DoubleFlip: A Motion Gesture Delimiter for Mobile Interaction

Jaime Ruiz*
University of Waterloo
Waterloo, ON, Canada
jgruiz@cs.uwaterloo.ca

Yang Li
Google Research
Mountain View, CA, USA
yangli@acm.org

Figure 1: An illustration of the DoubleFlip gesture. In this figure, the user holds the phone right-handed. He rotates the phone along its long side so that the phone screen is away and then back.

ABSTRACT
To make motion gestures more widely adopted on mobile devices it is important that devices be able to distinguish between motion intended for mobile interaction and everyday motion. In this paper, we present DoubleFlip, a unique motion gesture designed as an input delimiter for mobile motion-based interaction. The DoubleFlip gesture is distinct from regular motion of a mobile device. Based on a collection of 2,100 hours of motion data captured from 99 users, we found that our DoubleFlip recognizer is extremely resistant to false positive conditions, while still achieving a high recognition rate. Since DoubleFlip is easy to perform and unlikely to be accidentally invoked, it provides an always-active input event for mobile interaction.

Author Keywords
Motion gestures, sensors, mobile interaction.

ACM Classification
H5.2. User Interfaces--Input devices and strategies.

General Terms
Design

INTRODUCTION
A major technical barrier for adopting motion-gesture-based interaction is the need for high recognition rates with low false positive conditions. This obstacle limits the potential of good interaction design since many proposed physical motions (e.g., flipping [3] and tilting [3,4,6,9]) are indistinguishable from everyday motions (i.e., walking). Hence, these gestures suffer from the inherent difficulty that they generate a large number of false positives. In addition, Rico and Brewster [10] reported that users rated motion gestures more socially acceptable when the gesture looks or feels similar to everyday motion, suggesting that distinctions between desired motion gestures and everyday motion may not necessarily exist.

While modeless interaction for motion gestures is ideal, it is difficult to achieve sufficient reliability to be practical for general interaction situations. The dilemma that we encounter is the same for other types of interactions, i.e., inking versus gesturing in pen-based user interfaces [5,7]. Previous work in delimiting normal motion from motion input has relied on hardware buttons [3,9] or additional sensors in order to determine the context of use [6]. Although successful, these mode-switching techniques require a designer to use interaction context (which is often hard to infer) or a user to switch input modalities (i.e. from motion input to pressing or touching a button).

We designed DoubleFlip to tackle the challenge of separating gestural input from normal motion. DoubleFlip is a unique motion gesture that acts as a delimiter for other motion gestures, and hence, separates normal mobile phone motion from a user's intended input. The gesture gives users the control to activate motion gestures without any hardware modifications to existing devices. The gesture is quick to perform and can be performed in a limited amount of physical space. We are unaware of any previous work that proposes the use of a distinct motion gesture to delimit motion gesture input.

As part of this work, we implemented a recognizer for the DoubleFlip gesture. Based on a collection of over 2,100 hours of motion data captured from the phones of 99 volunteers, we found that our DoubleFlip recognizer is extremely resistant to false positives (less than one false positive per eight hours of use). Thus, the DoubleFlip ges-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
CHI 2011, May 7–12, 2011, Vancouver, BC, Canada.
Copyright 2011 ACM 978-1-4503-0267-8/11/05....$10.00.

* The work was conducted during an internship at Google Research.

ture and recognizer can be used as a building block for other systems to provide motion-gesture-based interactions.

DESIGNING DOUBLEFLIP

Our goal for designing an input delimiter for motion gestures was inspired by an informative study conducted with 20 smartphone users on the potential uses of motion gestures. Although we instructed our participants to imagine a "magical" device capable of perfectly inferring their intention, participants preferred to have the explicit control over when motion input should be enabled. As one participant stated, "It's more of an assurance thing. If you said it worked perfect I still wouldn't trust you."

We considered two options for a motion input delimiter: a hardware button and a dedicated delimiter gesture. The use of a hardware button was ruled out for three reasons: 1) smartphones differ in the number and placement of buttons, thus, restricting motion gestures to smartphones with an available button; 2) there was no consensus among participants on where the button should be placed; and 3) requiring a user to activate mode on the device used for input increases the likelihood of mistaken activation [7].

