M. Weiss, and S. A. Moore. Activity recognition using cell phone accelerometers. *SIGKDD Explor. Newsl.*, 12(2):74–82, Mar. 2011. doi:10.1145/1964897.1964918.

[38] X. Li, H. Cao, E. Chen, and J. Tian. Learning to Infer the Status of Heavy-Duty Sensors for Energy-Efficient Context-Sensing. *ACM Trans. Intell. Syst. Technol.*, 3(2), Feb. 2012. doi:10.1145/2089094.2089111.

[39] H. Lu, J. Yang, Z. Liu, N. D. Lane, T. Choudhury, and A. T. Campbell. The Jigsaw continuous sensing engine for mobile phone applications. In *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*, SenSys '10, pages 71–84, Zürich, Switzerland, 2010. doi:10.1145/1869983.1869992.

[40] E. Miluzzo, N. D. Lane, S. B. Eisenman, and A. T. Campbell. CenceMe: injecting sensing presence into social networking applications. In *Proceedings of the 2nd European conference on Smart sensing and context*, EuroSSC'07, pages 1–28, Kendal, England, 2007. Online at http://portal.acm.org/citation.cfm?id=1775379.

[41] M. Musolesi, M. Piraccini, K. Fodor, A. Corradi, and A. T. Campbell. Supporting Energy-Efficient Uploading Strategies for Continuous Sensing Applications on Mobile Phones Pervasive Computing. In P. Floréen, A. Krüger, and M. Spasojevic, editors, *Proceedings of the 8th International Conference on Pervasive Computing*, Lecture Notes in Computer Science, pages 355–372, Helsinki, Finland, May 2010. doi:10.1007/978-3-642-12654-3_21.

[42] S. Nath. ACE: exploiting correlation for energy-efficient and continuous context sensing. In *Proceedings of the 10th international conference on Mobile systems, applications, and services*, MobiSys '12, pages 29–42, Low Wood Bay, Lake District, UK, 2012. doi:10.1145/2307636.2307640.

[43] J. G. Park, B. Charrow, D. Curtis, J. Battat, E. Minkov, J. Hicks, S. Teller, and J. Ledlie. Growing an organic indoor location system. In *Proceedings of the 8th international conference on Mobile systems, applications, and services*, MobiSys '10, pages 271–284, San Francisco, California, USA, 2010. doi:10.1145/1814433.1814461.

[44] U. S. Patent. Variable device graphical user interface of Apple Inc.,, 14 Jan. 2014. Online at http://patft.uspto.gov/netacgi/nph-Parser?Sect1=PTO2&Sect2=HITOFF&u=%2Fnetahtml%2FPTO%2Fsearch-adv.htm&r=1&p=1&f=G&l=50&d=PTXT&S1=(715%2F866.CCLS.+AND+20140114.PD.)&OS=ccl/715/866+and+isd/1/14/2014&RS=(CCL/715/866+AND+ISD/20140114).

[45] B. Priyantha, D. Lymberopoulos, and J. Liu. Enabling Energy Efficient Continuous Sensing on Mobile Phones with LittleRock. In *Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks*, IPSN '10, pages 420–421, Stockholm, Sweden, 2010. doi:10.1145/1791212.1791285.

[46] K. K. Rachuri, C. Mascolo, M. Musolesi, and P. J. Rentfrow. SociableSense: exploring the trade-offs of adaptive sampling and computation offloading for social sensing. In *Proceedings of the 17th annual international conference on Mobile computing and networking*, MobiCom '11, pages 73–84, Las Vegas, NV, USA, Sept. 2011. doi:10.1145/2030613.2030623.

[47] K. K. Rachuri, M. Musolesi, and C. Mascolo. Energy-Accuracy Trade-offs in Querying Sensor Data for Continuous Sensing Mobile Systems. Online at http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.172.7852, visited .

[48] V. Radu, J. Li, L. Kriara, M. K. Marina, and R. Mortier. Poster: A Hybrid Approach for Indoor Mobile Phone Localization. In *Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services*, MobiSys '12, pages 527–528, Low Wood Bay, Lake District, UK, 2012. doi:10.1145/2307636.2307717.

[49] V. Srinivasan and T. Phan. An Accurate Two-tier Classifier for Efficient Duty-cycling of Smartphone Activity Recognition Systems. In *Proceedings of the Third International Workshop on Sensing Applications on Mobile Phones*, PhoneSense '12, Toronto, Ontario, Canada, 2012. doi:10.1145/2389148.2389159.

[50] N. Store. Nokia Store: Download Nokia Energy Profiler and many other games, wallpaper, ringtones and phone apps on your Nokia phone, 22 Feb. 2014. Online at http://store.ovi.com/content/73969.

