Poster: SensingKit: a multi-platform mobile sensing framework for large-scale experiments.

Katevas, K; Haddadi, H; Tokarchuk, L

Poster: SensingKit – A Multi-Platform Mobile Sensing Framework for Large-Scale Experiments

Kleomenis Katevas†, Hamed Haddadi†⋄, Laurissa Tokarchuk†
†Queen Mary University of London, ⋄Qatar Computing Research Institute
{k.katevas, hamed.haddadi, laurissa.tokarchuk}@qmul.ac.uk

ABSTRACT

With the rapid rise in variety of available smartphones today and their rich sensing capabilities, there is an increasing interest in using mobile sensing in large-scale experiments and commercial applications. Motivated by the lack of a universal, multi-platform library, in this paper we present *SensingKit*, an efficient, open-source, client-server system that supports both iOS and Android mobile devices. SensingKit is capable of continuous sensing the device's motion (Accelerometer, Gyroscope, Magnetometer), location (GPS) and proximity to other smartphones (Bluetooth Smart). The data are temporarily saved to the device's memory and transmitted to a server for further analysis over any Internet connection. We believe that this platform will be beneficial to all researchers and developers who need to perform mobile sensing in their applications and experiments.

Categories and Subject Descriptors

C.4 [**Performance of Systems**]: Measurement techniques; D.2.13 [**Software Engineering**]: Reusable Software—*Reusable libraries*

General Terms

Experimentation, Measurement, Performance

Keywords

Mobile sensing, Spatio-temporal data, Motion data, iOS, Android

1. INTRODUCTION

The ubiquity of smartphones as well as the variety of their onboard sensors have enabled the automated acquisition of large scale data, inspiring a wealth of research opportunities. Mobile operating systems such as Android and iOS provide application programming interfaces (APIs) to access these sensors. Lane et al. [2] in a recent survey paper discussed the importance of continuous sensing among different mobile platforms. Various mobile sensing frameworks have been designed that provide continuous sensing, like MobiSens [4], EmotionSense [3] and Funf [1]. However, these platforms are currently limited to work on Android or Nokia Maemo phones, limiting the sampling space of users participating in different studies. Since Android and iOS are the two main players in the mobile ecosystem, there is a clear need for supporting continuous sensing in these two mobile environments.

In this work, we present an early prototype of SensingKit, a new mobile sensing framework compatible with iOS and Android devices. SensingKit enables capturing motion (Accelerometer, Gyroscope and Magnetometer), location (GPS) and proximity (Bluetooth Smart) data and transmitting them to a server over any Internet connection. Since the two operating systems are equipped with sensor fusion techniques, both the raw measurements, and fussed data like Linear Acceleration, Gravity and Rotation are supported. Furthermore, SensingKit can also be configured to capture user's natively-labelled activity in supported devices, classified as *standing*, *walking*, *running*, *driving* in iOS, with the addition of *cycling* in Android.

To achieve this, we have developed two open-source libraries, *SensingKit-iOS* and *SensingKit-Android*, that researchers can include into their custom made applications with only a few lines of code. In addition, a server framework has been developed, referred to as the *SensingServer*, which is responsible for receiving the data, synchronising them and finally exporting them into CSV or JSON format.

Beside the multi-platform characteristic, SensingKit has some unique features that are not available in other sensing libraries. It supports the Bluetooth Smart (4.0) specification for capturing the proximity between devices, a feature that has significantly reduced power consumption and highest sampling rate compared to the classic Bluetooth. Furthermore, in order to avoid timing issues when the user, or even when the device itself changes the system time, the timing in the sensor measurements depends on the device's CPU time base register rather than the system's clock. In some modern devices like the iPhone 5S and Nexus S5 that are equipped with a motion co-processor, the library is capable of using the processor's activity recognition, having only a minimum affect on the device's battery life. Finally, the server framework is equipped with a plug-in architecture that can automate some of the usual data processing before the extraction (e.g., perform data interpolation, produce magnitude vector, etc).

MobiCom'14, MobiCom'14, Sep 07-11 2014, Maui, HI, USA
ACM 978-1-4503-2783-1/14/09.
http://dx.doi.org/10.1145/2639108.2642910.

Our objective in this work is to provide an easy-to-use sensing framework that developers and researchers can use to provide continuous sensing in iOS and Android applications. In Section 2, we present the System Architecture and technical details of the framework. The system was evaluated, as reported in Section 3, by measuring the battery consumption of SensingKit running on three mobile phones. In Section 4 we present the conclusions and discuss the future research in this space.