As a result, we set out to design a delimiter that is not dependent on a touchscreen or buttons but instead relies on the sensors commonly available in current smartphones. From our study, we decided that the motion gesture delimiter should meet the following requirements: 1) the gesture should be resistant to false positives; 2) the screen must be visible to the user after completing the delimiter gesture so that they can continue to interact with the screen; 3) the physical space required to perform the gesture should be minimal to allow the gesture to be performed surreptitiously and in crowded spaces; 4) the gesture should be quick and easy to perform; and 5) the gesture should be distinct from any gestures that a user/developer would likely want to incorporate into an application.

Although the set of possible motion gestures a user can perform with a smartphone is large, our design requirements limit the number of suitable candidates for an input delimiter. Gestures that require multiple axes or complex movements were dismissed due to temporal and physical constraints imposed by our requirements. Simple gestures (e.g., shaking, flicking, and sudden impacts) were also dismissed because our participants often selected these gestures to interact with various applications. Using these gestures as an input delimiter would limit the ability of the designer to incorporate these gestures in their applications. In addition, these gestures are often indistinguishable from ambient motion which would result in a high number of false positives.

As shown in Figure 1, the DoubleFlip delimiter is performed by quickly rotating the wrist such that the phone's display is facing away from the user and back to the original position with the display of the phone facing the user. The design of DoubleFlip makes it resilient to false positives. During normal use, users commonly interact with the front of the device (due to the location of the display), which makes this gesture unlikely to be triggered accidently. The need to flip the phone back so the display is facing the user also allows us to fulfill our requirement that the display be visible to the user after performing the gesture. Finally, since the gesture can be performed with the simple rotation of the wrist and no arm movement, the gesture is compact and requires no more physical space than holding the phone for normal use. The gesture is also easy and quick to perform.

IMPLEMENTATION OF DOUBLEFLIP

The DoubleFlip gesture recognizer was developed in Java using the Android SDK [1] for use on the Nexus One phone with a Qualcomm QSD 8250 1 GHz processor, a three-axis accelerometer, and an orientation sensor.

Sensor input is matched to the DoubleFlip gesture using Dynamic Time Warping (DTW). DTW is a dynamic programming algorithm that measures the similarity of two time series with temporal dynamics [8] when given the function for calculating the distance between the two time samples. See [11] for more information.

Our implementation of the DoubleFlip gesture recognizer uses a weighted Euclidean distance function for calculating the distance between the quantized time series of acceleration and of orientation to the DTW templates. Since the display facing away from the user is an important characteristic of the gesture, we place a higher weight on the Euclidean distance for orientation values and a lower weight on the distance for acceleration. Giving a higher weight to the orientation values minimizes the probability that ambient motion will be recognized as a DoubleFlip gesture. If the calculated distance is less than a specified threshold, the recognizer reports a match for the DoubleFlip gesture.

Using data from an informal study benchmarking the number of DTW templates vs. computation time, we limited the number of DTW templates to 20 which were created from samples collected during a pilot study.

DETERMING OPTIMAL RECOGNITION THRESHOLDS

In this section, we present descriptions and results of our two experiments to find the optimal thresholds for our implementation. First, we describe an experiment to determine the appropriate threshold to minimize the number of false positives. This is followed by an experiment to find the balance between false negative and false positive conditions.

Methods

In both experiments data was collected using custom software written using the Android SDK to log the data stream received from a device's accelerometer and orientation sensors. The software periodically uploaded the recorded data to a secure server whenever the display was

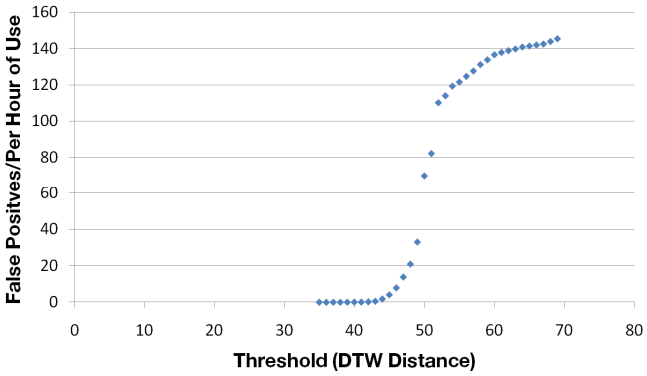

Figure 3: The DoubleFlip false positive rates versus distance threshold level.

