Introducing GaitLib: A Library for Real-time Gait Analysis in Smartphones

Michael M.A. Wu
mike.wu@alumni.ubc.ca

Oliver S. Schneider
oschneid@cs.ubc.ca

Idin Karuei
idin@cs.ubc.ca

Larissa A. Leong
larissa.leong@alumni.ubc.ca

Karon E. MacLean
maclean@cs.ubc.ca

Department of Computer Science
University of British Columbia
Vancouver, Canada

ABSTRACT

Modern smartphones are pervasive, powerful, and richly endowed with sensors. These have recently enabled smartphone use for gait analysis, a powerful resource for many applications including biometric identification and context-aware apps that motivate exercises. However, there is little support for software R&D with mobile gait analysis beyond basic sensing. Through a participatory design process, we developed GaitLib, a library for real-time gait analysis in smartphones. With on-board accelerometers and other sensors, GaitLib supports both cadence estimation and gait classification. The library is implemented on the Android platform, using Weka as the classification engine while supporting customizable gait analysis algorithms. An end user who participated in the design team used successive versions of the library in a series of studies, providing design input which was used to improve the library's functionality and usability. This library can support and stimulate future research in gait analysis and the development of innovative applications.

Author Keywords

open-source library; mobile; cadence; gait classification; Android

INTRODUCTION

As smartphones become more powerful and well resourced, they are increasingly present in our day-to-day life. This pervasiveness means context-aware applications, such as recognizing attributes in the wearer's gait, have great potential. Gait analysis plays an important role in, for example, medical monitoring [2], user identification to improve security [4], and exercise applications [3].

However, few mobile apps use gait information intelligently beyond basic sensing. This is in part due to the absence of any general gait analysis tools for developers, which makes doing analysis of any sophistication very challenging. Consequently, many applications of gait classification remain unexplored, given the limited resources of most mobile app development teams and academic researchers.

In this paper, we introduce GaitLib, an easy-to-use library for real-time gait analysis on smartphones. GaitLib currently supports two challenges in gait analysis: cadence estimation and gait classification. We provide a default algorithm for each, but support easy addition of alternative algorithms. This extensibility gives developers the flexibility to customize the approach they take to responding intelligently to users' gait. GaitLib is open-sourced and ready for development on commercially-available Android devices.

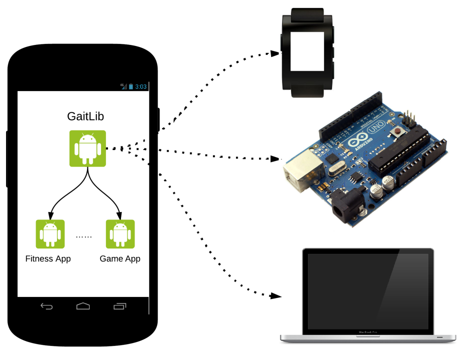

Figure 1. GaitLib can be used in the implementation of smartphone apps, smartwatch apps, Arduino programs, etc. Arrows indicate flow of real-time cadence and gait data; wireless connections are shown with dotted lines.

GaitLib was inspired by the need of our own group to reuse algorithms we developed for spatio-temporal guidance and exergame support. It was developed in a participatory design approach wherein architecture and development team members (referred to here as Dev) supported another author (the end-user member, or EU) who was extensively consulted on the design. We strove to make GaitLib robust, easy-to-use and extensible; using the library ourselves – a.k.a. "dogfooding", let us achieve this.

Prior to GaitLib, EU developed one of the default algorithms, RRACE [8] for cadence detection, and then used RRACE in subsequent experiments as a measurement for ground truth. During these experiments, EU used an earlier version of GaitLib to make his job easier and provided feedback to Dev. Later, EU used an updated version of GaitLib to build a demo for a conference. After the demo was built and piloted, a member of Dev conducted an open-ended post interview to get an idea of how well GaitLib worked for EU.

We will begin by discussing related studies that we consulted during the design of GaitLib and the novelty of GaitLib over similar systems. Then, we will detail our approach and describe the architecture of the library, followed by an overview the functionalities and the customizable features. We will conclude this paper with a discussion of the performance and usability of the library and some of our planned future works.

RELATED WORK

The three main types of sensor technologies that have been used in systems for gait analysis are cameras and machine vision, floor sensors, and wearable sensors [4]. Methods involving wearable sensors began by attaching multiple sensors to specific parts of user's body [1], and evolved to carrying a single three-axis accelerometer mounted on a cell phone, where the accuracy of gait classification (processed on a server) reached 80% [7]. Smartphone's on-board accelerometers have been shown to achieve accuracy comparable to that of conventional standalone accelerometers [10]. Thus, smartphones equipped with accelerometers are suitable for gait analysis.