2. PLATFORM ARCHITECTURE

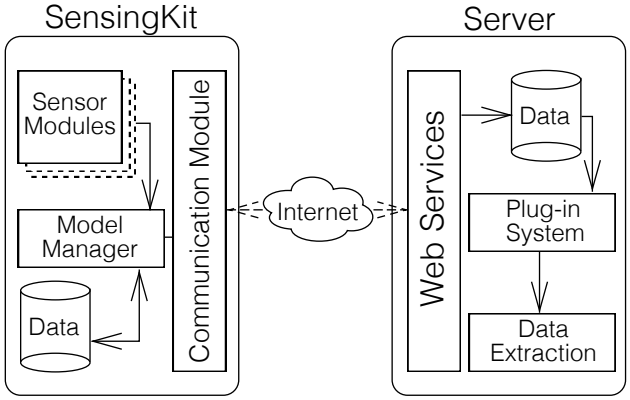


Figure 1: SensingKit System Architecture

SensingKit is a client/server system, consisting of two separate software frameworks: the *client library*, located in the users mobile devices and the *server framework*. Figure 1 gives an overview of the system architecture.

Client libraries for Android and iOS

SensingKit is a modular mobile framework developed in the native programming language of each platform (Java for the Android and Objective-C for the iOS version). It supports mobile devices running iOS v7.0 and Android v2.3.3[1] and above.

For every sensing category, a sensing module exists in SensingKit. Table 1 presents the available sensing modules of the framework. It is important to mention that due to the modular design of this library, it is easy to develop a new module and extend its sensing capabilities.

Table 1: Modules of SensingKit

Module	Information sensed
Motion	Linear Acceleration, Gravity, Rotation, Activity Classification.*
Location	Outdoor location using GPS sensor.
Proximity	Proximity sensing using Bluetooth Smart (4.0).
Battery	Power consumption of the device.

*Raw data from the 3-axis Accelerometer, Gyroscope and Magnetometer sensors are also provided.

For proximity sensing, SensingKit uses the new Bluetooth Smart (4.0) proximity profile. This profile allows to broadcast a device's presence, scan for other devices and estimate the distance between them. Bluetooth Smart is only fully supported in the latest Android *L* mobile operating system. Android 4.3 devices are only limited to scan and connect to other devices (*Observer and Central mode*) and not to advertise its presence to the nearby devices (*Peripheral mode*).

The library collects the data from the sensing modules and saves them to the device's memory using ModelManager module. When a Wi-Fi connection is available it submits them on the server through CommunicationManager. If the data remain to the memory for longer than 24 hours, it will try to submit them over a Cellular network.

Server platform

The server platform is built using the Python programming language. It is responsible for collecting the data captured from a client app, over an Internet connection. It consists of three main components: the *web-services*, the *plug-in system* and the *data extraction module*.

The web-services component is responsible for providing the appropriate communication methods to the client libraries. It provides methods for assigning a unique identification to each client, for sending the server's time so that the data can easily be synchronised later, and finally, for receiving the collected data.

The plug-in system provides a Python interface for writing custom plug-ins that pre-process the data before extraction. This is useful for the researcher as it can perform some initial automated data processing. The mandatory data-sync plug-in is pre-installed and processes the data so that they have the same timing with minimum delay between devices. Further optional plug-ins such as the *interpolation* and *magnitude* are also available.

Finally, the data extraction module is used to extract the data in various formats such as JSON and CSV.

3. USE-CASE DEMONSTRATION

In order to demonstrate a sample use of our sensing platform, we measured the battery life performance while using SensingKit in three mobile devices: an iPhone 5S running iOS 7.1.2, a Google Nexus 4 running Android KitKat 4.4.4 and a Samsung Galaxy S2 running Android Jelly Bean 4.1.2. All devices were set into Flight Mode, having Wi-Fi and Cellular connectivity disabled. No application or process rather than SensingKit was running in the background. Since Galaxy S2 devices are not equipped with a Bluetooth Smart (4.0) radio, proximity sensing was not possible with that device. Finally, due to limitations of the current Android platform (Android KitKat 4.4.4) the Nexus 4 device was only able to scan for other devices and not to broadcast its presence. Table 2 shows the specifications of each device, including the highest sampling rate that is supported.