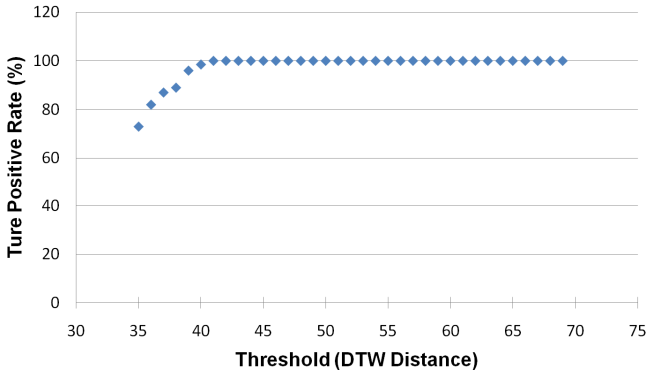

Figure 2: The DoubleFlip true positive rate versus distance threshold level.

off. The ability to remotely collect the data allowed us to recruit a large number of participants who do not have to be physically present.

To analyze the dataset, we developed a tool to replay the recorded sensor streams. As the sensor stream was replayed, gesture recognition was performed and each detection of the DoubleFlip gesture was flagged. To investigate the effect of varying the threshold of the recognizer, the program iterated through the complete dataset by incrementing the threshold distance for each iteration.

Experiment One: False Positives

The goal of our first experiment was to determine the appropriate threshold such that the number of false positives is minimized. Participants were recruited by sending an email to an internal list of employees at a high-tech company asking for volunteers. The email directed the employees to an internal webpage that provided information on the software and a link to download the application. Participants were required to use a Google Nexus One smartphone as their primary mobile device and were asked to run the software for at least 24 hours.

Ninety-nine participants installed and ran the software long enough so that at least one log was uploaded to our server. In total, 2,169 hours of sensor data was collected.

Results

Figure 3 illustrates the effect of varying the distance on the false positive rate. As the threshold increases, the number of false positive conditions increases dramatically. However, DoubleFlip benefits from a low false positive condition even at larger distance thresholds, which implies its inherent distinctiveness from normal use motion.

Experiment Two: True Positives

Our first experiment showed that a higher threshold distance will lead to more false positives. However, a lower distance threshold tends to result in more false negatives, which can result in user frustration. Therefore, our second experiment is designed to find a balance between false positive and false negative conditions.

Our second experiment required participants to be physically present during the experiment. Twenty volunteers were recruited via the same email list as the first experiment. At the beginning of the study, the researcher described the procedure and demonstrated the DoubleFlip gesture. The researcher then handed the participant a phone running the custom software and asked the participant to perform the DoubleFlip gesture twenty times, with approximately a five second pause after each gesture. Upon completion, the participants were asked to rate how easy the gesture was to perform using a five-point Likert. A trial took approximately five minutes to complete.

Results

The relationship between the true positive rate and the threshold level is shown in Figure 2. Based on the collected data, the true positive rate quickly converges to 100%. Using the ambient motion false positive rates form our first experiment and the recognition rates of our second experiment, we plotted the ROC curve to examine the relationship between false and true positives as the threshold is adjusted (see Figure 4). Based on the ROC curve, the optimal distance threshold achieved a 100% true positive rate with only one false positive per 8 hours of mobile phone use.

Our study participants highly rated the DoubleFlip gesture with an average rating of 4.6 (1 for difficult and 5 for easy). Only one participant rated the DoubleFlip gesture lower than "somewhat easy" and indicated that the gesture may cause fatigue if the gesture is performed frequently in succession for a long period of time.

DISCUSSION

Our goal was to develop a novel delimiter for motion-gesture-based interaction on mobile phones. Based on our interviews with smartphone users, we compiled a list of requirements that an effective delimiter gesture should meet. A delimiter gesture should be resilient to false positives and quick to perform, require limited physical space, and allow the screen to be visible to the user upon completing the gesture.

By testing our implementation of the DoubleFlip gesture recognizer against the sensor streams of naturally occurring motion, we found our DoubleFlip recognizer is resistant to false positives while still retaining a high true positive rate. Prior methods often rely on the current state (or context) of the system in order to reduce false positives (e.g., interpreting a shake as a shuffle gesture when playing music). In contrast, the DoubleFlip gesture and its detection are state-independent. Being state-independent is an important feature and it allows DoubleFlip to employ an "always on" listening strategy, which is a key requirement for any input delimiter.

The ability to distinguish the DoubleFlip gesture from normal motion also enables the gesture to be used as either the prefix or postfix to delimit an input motion gesture from everyday motion. For example, assume we want to use a shaking gesture to redial the last number called. Without the DoubeFlip gesture we are susceptible to high false positive conditions given the commonality of the motion. By incorporating the DoubleFlip gesture into the shake gesture we negate any false positives during normal usage. The user simply performs the DoubleFlip gesture and immediately performs the shake gesture, causing the phone to redial. When DoubleFlip is used as a postfix, the user would instead shake the phone and immediately perform the DoubleFlip gesture.