[51] H. Wang, S. Sen, A. Elgohary, M. Farid, M. Youssef, and R. R. Choudhury. No need to war-drive: unsupervised indoor localization. In *Proceedings of the 10th international conference on Mobile systems, applications, and services*, MobiSys '12, pages 197–210, Low Wood Bay, Lake District, UK, 2012. doi:10.1145/2307636.2307655.

[52] 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, pages 179–192, Kraków, Poland, 2009. doi:10.1145/1555816.1555835.

[53] Yelp. San Francisco Restaurants, Dentists, Bars, Beauty Salons, Doctors, 28 Mar. 2014. Online at http://www.yelp.com/.

[54] M. Youssef, M. A. Yosef, and M. El-Derini. GAC: Energy-Efficient Hybrid GPS-Accelerometer-Compass GSM Localization, 19 Apr. 2010. Online at http://arxiv.org/abs/1004.3174.

[55] L. Zhang, B. Tiwana, Z. Qian, Z. Wang, R. P. Dick, Z. M. Mao, and L. Yang. Accurate online power estimation and automatic battery behavior based power model generation for smartphones. In *Proceedings of the eighth IEEE/ACM/IFIP international conference on Hardware/software codesign and system synthesis*, CODES/ISSS '10, pages 105–114, Scottsdale, Arizona, USA, 2010. doi:10.1145/1878961.1878982.

A The complete set of experiment samples

A.1 Google Nexus 7

The logs for all samples are available in the *sensors_energy_measurement/experiments_results* directory.

Plain	902	902	841
Ambient Light	841	902	841
Bluetooth Low Energy*	782	842	781
Magnetic field:	782	782	721
Microphone	721	781	782
Bluetooth classic	781	721	722
Accelerometer:	781	721	661
Gyroscope	721	722	721
Bluetooth Low Energy**	722	721	661

A.2 HTC Flyer

Plain	30	270	270	270	271	270		
Bluetooth	181ˆ	270	30	60	270	271	270	241
Accelerometer	240	270	180ˆ	271	240	270		
Magnetic field	270	180ˆ	240	30	300‘	240	270	270
Light	240	240	271	270	270			
Microphone	240	240	240	300‘	270	240		
WiFiScan	240	241	236	248	240			
GPS	241	240	210	240	30	210		
Camera	121ˆ	240	180‘	210	241	240	210	

A.3 HTC Desire

Plain	250	253	251	201	250						
Bluetooth	251	251	200	250	250						
Light Sensor**	251	251	251	200	451ˆ	451ˆ	200				
Proximity*	251	501ˆ	451ˆ	200	501ˆ	463ˆ	251**	251	403ˆ	401ˆ	200
Accelerometer	251	451ˆ	200	401ˆ	200	250	250				
Magnetic Field	201	250	201	400ˆ	200	253					
Microphone	400ˆ	200	200	250	200	450ˆ	250				
WifiScan	150	300ˆ	251	201	250	151					
Camera	150	200	201	200	200						
GPS	200	201	150	200	151						
Light Sensor*	501ˆ	451ˆ	501ˆ	451ˆ	502ˆ						

B Media codecs for sensor measurement applications

B.1 Camera application

Parameter	Google Nexus 7	HTC desire	HTC flyer
Codec	H263	H263	H263
Resolution	176&144	176&144	176&144
Frame rate	15	12	10

B.2 Microphone application

Parameter	Google Nexus 7	HTC desire	HTC flyer
Codec	AMR narrow band(samr)	AMR narrow band(samr)	AMR narrow band(samr)
Channels	mono	1	1
Sample rate	8000 Hz	8000 Hz	8000 Hz
Bits per sample	32	16	16

C Ethics form

UNIVERSITY OF ST ANDREWS

SCHOOL OF COMPUTER SCIENCE
PRELIMINARY ETHICS SELF-ASSESSMENT FORM

This Preliminary Ethics Self -Assessment Form is to be conducted by the researcher, and completed in conjunction with the Guidelines for Ethical Research Practice. All staff and students of the School of Computer Science must complete it prior to commencing research.

This Form will act as a formal record of your preliminary ethics considerations.

PROJECT TYPE (please ✓)

☐ Staff
☑ Undergraduate
☐ Postgraduate
  ☐ MSc
  ☐ PhD

PROJECT TITLE

Energy-efficient sensing with Android smartphones	
Name of researcher(s)	Maciej Kukla
Name of supervisor (for student research)	Tristan Henderson

OVERALL ASSESSMENT (to be signed after questions, overleaf, have been completed)

Self-audit has been conducted (Please ✓)

☑ YES ☐ NO

Ethical Issues (Please ✓)

☐ There are **NO** ethical issues raised by this project
☑ There are ethical issues raised by this project

Signed Kukla **Print Name** MACIEJ KUKLA **Date** 27th Sept., 2013
(Student Researcher(s), if applicable)

Signed T Hen **Print Name** TRISTAN HENDERSON **Date** 27th Sept., 2013
(Lead Researcher or Supervisor)

This form must be date stamped and held in the files of the School Ethics Committee. The School Ethics Committee will be responsible for monitoring assessments.