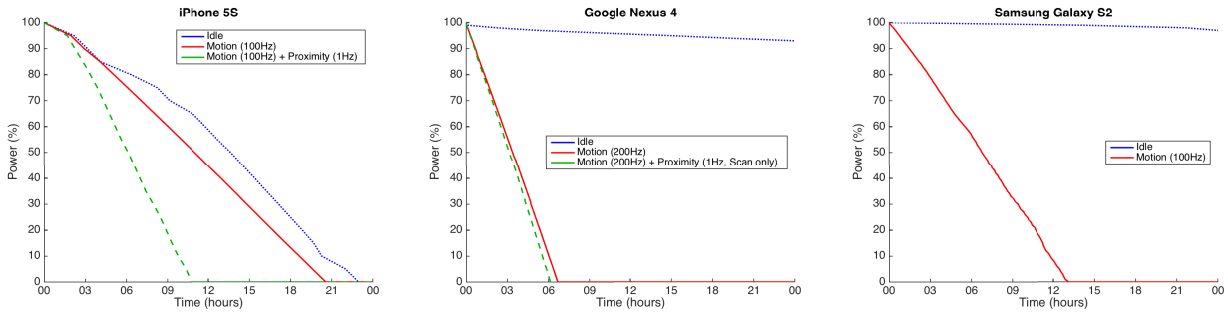
Figure 2 shows the energy consumption of SensingKit running for 24 hours on the three mobile devices described above. For each device, we show the consumption of the library while sensing motion, proximity and battery levels. In addition, we visualise the library running in "idle" mode, when it only senses the battery levels. Due to limitations in the iOS platform, it is only possible to read the iPhone's battery level with a 5% resolution whereas in Android the resolution is 1%.

The results show that the Galaxy S2 lasted for 13.1 hours while capturing motion in 100Hz but not proximity. The

[1]Android *L*/iOS 7 and Bluetooth 4.0 radio is required to use Bluetooth Smart proximity sensing.

Table 2: Device Specification

Device	Op. System	Processor	Memory	Battery	Bluetooth	Max Sampling
iPhone 5S	iOS 7.1.2	1.3 Ghz Dual-core	1GB	1560 mAh	4.0	100Hz
Nexus 4	KitKat 4.4.4	1.5 Ghz Quad-core	2GB	2100 mAh	4.0	200Hz
Galaxy S2	Jelly Bean 4.1.2	1.2 Ghz Dual-core	1GB	1650 mAh	3.0+HS	100Hz



(a) (b) (c)

Figure 2: Battery consumption of SensingKit running on an iPhone 5S, Google Nexus 4 and Samsung Galaxy S2 smartphone.

iPhone 5S was capturing motion in the same sampling rate (100Hz), and lasted 11.5 hours with proximity sensing enabled (1Hz) and 20.3 hours with proximity sensing disabled. Finally, the Nexus 4 device was measuring motion with double the sampling rate of the other two devices (200Hz), and lasted 6.1 hours with proximity sensing enabled (1Hz) and 6.4 hours with proximity sensing disabled.

The effect of SensingKit in the iPhone's battery life is quite noticeable when it runs in "idle" mode, as the library keeps the device's CPU usage to 6-7%, just for monitoring the battery in the background. In addition, proximity sensing also has an impact on the battery life, showing a 43.3% decrease on the iPhone 5S whereas in the case of Nexus 4, the device only works in *Observer mode* and the battery slightly decreases by 4.7%.

It is important to mention that Figure 2 only represents the battery consumption of the specific mobile devices listed in Table 2 and should not be viewed as a comparison between the three devices.

4. CONCLUSIONS AND FUTURE WORK

In this work, we have presented an early implementation of a continuous sensing system that works in both Android and iOS environments. A first prototype of this system has already been developed and is being used in related crowd-sensing experiments. We plan to continue the development of this framework, extend its sensing capabilities, evaluate it and release it to the public in open-source in the near future. We believe that this work will be beneficial for researchers willing to conduct large-scale experiments using mobile sensing.

More information about SensingKit as well as the complete source-code will be made available on www.sensingkit.org.

5. ACKNOWLEDGEMENTS

This work is supported by funding from the UK Defence Science and Technology Laboratory.

6. REFERENCES

[1] N. Aharony, W. Pan, C. Ip, I. Khayal, and A. Pentland. Social fMRI: Investigating and shaping social mechanisms in the real world. *Pervasive and Mobile Computing*, 7(6):643–659, 2011.

[2] N. D. Lane, E. Miluzzo, H. Lu, D. Peebles, T. Choudhury, and A. T. Campbell. A survey of mobile phone sensing. *Communications Magazine, IEEE*, 48(9):140–150, 2010.

[3] K. K. Rachuri, M. Musolesi, C. Mascolo, P. J. Rentfrow, C. Longworth, and A. Aucinas. EmotionSense: a mobile phones based adaptive platform for experimental social psychology research. In *Ubicomp '10: Proceedings of the 12th ACM international conference on Ubiquitous computing*, page 281, New York, USA, Sept. 2010.

[4] P. Wu, J. Zhu, and J. Y. Zhang. MobiSens: A Versatile Mobile Sensing Platform for Real-World Applications. *Mobile Networks and Applications*, 18(1):60–80, Nov. 2012.