Real-time on-device cadence estimation and gait classification were shown to be feasible in several studies. Using a single accelerometer mounted on the waist as the sensor, several gait cycle parameters, including cadence, can be estimated by an on-device PIC microcontroller in real time [13]. Gait classification was also achieved with high accuracy and orientation-independence [14], which is an important factor for making gait recognition on mobile phones practical.

The task of gait classification has been widely studied and thoroughly reviewed within a structured framework [11]. Two groups have developed systems for activity and gait classification on mobile phones [5, 12]. They used algorithms that allow the systems to build classifier models and perform classification in real time on a smartphone. Unlike GaitLib, theses systems are not development tools that software developers and researchers can readily use for their own projects.

APPROACH

Our approach centered around requirements gathering and iterative implementation of a tool to support EU, as a proxy for uses in future research planned by our group and by collaborators. EU was involved in the initial requirements gathering and provided feedback in each development iteration.

Key Requirements Gathered and Addressed

Cadence Estimation

Cadence is the measure of one's step rate, in steps per second. This information can loosely categorize the user's activity. GaitLib also takes into account the overall speed at which the user is moving, and from cadence and speed, stride length and distance can be determined.

Gait Classification

The goal of gait classification is to recognize the user's current gait and infer the current activity performed by the user. This provides a more specific description about the user's state than cadence value. Given a list of gaits and a corresponding classifier model, GaitLib performs classification on the device and returns the result.

Filtering

In signal processing, filtering is a technique that modifies the original signal to extract properties of the signal. GaitLib supports applying filters to the cadence values so that developers can further fine-tune the result.

Real-time

GaitLib provides gait analysis at a real-time latency suitable for interactive applications. Sensor data are continuously received and cached, while calculations and analyses are performed at a tunable sampling interval, generally around 1 second.

Extensibility

Besides custom filters, GaitLib supports using alternative classifier models for gait classification and different algorithms for cadence estimation. Developers can build custom models that are tailored to their specific needs while still using GaitLib's framework and low-level sensor management.

Development Platform

For its flexibility with OS accesses and compatibility with machine learning packages, we chose Android as the first development platform on which to implement the library. GaitLib interacts with Android to receive data from the sensors and to read and write files in the device storage. The classification algorithms are provided by Weka for Android [9], an adapted version of Weka 3: Data Mining Software [6].

Testing

GaitLib employs unit testing for internal functions. We built an example application to test for interactive scenarios (e.g., receiving sensor data) and EU provided real-world testing.

LIBRARY ARCHITECTURE

In this section, we present the architecture of the system. The overall structure of GaitLib is shown in Figure 2.

Primary Classes

The main class in GaitLib is `GaitAnalysis`. It handles starting and stopping gait analysis, defining parameters, directly accessing latest results of cadence estimation and gait classification. It contains a `CadenceDetector` and a `GaitClassifier`, which are responsible for estimating cadence and classifying gait, respectively.

Extensibility

The `CadenceDetector` and `GaitClassifier` classes are abstract. They contain all of the functions except those that are specific to the algorithms. Beyond the two included default implementations, additional custom algorithms can be integrated by extending one of the abstract classes.

IFilter Interface

GaitLib supports filtering cadence values with an `IFilter` interface, which can be added through `GaitAnalysis`.

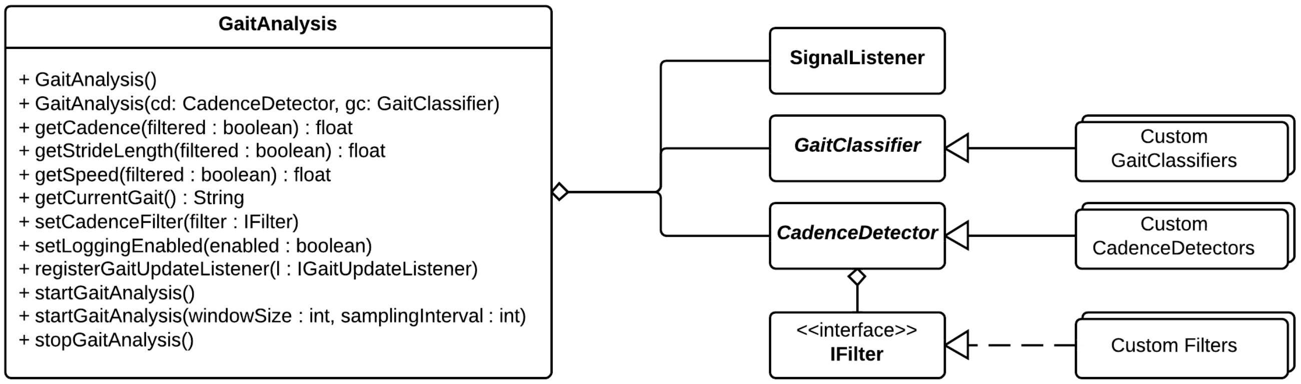
GaitAnalysis
+ GaitAnalysis()
+ GaitAnalysis(cd: CadenceDetector, gc: GaitClassifier)
+ getCadence(filtered : boolean) : float
+ getStrideLength(filtered : boolean) : float
+ getSpeed(filtered : boolean) : float
+ getCurrentGait() : String
+ setCadenceFilter(filter : IFilter)
+ setLoggingEnabled(enabled : boolean)
+ registerGaitUpdateListener(l : IGaitUpdateListener)
+ startGaitAnalysis()
+ startGaitAnalysis(windowSize : int, samplingInterval : int)
+ stopGaitAnalysis()
SignalListener
GaitClassifier
Custom GaitClassifiers
CadenceDetector
Custom CadenceDetectors
<<interface>> IFilter
Custom Filters

