βTap: Back-of-Device Tap Input with Built-in Sensors

Emilio Granell
PRHLT, Universitat Politècnica de València
`egranell@dsic.upv.es`

Luis A. Leiva
Sciling
`name@sciling.com`

MobileHCI '17, September 4–7, 2017, Vienna, Austria
ACM 978-1-4503-5075-4/17/09.
https://doi.org/10.1145/3098279.3125440

Abstract
We present βTap, a Back-of-device (BoD) tap detection software for mobile devices that uses commodity sensors, without the need to instrument the device. Although just basic interactions are supported (namely, single and double taps), βTap is highly accurate and performance-friendly, since it uses a low-cost yet highly discriminative set of features. Our software is publicly available at the Google Play Store, so that others can build upon our work.

Author Keywords
BoD interaction; Tap-based input; Sensors; Machine learning; Feature selection; Software

ACM Classification Keywords
H.5.2 [User Interfaces]: Input devices and strategies; Interaction styles; I.5.4 [Pattern Recognition]: Applications

Introduction
This demonstration paper complements a recent publication on Back-of-device (BoD) tap input detection [3]. Overall, BoD interaction enables eyes-free indirect input through an alternative set of interactions to augment (rather than replace) touchscreen input. For example, it can address severe occlusion problems on

small touchscreens [1], has shown to be useful in increasing privacy by preventing shoulder-surfing attacks [8], and can be used to control screen sharing apps such as multi-player games [2] or even to predict users' intention by the way they hold the device [10].

BoD interactions are largely dependent on sensors. Recently, it has been shown that this input modality can be accomplished on today's mobile devices with their built-in hardware like accelerometers, microphones, or gyroscopes [15, 17]. For this, many different features derived from sensor readings have been proposed, like sound volume, device motion, or frequency analysis. However, we have previously shown that not all sensor measurements have an equal value for detecting BoD interactions reliably and efficiently [3]. And while modern smartphones are incorporating low-power coprocessors for managing the built-in sensors, energy consumption is still a matter of great concern [14, 16, 17].

In this demonstration, we present βTap, the accompanying software of our most recent work on BoD tap input detection [3]. βTap relies on current smartphone sensors, without the need to instrument the device. Although just basic interactions are supported (namely, single and double taps), βTap is highly accurate and performance-friendly, since it uses a low-cost yet highly discriminative set of features, removing thus unnecessary computations that could drain battery life after extended periods of use. Further, βTap is segmentation-free which allows for practical, real-world use cases. Our software is publicly available, so that others can build upon our work.

Related Research

Generally, sensing the mobile device with additional hardware has been very popular for enabling BoD interactions. In early work, Hinckley and Horvitz [5], instrumented a PocketPC with external sensors to demonstrate novel interactions that could be supported. Other researchers have proposed to use a touchpad [1] or touch-sensitive surface [6], a stethoscope [9], tactile landmarks [2], or an external accelerometer attached to the device [7]. However, today's smartphones already include sophisticated built-in sensors, such as gyroscope, accelerometer, or gravity sensors that can be exploited to detect BoD tap input.

Previous works that focus on BoD interaction using current hardware and sensors include e.g. ForceTap [4], JerkTilts [11], TimeTilt [13], TapBack [12], BackPat [15], and BackTap [18]. Altogether, these works have relied on an unknown or a relatively arbitrary choice of sensors and features. Furthermore, to ensure accurate BoD tap event *segmentation*, researchers have approached this problem either from manual or ad-hoc perspectives. Our previous work [3] was the first systematic examination in this regard. As a result, we have released βTap, a highly accurate and performance-friendly BoD tap-based input detection software. Concretely, βTap has been released as an Android service. This way, developers simply have to instantiate the service in their own apps. In the remainder of this paper we describe our implementation, followed by a number of use cases that illustrate a range of applications that can be developed with the above-mentioned service.

Implementation

βTap is implemented as a background service that abstracts the logic for BoD tap input detection, based on an optimized feature set and a generalized linear model classifier. The reader may consult our previous publication [3] to get into the technical details.

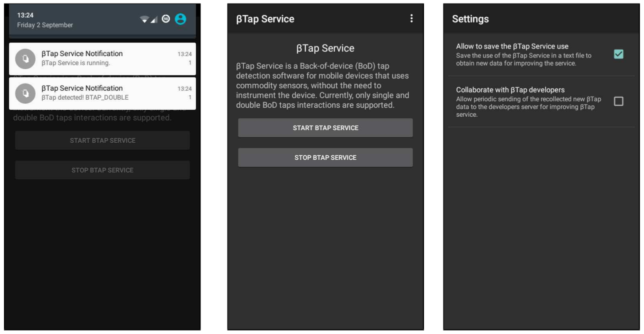
The service does not provide a dedicated user interface, this way developers simply have to instantiate the service in their own apps. Moreover, the service runs in a single background thread. This allows it to detect BoD tap input without affecting the user interface's responsiveness.