Appending the DoubleFlip gesture either before or after another gesture provides the user with a fluid method to enable motion gesture mode and perform the desired action. Similar to the benefits observed from using the Scriboli pigtail [5], the use of the DoubleFlip gesture may provide performance benefits over traditional mode switching techniques.

In addition to the quasi-mode switching[1] discussed in the previous example, the DoubleFlip gesture can be used to indicate the beginning and ending of gestural input. This approach allows the user to perform several sequential gestural commands. This approach is particularly useful in situations where the user would like to continuously pan or scroll content by tilting the phone, a common command in numerous contexts (e.g., navigating in a map application).

CONCLUSION & FUTURE WORK

In this paper, we presented DoubleFlip, a unique motion gesture that was designed as an input delimiter for mobile motion gestures. Our experiments showed that the DoubleFlip gesture is extremely resistant to false positive conditions, while still achieved a high recognition rate. Since DoubleFlip is easy to perform and unlikely to be accidentally invoked, it provides an always-active input event for mobile interaction.

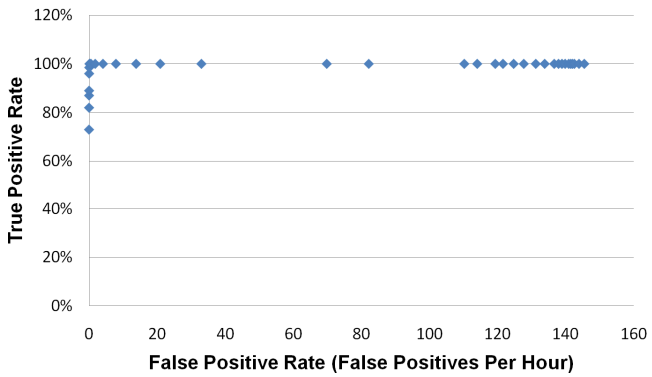

Figure 4: The ROC Curve of the DF recognizer.

Our future work will focus on more extensive evaluation of the DoubleFlip gesture as both a prefix and postfix gesture. We also plan on exploring and evaluating different methods to instruct and describe motion gestures to users. An implementation of the Double Flip gesture has been incorporated into Google Gesture Search [2] as an always-on activation method.

REFERENCES

1. *Android Open Source Project*. Google Inc.
2. *Google Gesture Search*. http://gesturesearch.googlelabs.com
3. Bartlett, J.F. Rock 'n' Scroll Is Here to Stay. *IEEE Comput. Graph. Appl. 20*, 3 (2000), 40–45.
4. Harrison, B.L., Fishkin, K.P., Gujar, A., Mochon, C., and Want, R. Squeeze me, hold me, tilt me! An exploration of manipulative user interfaces. *Proceedings of CHI '98*, ACM Press/Addison-Wesley Publishing Co. (1998), 17–24.
5. Hinckley, K., Baudisch, P., Ramos, G., and Guimbretiere, F. Design and analysis of delimiters for selection-action pen gesture phrases in scriboli. *Prococedings of CHI '05*, ACM (2005), 451–460.
6. Hinckley, K., Pierce, J., Sinclair, M., and Horvitz, E. Sensing techniques for mobile interaction. *Proceedings of UIST '00*, ACM (2000), 91–100.
7. Li, Y., Hinckley, K., Guan, Z., and Landay, J.A. Experimental analysis of mode switching techniques in pen-based user interfaces. *Proceedings of CHI '05*, ACM (2005), 461–470.
8. Myers, C. and Rabiner, L. A comparative study of several dynamic time-warping algorithms for connected word recognition. *The Bell System Tech Journal 60*, 7 (1981), 1389-1409.
9. Rekimoto, J. Tilting operations for small screen interfaces. *Proceedings of UIST '96*, ACM (1996), 167–168.
10. Rico, J. and Brewster, S. Usable gestures for mobile interfaces: evaluating social acceptability. *Proceedings of CHI 2010*, ACM (2010), 887–896.
11. Wobbrock, J.O., Wilson, A.D., and Li, Y. Gestures without libraries, toolkits or training: a $1 recognizer for user interface prototypes. *Proc. of UIST '07*, ACM (2007), 159–168.

[1] In quasi-mode switching techniques, the system state is automatically returned to the original state. Therefore, the user is not required to deactivate the mode by performing another mode switch.