Poster Abstract: High reliability Android application for multidevice multimodal mobile data acquisition and annotation

Mathias Ciliberto, Francisco Javier Ordoñez Morales, Hristijan Gjoreski, Daniel Roggen
Wearable Sensor Technologies Lab
Sensor Technologies Research Centre
University of Sussex, Brighton
{m.ciliberto,h.gjoreski,droggen}@sussex.ac.uk,
fjordonez@gmail.com

Sami Mekki, Stefan Valentin
Mathematical and Algorithmic Sciences Lab
PRC, Huawei Technologies France
{sami.mekki,stefan.valentin}@huawei.com

ABSTRACT

We have completed the collection of one of the richest accurately annotated mobile dataset of modes of transportation and locomotion. To do this, we developed a highly reliable Android application called DataLogger capable of recording multisensor data from multiple synchronized smartphones simultaneously. The application allows real-time data annotation. We explain how we designed the app to achieve high reliability and ease of use. We also present an evaluation of the application in a big-data collection (750 hours, 950 GB of data, 17 different sensor modalities), analysing the data loss (less than 0.4‰) and battery consumption (≈ 6% on average per hour). The application is available as open source.

CCS CONCEPTS

• **Human-centered computing** → *Ubiquitous and mobile computing systems and tools*;

KEYWORDS

High reliability, Android, Data collection, Activity recognition

1 INTRODUCTION

In this paper, we present DataLogger, our application for high reliability multi-modal mobile data collection. To the best of our knowledge this is the first one that allows synchronized multi-device (up to 4) recordings. It also includes a wider configurable set of sensors compared to the two main existing sensing frameworks Funf [3] and Mobile Sensor Framework (MSF) [4], and, more importantly, it allows the users to label the data in real time. These features are fundamental to most effectively exploit the availability of data collection participants and gather as much data as possible (here from 4 body location) with accurate annotations. This paper explains how we addressed the real-time annotation and multi-device synchronisation challenges. Eventually, we present an evaluation of the application during a terabyte-sized data collection in term of trustworthiness and battery life.

SenSys '17, November 6–8, 2017, Delft, Netherlands

ACM ISBN 978-1-4503-5459-2/17/11…$15.00
https://doi.org/10.1145/3131672.3136977

2 APPLICATION FEATURES

The application is capable of logging accelerometer, gyroscope, magnetometer, linear acceleration, orientation, gravity, temperature, light sensor, pressure, humidity, location, satellites, network cells, WiFi networks, battery level and the audio from the microphone, each one with a configurable sampling frequency.

The application saves the sensor data in a matrix format in text files, on the smartphone flash storage. Each line corresponds to a sample that starts with two timestamps then followed by the sensor data, with one column per channel. The first timestamp is the epoch time (or Unix time) of the moment when the line is written to the log. The second one is the time in nanoseconds when the Android system acquired the sensor data.

DataLogger is used for labelling the recorded data in real time. The labels are saved on the internal storage too, with an indication of the starting and ending time as well as with a numeric value corresponding to each label. The set of labels can be configured in the code by modifying the corresponding XML file.

2.1 Synchronization and dependability

DataLogger can record data from up to 4 synchronized devices. Typically the devices are placed on different body locations in order to develop activity recognition models that can adapt to different positions of the smartphones [7]. The master smartphone, typically the one on the hand, instructs the slaves as to the status of the logging (on/off), the current label and the time offset to synchronize the data among all the phones. This is done through a continuous Bluetooth connection implemented using a customized version of the Android-Multi-Bluetooth-Library [1].

The application for master and slave is the same, but the user interface changes accordingly. The interface of the master displays the controls to start the data collection, to select the label and to start the annotation. The slaves' interface only presents information about the logging status, the connection status to the master and a button to force the reconnection.

The data synchronization during the logging is fundamental to obtain a valuable dataset. Therefore, the master sends a keep-alive packet every 30 seconds to the slaves. This packet includes the status of the logging, the current label and the nanosecond time of the master. The nanosecond time is used by the slave to compute the difference between the received time and their own, in order to

save this offset in the log files. These keep-alive packets assure also that the system does not close the Bluetooth connection. In case the Bluetooth connection is dropped during the data logging despite all the precautions, the master device vibrates to notify the user that something may happened to the slaves which are not usually under constant supervision.

We designed DataLogger to make it robust and user-proof. The application runs in background, using what in Android is called "foreground service". This is a service displays a persistent notification, in our case the logging status information, excluding itself from the processes to be killed by the system when the memory runs out. This service handles "sensor-collectors" and the Bluetooth communication. A sensor-collector is an object that initializes the sensor listener and logs on file each sensor event.

The application is automatically relaunched at the boot of the devices, detecting the event "RECEIVE_BOOT_COMPLETED" broadcasted by the Android system. The logging status of the application (idle or collecting data) and the current label are always saved in the local storage of the device in order to restore the last state if the application is relaunched after a reboot.