In order to avoid interfering with unintended motion data, the service does not read the device sensors unless some application requests using the service. This also helps to reduce energy consumption to a great extent, since BoD tap detection is performed only when an app explicitly requires it.

Our implementation provides single and double BoD taps detection off-the-shelf. Therefore, the service can be used to handle basic interactions such as controlling a music player or selecting text onscreen. The service can also be used to complement traditional interactions such as muting an incoming call or playing one-button games. At the moment, the software is available for the Android operating system and can be downloaded from https://btap.tech. It also can be installed straight from the Google Play Store at http://goo.gl/3X8hAp.

Usage

βTap is available to other applications as a background service. Optionally, βTap includes an activity that allows users to test the service, without having to implement it in the first place. Moreover, through this activity users can decide whether to collaborate sending periodically usage data to improve the service. Figure 1 shows some screenshots of this activity.



(a) Service notifications **(b)** Testing activity **(c)** Settings menu

Figure 1: The βTap service informs the user via notifications (a). It also allows the user to test the service (b). The configuration menu (c) allows developers to store the sensor data and to share them with the βTap service developers.

The process that a developer should follow when creating a βTap-based app is the standard for using bound services. In short, they should follow the following guidelines:

1. Create a **`ServiceConnection`** and bind the βTap service when creating the main activity.
2. Start the service when required.
3. Handle **`BTAP_SINGLE`** or **`BTAP_DOUBLE`** events.
4. Stop the service when no longer required.
5. Unbind the service when destroying the activity.

In https://btap.tech we provide more information regarding usage details, including a tutorial and a number of demonstration apps.

Example Applications

The following is a set of applications we have already developed with βTap, in order to showcase some of the possibilities where BoD tap-based input can be of help. All applications are free and open-source software (FOSS), so they can be publicly downloaded at no cost.

βTap Camera

This is a sample application that showcases the basic use of Android's `Camera2` API, available at https://developer.android.com/samples/Camera2Basic/. We have instrumented the application with βTap so that pictures can be taken when tapping on the back of the device. This not only allows for a more comfortable way of taking pictures with a single hand, but also prevents unintended touches on the screen because of the hand's palm; see Figure 2b. In this application, a single BoD tap triggers the camera focus and then takes a picture, which is then saved to the device's internal storage.

(a) App screenshot **(b)** Non-BoD capable **(c)** BoD capable

Figure 2: The βTap camera app (a) allows for a more comfortable holding; cf. (b) *vs.* (c).

βTap Flutter Cow

Available at https://play.google.com/store/apps/details?id=com.quchen.flappycow. This is a side-scrolling one-button game aking FlappyBird[1] or Badland.[2] The gameplay is fairly simple. The objective is to direct a flying cow, who moves continuously to the right, between sets of obstacles. The cow briefly flaps upward each time the player taps on the case of the device. If the case is not tapped, the cow falls because of gravity. If the player touches any obstacle, they lose. Here, BoD interaction allows for one-handed playing, freeing thus the other hand. Figure 3 depicts some screenshots of this game.

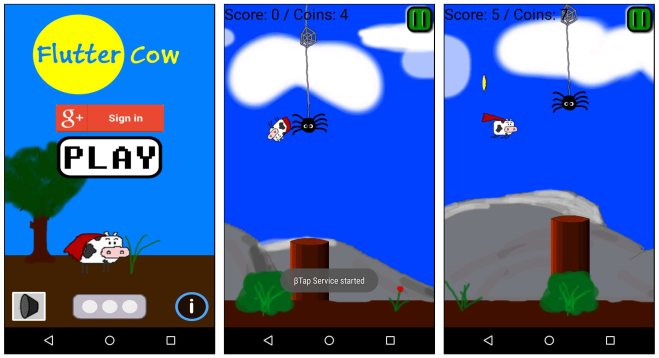


Figure 3: Screenshots of the βTap flutter cow game.

βTap Music Player

Available at https://play.google.com/store/apps/details?id=music.uamp. This is a sample application that showcases the basic use of Android's `MediaPlayer` API across a range of devices, including e.g. tablets, Android Wear, and

[1] http://flappybird.io
[2] http://badlandgame.com

Google Cast devices. Here, βTap is used to play/pause a song with a single tap, and advance to the next song with a double tap. It can also be used to play/stop casting to a nearby TV. These basic interactions allow e.g. for full screen visibility. Figure 4 shows some screenshots of this application.