Figure 2. Class diagram of GaitLib.

GaitLib contains a number of frequently-used filters, such as mean filter and median filter. User can also define custom filters by implementing the `IFilter` interface.

Sensor Data Processing

The `SignalListener` in GaitLib is used to receive data from the device's accelerometer, gyroscope, and location services. Based on the window size, `SignalListener` caches recent sensor readings, and then `CadenceDetector` and `GaitClassifier` can retrieve the values as input to their algorithms.

Logging Component

Relevant classes implements the `ILoggable` interface to facilitate logging of the raw sensor data and the results of cadence estimation and gait classification. The logs are exported as CSV files to the device storage by the logger.

EXAMPLE APPLICATION

In this section, we describe the workflow of setting up an application that uses GaitLib. Suppose Jim wants to create an application that represents the user's gait by playing a different sound clip for each identified gait at the frequency determined by user's cadence.

In a service of the Android app, Jim creates a `GaitAnalysis` object. Then, he registers its `SignalListener` with Android's sensor manager. Next, he defines the actions to take on the cadence and gait values received through a listener; in this case, he sends the values to another service that plays the sound clips and he displays the values on the screen (Figure 3). Finally, he starts gait analysis by calling the method in `GaitAnalysis`.

By defining other actions on the cadence and gait values received from GaitLib, developers can create applications that send the real-time information to external devices or use the information within the app (Figure 1).

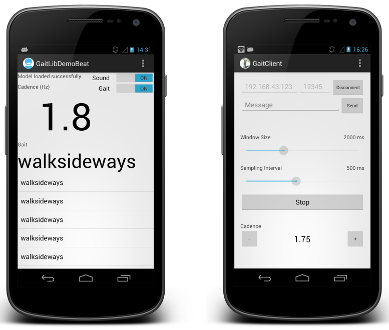

Figure 3. Left: the example app showing the current cadence and gait. Right: a demo app sending cadence via network with parameter control.

CUSTOMIZABLE FEATURES

To personalize the gait classification, or to classify using an alternative set of features, developers can build custom classifier models to replace the default model. First, use the companion GaitLogger app to record user's gait for different activities, then generate the features to be used for classification. Finally, train the model in Weka and load it onto the device. Figure 4 overviews this workflow.

In addition to a custom classifier model, to use an alternative feature set for classification, developers need to implement the algorithms for feature extraction and classification in a concrete `GaitClassifier`. Similarly, developers can implement additional concrete `CadenceDetector` to replace the default cadence estimation algorithm.

PERFORMANCE AND USABILITY

Performance: The default algorithms' performance was evaluated in [8, 11], and generally found to be competitive with the reported state of the art. We informally tested the power consumption of the example application, which uses these two algorithms, on three Galaxy Nexus devices and observed that the app consumed 45% more power than idling.

Usability: Beyond addressing all raised concerns, we evaluated GaitLib's usability through a post interview with our end-user team member (EU).

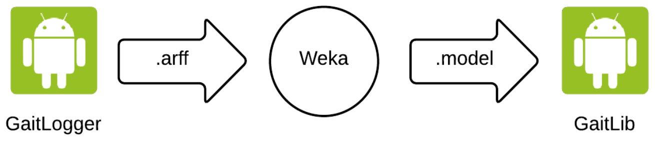

Figure 4. Workflow of training alternative classifier model.

EU used the library in two projects. In the first, he modified the example app to record cadence data for two related experiments, tweaked the window size, double checked the sampling rate, and modified the UI to keep track of the last 10 cadence values (for visibility and verification). He described that it only took "a day or half a day" to become familiar with the library. EU used 4 phones for additional accuracy. He reported the problems he encountered, including several crashes, to the development team. In retrospect, most of his effort was spent managing and analyzing the collected data, not setting up the application. EU said that without the library, he wouldn't have been able to run the experiments as he did, because the library let him make his own app without implementing the algorithm from scratch.