Computer Science Preliminary Ethics Self-Assessment Form

Research with human subjects

1. Does your research involve human subjects or have potential adverse consequences for human welfare and wellbeing? **YES** **NO**

For example:

Will you be surveying, observing or interviewing human subjects?

Will you be testing any systems or programs on human subjects?

Will you be collecting data from computers or networks used by human subjects?

If YES, *full ethics review may be required*

If NO, *go to question 16, then sign this form and return to the School Ethics Committee Secretary.*

Potential physical or psychological harm, discomfort or stress

2. Will your participants be exposed to any risks greater than those encountered in their normal working life? **YES** **NO**

If YES, *full ethics review required*

Investigators have a responsibility to protect participants from physical and mental harm during the investigation. The risk of harm must be no greater than in ordinary life. Areas of potential risk that require ethical approval include, but are not limited to, investigations that occur outside usual laboratory areas, or that require participant mobility (e.g., walking, running, use of public transport), unusual or repetitive activity or movement, that use sensory deprivation (e.g., ear plugs or blindfolds), bright or flashing lights, loud or disorienting noises, smell, taste, vibration, or force feedback.

3. Do your experimental materials comprise software running on non-standard hardware? **YES** **NO**

If YES, *full ethics review required*

Participants should not be exposed to any risks associated with the use of non-standard equipment: anything other than pen-and-paper, standard PCs, mobile phones, and PDAs is considered non-standard.

4. Will you offer incentives to your participants? **YES** **NO**

If YES, *full ethics review required*

The payment of participants must not be used to induce them to risk harm beyond that which they risk without payment in their normal lifestyle.

Data Protection

5. Will any data collected from the participants not be stored in a secure and anonymous form? **YES** **NO**

If YES, *full ethics review required*

All participant data (hard-copy and soft-copy) should be stored securely, and in anonymous form.

6. Will you collect more data than are required for your immediate experimental hypotheses? **YES** **NO**

If YES, *full ethics review required*

Any collection of personal data should be adequate, relevant and not excessive in relation to the purposes for the collection.

Informed consent

7. Will participants participate without their explicit agreement to participate, or their explicit agreement that their data can be used in your project? **YES NO**

If YES, *full ethics review required*

If the results of the experiments are likely to be used beyond the term of the project (for example, the software is to be deployed, or the data is to be published), then signed consent is necessary. A separate consent form should be signed by each participant. Otherwise, verbal consent is sufficient, and should be explicitly requested in the introductory script.

8. Will you withhold any information about your experiment or materials from your participants? **YES NO**

If YES, *full ethics review required*

Withholding information or misleading participants is unacceptable if participants are likely to object or show unease when debriefed.

9. Are any of your participants under the age of 18? **YES NO**

If YES, *full ethics review required*

Working with human subjects under the age of 18 requires you to obtain an Enhanced Disclosure from Disclosure Scotland.

10. Do any of your participants have an impairment that may limit their understanding or communication? **YES NO**

If YES, *full ethics review required*

Additional consent is required for participants with impairments.

11. Are you or your supervisor in a position of authority or influence over any of the participants? **YES NO**

If YES, *full ethics review required*

A position of authority or influence over any participant must not be allowed to pressurise participants to take part in, or remain in, any experiment.

12. Will participants participate without being informed that they can withdraw at any time? **YES NO**

If YES, *full ethics review required*

All participants have the right to withdraw at any time during the investigation. They should be told this in the introductory script.

13. Will participants participate without being informed of your contact details and those of your supervisor? **YES NO**

If YES, *full ethics review required*

All participants must be able to contact the investigator after the investigation. They should be given the details of both student and module co-ordinator or supervisor as part of the debriefing.

14. Will any participants not have the opportunity to discuss the experiment and answer questions at the end of your experimental session? **YES NO**

If YES, *full ethics review required*

You must provide the participants with sufficient information in the debriefing to enable them to understand the nature of the investigation.

Conflicts of interest

15. Do any conflicts of interest arise? **YES (NO)**

If YES, *full ethics review required*

For example:

Might research objectivity be compromised by sponsorship?

Might any issues of intellectual property or roles in research be raised?

Funding

16. Is your research funded externally? **YES (NO)**

If YES, is the funder missing from the 'currently automatically approved' list on the UTREC website?

YES NO

If YES, you will need to submit a Funding Approval Application as per instructions on the UTREC website.

You (and, where appropriate, your supervisor) should now sign this form and return it to the School Ethics Committee Secretary.

Check whether you need to submit a full Ethics Application Form as well. If you have a supervisor, also check with them. If still in doubt, please contact the School Ethics Committee for advice.

Where appropriate, your experiments must use information sheets based on the examples provided on the Ethics website (http://www.cs.st-andrews.ac.uk/ethics), and these should be submitted with your final report.