A new log text file for the sensor data and a new audio file are created every 5 minutes, reducing the risk that reboots or other problems would cause a larger data loss while writing the data on the disk. The interval is configurable.

We designed the UI to minimise inadvertent user entries. The buttons to start the data collection and the labelling requires a long press to be activated in order to avoid accidental start/stop of these operations. For each label some sub-labels can be specified. These are pre-determined in order to avoid wrong combinations of labels and sub-labels. A simple flag-event mechanism has been implemented that allows the user to point out outstanding situations during daylong recordings, such as forgetting to start/stop a label, or a wrong label. These annotations are saved persistently on the internal storage and are used during an a posteriori label curation phase.

3 RELIABILITY AND BATTERY LIFE

We used DataLogger in a data collection that involved 3 participants, wearing 4 smartphones on 4 location (hand, torso, bag and hips) for 7 months. We collected more than 750 hours of data from 17 different sensors for a total of 950GB of raw data [6]. The participants were asked to perform 8 activities related to different transportation modes: staying still, walking, running, cycling, driving (or being in a car as passenger), travelling on a train, a bus or in the subway. Thanks to this data collection, we were able to study the reliability and the battery consumption of the application. In our data collection campaign, we used Huawei Mate 9 smartphones that run Android 7. This model is equipped with a 4000 mAh battery.

Despite all our precautions, we experienced few reboots of the phones during the data collection. Within 750 hours of collected data, the amount of lost data is lower than the 0.15‰ for each device. The hips' device appears as the most subject to reboot, likely because it is more subject to accidental long-pressing of the power off button.

We analysed the discharging rate (in percentage) per hour. The network interfaces, especially the mobile network module, and the display are the most power consuming [5]. In this context we

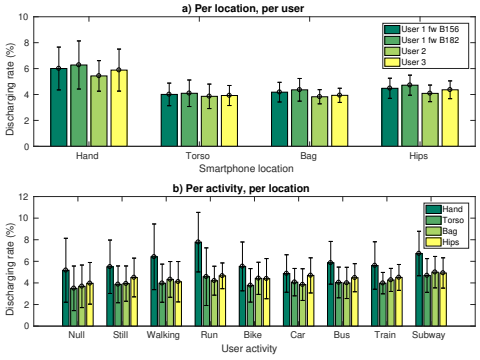

Figure 1: Discharging rate per hour

studied the discharging rate per location and per activity in order to evaluate the impact of these factors during the data collection.

Figure 1a displays the discharging rate per hour per location. During the data collection campaign, the smartphones of the User 1 updated their firmwares from the build B156 to B182, without changing the Android version. The update increased the sampling frequency of the inertial data (accelerometer, gyroscope and magnetometer) from 100 Hz to 200 Hz and it decreased the sampling frequency of the pressure sensor data from 100 Hz to 10 Hz. A slight increase in the power consumption can be noticed after the update. As expected the hand's device is the one with the highest power consumption, with a mean discharging rate of 5.9%, due to the interaction of the user during the real time labelling. Figure 1b presents the discharging rate per hour for each activity for each location of the smartphone on the body. The values are averaged among the users. Surprisingly, the subway is not the most power consuming activities as we expected according to [5]. Indeed, due to the lack of network in the underground, we hypothesized the stress of the GSM module would have increased substantially the power consumption.

4 CONCLUSION

We presented DataLogger as the first highly reliable tool for multimodal multi-device data collection and real-time annotation. The evaluation results show that even in the worst scenario, it can be used for up to 10 hours with a single power charge. Moreover, with only 0.4‰ of total data loss, it is a very dependable solution for big data collection. The code of the application is released under MIT License and it is available at [2]. The dataset will be released soon to the community. [1].

ACKNOWLEDGMENTS

This work was supported by HUAWEI Technologies within the project *"Activity sensing technologies for mobile user"*.

REFERENCES

[1] 2016. Android Multi-Bluetooth Library. (2016). https://github.com/arissa34/Android-Multi-Bluetooth-Library
[2] 2017. DataLogger source code. (2017). https://github.com/sussexwearlab/DataLogger
[3] N Aharony, A Gardner, C Sumter, and A Pentland. [n. d.]. Funf: Open sensing framework. ([n. d.]). http://funf.org/
[4] G Cardone, A Cirri, A Corradi, L Foschini, and D Maio. 2013. MSF: An efficient mobile phone sensing framework. *I. J. of Distributed Sensor Networks* (2013).
[5] A Carroll and G Heiser. 2010. An analysis of power consumption in a smartphone. *USENIX* (2010).
[6] H Gjoreski, M Ciliberto, F J Morales, D Roggen, S Mekki, and S Valentin. 2017. A Versatile Annotated Dataset for Multimodal Locomotion Analytics with Mobile Devices. *Sensys* (2017).
[7] U Maurer, A Smailagic, D.P. Siewiorek, and M Deisher. 2006. Activity recognition and monitoring using multiple sensors on different body positions. *BSN* (2006).

[1] www.shl-dataset.org