In his second project, EU used an updated version of GaitLib to create a demo for his work. He encountered none of the issues he had encountered before. Although we haven't confirmed the reasons for the earlier crashing on his app, we expect it is related to the operating system attempting to reduce power consumption when the screen is off.

CONCLUSION AND FUTURE WORK

In this paper, we presented GaitLib, a library for real-time mobile gait analysis on the Android platform. It supports detecting cadence and classifying gait continuously in real time, and it is extensible to incorporate alternative algorithms and different classifier choices. We developed the library through a participatory design approach, in which an author, primarily an end user, provided input in the design and gave feedback as he used the library in two occasions. Overall, it is an easy-to-use tool for developers to apply in many application areas, such as apps that motivate exercise and ones that communicate user's context to external devices.

GaitLib, along with user guides and example applications, is open-sourced and publicly available[1]. Future work includes testing GaitLib on a variety of Android devices with different sensor specifications to ensure consistency and robustness. The classifier model training process could be further simplified, e.g., adding support for training simple classifier models on the device. A built-in user profile management could be beneficial as, for example, the average stride length of a person can be used in estimation of speed when GPS is unavailable. Platform support could be expanded to iOS and Windows Phone for app developers to provide seamless experience of their products across different devices.

ACKNOWLEDGEMENTS

This work was funded in part by the Natural Sciences and Engineering Research Council of Canada (NSERC).

[1] `https://github.com/m-wu/gaitlib`

REFERENCES

1. Bao, L., and Intille, S. Activity Recognition from User-Annotated Acceleration Data. In *Pervasive Computing*, A. Ferscha and F. Mattern, Eds. Springer Berlin / Heidelberg, 2004, 1–17.
2. Begg, R. K., Palaniswami, M., Owen, B., and Member, S. Support vector machines for automated gait classification. *IEEE transactions on bio-medical engineering 52*, 5 (May 2005), 828–38.
3. Consolvo, S., Klasnja, P., McDonald, D. W., Avrahami, D., Froehlich, J., LeGrand, L., Libby, R., Mosher, K., and Landay, J. A. Flowers or a robot army? In *Proc. UbiComp '08*, ACM Press (Sept. 2008), 54.
4. Derawi, M. O., Nickel, C., Bours, P., and Busch, C. Unobtrusive User-Authentication on Mobile Phones Using Biometric Gait Recognition. In *IIH-MSP 2010*, IEEE (Oct. 2010), 306–311.
5. Frank, J., Mannor, S., and Precup, D. Activity recognition with mobile phones. In *Machine Learning and Knowledge Discovery in Databases*. Springer Berlin / Heidelberg, 2011, 630–633.
6. Hall, M., National, H., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. The WEKA data mining software: an update. *SIGKDD Explorations 11*, 1 (2009), 10–18.
7. Iso, T., and Yamazaki, K. Gait analyzer based on a cell phone with a single three-axis accelerometer. In *Proc. MobileHCI '06*, ACM Press (2006), 141.
8. Karuei, I., Schneider, O. S., Stern, B., Chuang, M., and MacLean, K. E. Rrace: Robust realtime algorithm for cadence estimation. *Pervasive and Mobile Computing* (2013).
9. Marsan, R. J. Weka for Android, 2011. `https://github.com/rjmarsan/Weka-for-Android`.
10. Nishiguchi, S., Yamada, M., Nagai, K., Mori, S., Kajiwara, Y., Sonoda, T., Yoshimura, K., Yoshitomi, H., Ito, H., Okamoto, K., et al. Reliability and validity of gait analysis by android-based smartphone. *Telemedicine and e-Health 18*, 4 (2012), 292–296.
11. Schneider, O. S., MacLean, K. E., Altun, K., Karuei, I., and Wu, M. Real-time gait classification for persuasive smartphone apps: structuring the literature and pushing the limits. In *Proc. IUI 2013*, ACM (2013), 161–172.
12. Siirtola, P., and Röning, J. Recognizing human activities user-independently on smartphones based on accelerometer data. *International Journal of Interactive Multimedia & Artificial Intelligence 1*, 5 (2012).
13. Yang, C.-C., Hsu, Y.-L., Shih, K.-S., and Lu, J.-M. Real-time gait cycle parameter recognition using a wearable accelerometry system. *Sensors (Basel, Switzerland) 11*, 8 (Jan. 2011), 7314–26.
14. Yang, J. Toward physical activity diary: motion recognition using simple acceleration features with mobile phones. In *Proc. IMCE'09* (2009), 1–9.