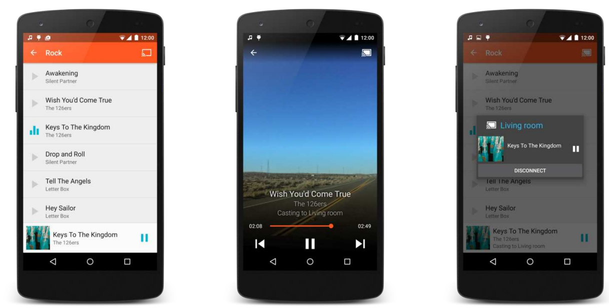


Figure 4: Screenshots of the βTap music player.

Conclusion

βTap is a reliable software for detecting BoD tap-based input on current smartphones using commodity sensors. The value of βTap lies in the fact that we use low-cost yet highly discriminative features. In general, selecting the right features can mean a difference between low performance with long computation times and high performance with short computation times. We have made our software public so that others can build upon our work. The software can be easily incorporated into production-ready applications, as developers simply have to instantiate a background activity that abstracts the logic for BoD tap input detection.

ACKNOWLEDGMENTS

This work was partially supported by SmartWays (MINECO, RTC-2014-1466-4) and CoMUN-HaT (MINECO/FEDER, TIN2015-70924-C2-1-R).

REFERENCES

1. Baudisch, P., and Chu, G. Back-of-device interaction allows creating very small touch devices. In *Proc. CHI* (2009).
2. Corsten, C., Cherek, C., Karrer, T., and Borchers, J. HaptiCase: Back-of-device tactile landmarks for eyes-free absolute indirect touch. In *Proc. CHI* (2015).
3. Granell, E., and Leiva, L. A. Less is more: Efficient back-of-device tap input detection using built-in smartphone sensors. In *Proc. ISS* (2016).
4. Heo, S., and Lee, G. ForceTap: extending the input vocabulary of mobile touch screens by adding tap gestures. In *Proc. MobileHCI* (2011).
5. Hinckley, K., and Horvitz, E. Toward more sensitive mobile phones. In *Proc. UIST* (2001).
6. Holman, D., Hollatz, A., Banerjee, A., and Vertegaal, R. Unifone: Designing for auxiliary finger input in one-handed mobile interactions. In *Proc. TEI* (2013).
7. Hudson, S. E., Harrison, C., Harrison, B. L., and LaMarca, A. Whack gestures: inexact and inattentive interaction with mobile devices. In *Proc. TEI* (2010).
8. Leiva, L. A., and Catalá, A. BoD Taps: An improved back-of-device authentication technique on smartphones. In *Proc. MobileHCI* (2014).

9. Lopes, P., Jota, R., and Jorge, J. A. Augmenting touch interaction through acoustic sensing. In *Proc. ITS* (2011).
10. Mohd Noor, M. F. Exploring back-of-device interaction. In *Adj. Proc. UIST* (2013).
11. Oakley, I., and Park, J. A motion-based marking menu system. In *Proc. CHI EA* (2007).
12. Robinson, S., Rajput, N., Jones, M., Jain, A., Sahay, S., and Nanavati, A. TapBack: Towards richer mobile interfaces in impoverished contexts. In *Proc. CHI* (2011).
13. Roudaut, A., Baglioni, M., and Lecolinet, E. Timetilt: using sensor-based gestures to travel through multiple applications on a mobile device. In *Proc. INTERACT*, vol. 5726 of *LNCS*. 2009.
14. Saeedi, R., Schimert, B., and Ghasemzadeh, H. Cost-sensitive feature selection for on-body sensor localization. In *Adj. Proc. UbiComp* (2014).
15. Seipp, K., and Devlin, K. BackPat: One-handed off-screen patting gestures. In *Proc. MobileHCI* (2014).
16. Serrano, M., Lecolinet, E., and Guiard, Y. Bezel-Tap gestures: Quick activation of commands from sleep mode on tablets. In *Proc. CHI* (2013).
17. Zhang, C., Guo, A., Zhang, D., Southern, C., Arriaga, R., and Abowd, G. BeyondTouch: Extending the input language with built-in sensors on commodity smartphones. In *Proc. IUI* (2015).
18. Zhang, C., Parnami, A., Southern, C., Thomaz, E., Reyes, G., Arriaga, R., and Abowd, G. D. BackTap: Robust four-point tapping on the back of an off-the-shelf smartphone. In *Adj. Proc. UIST* (2013).